

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»

Виконала:

Студентка групи АІ-202
Гребенік Анжеліка Олександрівна

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 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:

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;

T1 и T2

```
grebenik_anzhelika@vpsj3leQ:~  
Type "help" for help.  
  
grebenik_anzhelika=> start transaction;  
START TRANSACTION  
grebenik_anzhelika=> select xmin,xmax,a_id from auto;  
  xmin | xmax | a_id  
-----+-----+-----  
  2461 |    0 |    2  
  2518 | 2520 |    1  
(2 rows)  
  
grebenik_anzhelika=> insert into auto values(3,'Cadillac',2017);  
INSERT 0 1  
grebenik_anzhelika=> select * from auto;  
 a_id | name      | year  
-----+-----+-----  
    2 | Audi      | 2006  
    1 | BMW 5     | 2010  
    3 | Cadillac  | 2017  
(3 rows)  
  
grebenik_anzhelika=> commit;  
COMMIT  
grebenik_anzhelika=>
```

```
grebenik_anzhelika@vpsj3leQ:~  
grebenik_anzhelika@91.219.60.189's password:  
Last login: Wed May  5 03:45:17 2021 from 46.149.53.142  
[grebenik_anzhelika@vpsj3leQ ~]$ psql  
psql (9.5.25)  
Type "help" for help.  
  
grebenik_anzhelika=> start transaction;  
START TRANSACTION  
grebenik_anzhelika=> select * from auto;  
 a_id | name      | year  
-----+-----+-----  
    2 | Audi      | 2006  
    1 | BMW 5     | 2010  
(2 rows)  
  
grebenik_anzhelika=> select * from auto;  
 a_id | name      | year  
-----+-----+-----  
    2 | Audi      | 2006  
    1 | BMW 5     | 2010  
    3 | Cadillac  | 2017  
(3 rows)  
grebenik_anzhelika=>
```

– T2 – постійний перегляд вмісту таблиці

– Т3 – видалення рядку з наступною відміною цієї операції;
Т3 и Т2

```
grebenik_anzhelika@vpsj3leQ:~  
login as: grebenik_anzhelika  
grebenik_anzhelika@91.219.60.189's password:  
Last login: Wed May 5 03:46:38 2021 from 46.149.53.142  
[grebenik_anzhelika@vpsj3leQ ~]$ psql  
psql (9.5.25)  
Type "help" for help.  
  
grebenik_anzhelika=> start transaction;  
START TRANSACTION  
grebenik_anzhelika=> delete from auto where a_id=3;  
DELETE 1  
grebenik_anzhelika=> rollback;  
ROLLBACK  
grebenik_anzhelika=> commit;  
WARNING: there is no transaction in progress  
COMMIT  
grebenik_anzhelika=>   
  
grebenik_anzhelika@vpsj3leQ:~  
a_id | name | year  
-----+-----+-----  
2 | Audi | 2006  
1 | BMW 5 | 2010  
3 | Cadillac | 2017  
(3 rows)  
  
grebenik_anzhelika=> select * from auto;  
a_id | name | year  
-----+-----+-----  
2 | Audi | 2006  
1 | BMW 5 | 2010  
3 | Cadillac | 2017  
(3 rows)  
  
grebenik_anzhelika=> select * from auto;  
a_id | name | year  
-----+-----+-----  
2 | Audi | 2006  
1 | BMW 5 | 2010  
3 | Cadillac | 2017  
(3 rows)  
  
grebenik_anzhelika=>   

```

– Т4 – зміна значення однієї з колонок рядка.

```
grebenik_anzhelika@vpsj3leQ:~  
Type "help" for help.  
  
grebenik_anzhelika=> start transaction;  
START TRANSACTION  
grebenik_anzhelika=> update auto set year=2019 where a_id=3;  
UPDATE 1  
grebenik_anzhelika=> commit;  
COMMIT  
grebenik_anzhelika=>   
  
grebenik_anzhelika@vpsj3leQ:~  
a_id | name | year  
-----+-----+-----  
2 | Audi | 2006  
1 | BMW 5 | 2010  
3 | Cadillac | 2017  
(3 rows)  
  
grebenik_anzhelika=> select * from auto;  
a_id | name | year  
-----+-----+-----  
2 | Audi | 2006  
1 | BMW 5 | 2010  
3 | Cadillac | 2017  
(3 rows)  
  
grebenik_anzhelika=> select * from auto;  
a_id | name | year  
-----+-----+-----  
2 | Audi | 2006  
1 | BMW 5 | 2010  
3 | Cadillac | 2019  
(3 rows)  
  
grebenik_anzhelika=>   

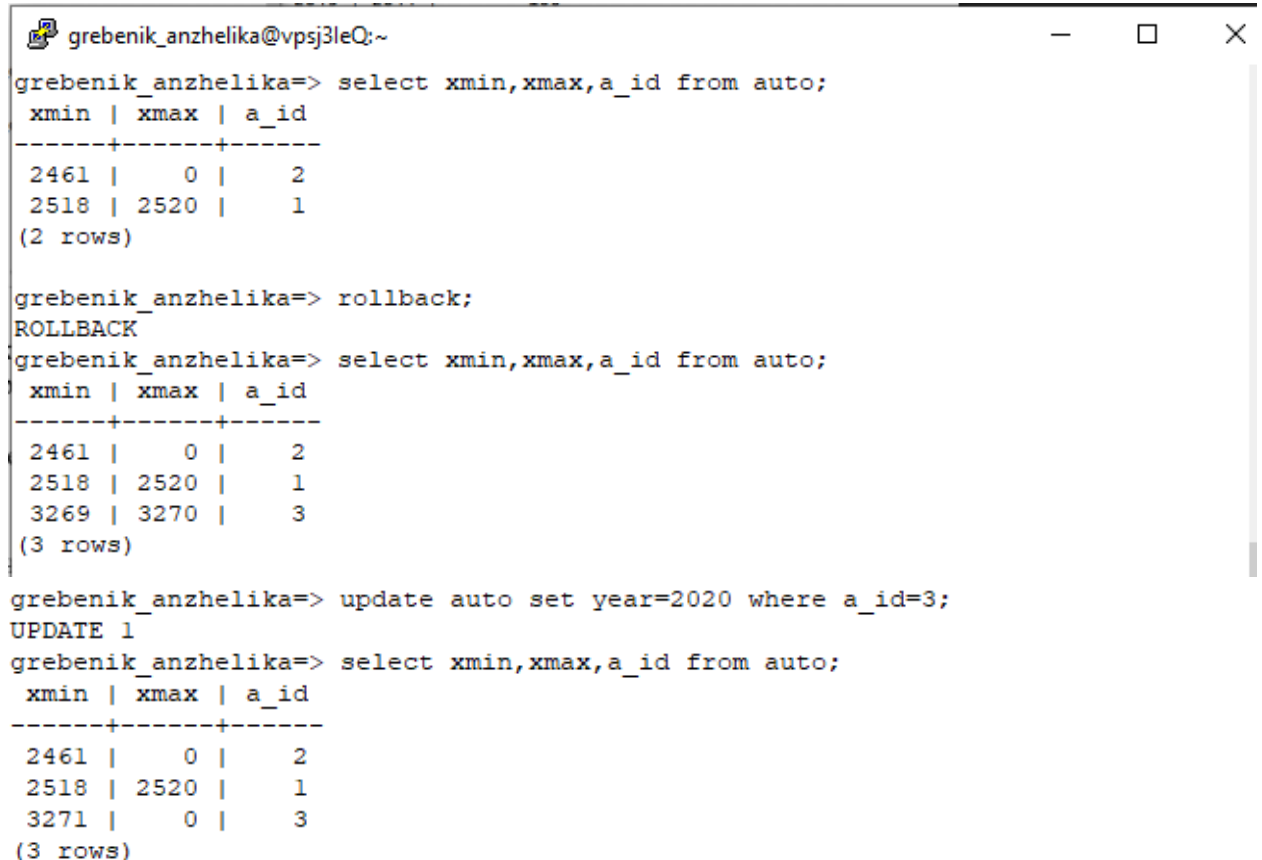
```

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

```
grebenik_anzhelika=> select xmin,xmax,a_id from auto;
xmin | xmax | a_id
-----+-----+-----
2461 |    0 |    2
2518 | 2520 |    1
(2 rows)
```

```
grebenik_anzhelika=> select xmin,xmax,a_id from auto;
xmin | xmax | a_id
-----+-----+-----
2461 |    0 |    2
2518 | 2520 |    1
3269 |    0 |    3
(3 rows)
```

```
grebenik_anzhelika=>
```



```
grebenik_anzhelika@vpsj3leQ:~
grebenik_anzhelika=> select xmin,xmax,a_id from auto;
xmin | xmax | a_id
-----+-----+-----
2461 |    0 |    2
2518 | 2520 |    1
(2 rows)

grebenik_anzhelika=> rollback;
ROLLBACK

grebenik_anzhelika=> select xmin,xmax,a_id from auto;
xmin | xmax | a_id
-----+-----+-----
2461 |    0 |    2
2518 | 2520 |    1
3269 | 3270 |    3
(3 rows)

grebenik_anzhelika=> update auto set year=2020 where a_id=3;
UPDATE 1

grebenik_anzhelika=> select xmin,xmax,a_id from auto;
xmin | xmax | a_id
-----+-----+-----
2461 |    0 |    2
2518 | 2520 |    1
3271 |    0 |    3
(3 rows)
```

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

Після заміни значення року та виконення `commit` змінилося значення `xmin`, а `xmax` змінило своє значення на 0

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

```
grebenik_anzhelika@vpsj3leQ:~$
login as: grebenik_anzhelika
grebenik_anzhelika@91.219.60.189's password:
Last login: Wed May 5 03:47:47 2021 from 46.149.53.142
[grebenik_anzhelika@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

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

```
grebenik_anzhelika@vpsj3leQ:~$
grebenik_anzhelika=> select relation,locktype,virtualtransaction,pid,mode,granted
from pg_locks where locktype='relation';
 relation | locktype | virtualtransaction | pid | mode | granted
-----+-----+-----+-----+-----+-----
11673 | relation | 23/1941 | 4837 | AccessShareLock | t
16729 | relation | 20/6095 | 13032 | RowExclusiveLock | t
11673 | relation | 8/27158 | 30304 | AccessShareLock | t
11673 | relation | 15/7259 | 2289 | AccessShareLock | t
16639 | relation | 24/5906 | 13111 | AccessShareLock | t
16639 | relation | 17/3324 | 30269 | AccessShareLock | t
16639 | relation | 17/3324 | 30269 | RowExclusiveLock | t
16729 | relation | 22/1715 | 13053 | RowShareLock | t
3455 | relation | 9/20521 | 12511 | AccessShareLock | t
2663 | relation | 9/20521 | 12511 | AccessShareLock | t
2662 | relation | 9/20521 | 12511 | AccessShareLock | t
2685 | relation | 9/20521 | 12511 | AccessShareLock | t
2684 | relation | 9/20521 | 12511 | AccessShareLock | t
2615 | relation | 9/20521 | 12511 | AccessShareLock | t
1259 | relation | 9/20521 | 12511 | AccessShareLock | t
16714 | relation | 18/8325 | 8045 | AccessShareLock | t
16714 | relation | 18/8325 | 8045 | RowExclusiveLock | f
16714 | relation | 14/5291 | 5332 | ShareRowExclusiveLock | f
16714 | relation | 10/11259 | 5355 | RowShareLock | t
--More--
```

SIX-IX

```
grebenik_anzhelika@vpsj3leQ:~$
login as: grebenik_anzhelika
grebenik_anzhelika@91.219.60.189's password:
Last login: Wed May 5 04:56:07 2021 from 46.149.53.142
[grebenik_anzhelika@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

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

grebenik_anzhelika@vpsj3leQ:~$
login as: grebenik_anzhelika
grebenik_anzhelika@91.219.60.189's password:
Last login: Wed May 5 04:56:49 2021 from 46.149.53.142
[grebenik_anzhelika@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

grebenik_anzhelika=> start transaction;
START TRANSACTION
grebenik_anzhelika=> lock table in share row exclusive mode;
ERROR: syntax error at or near "in"
LINE 1: lock table in share row exclusive mode;
^
grebenik_anzhelika=>
```

```
grebenik_anzhelika@vpsj3leQ:~$
grebenik_anzhelika=> select relation,locktype,virtualtransaction,pid,mode,granted
from pg_locks where locktype='relation';
 relation | locktype | virtualtransaction | pid | mode | granted
-----+-----+-----+-----+-----+-----
11673 | relation | 23/1941 | 4837 | AccessShareLock | t
11673 | relation | 20/6215 | 14736 | AccessShareLock | t
11673 | relation | 8/27158 | 30304 | AccessShareLock | t
11673 | relation | 15/7259 | 2289 | AccessShareLock | t
16729 | relation | 24/5920 | 14769 | RowExclusiveLock | t
16639 | relation | 17/3324 | 30269 | AccessShareLock | t
16639 | relation | 17/3324 | 30269 | RowExclusiveLock | t
3455 | relation | 9/20521 | 12511 | AccessShareLock | t
2663 | relation | 9/20521 | 12511 | AccessShareLock | t
2662 | relation | 9/20521 | 12511 | AccessShareLock | t
2685 | relation | 9/20521 | 12511 | AccessShareLock | t
2684 | relation | 9/20521 | 12511 | AccessShareLock | t
2615 | relation | 9/20521 | 12511 | AccessShareLock | t
1259 | relation | 9/20521 | 12511 | AccessShareLock | t
16714 | relation | 18/8325 | 8045 | AccessShareLock | t
16714 | relation | 18/8325 | 8045 | RowExclusiveLock | f
16714 | relation | 14/5291 | 5332 | ShareRowExclusiveLock | f
16714 | relation | 10/11259 | 5355 | RowShareLock | t
16714 | relation | 19/3441 | 8062 | AccessShareLock | t
--More--
```

SIX-IS

```
grebenik_anzhelika@vpsj3leQ:~  
login as: grebenik_anzhelika  
grebenik_anzhelika@91.219.60.189's password:  
Last login: Wed May 5 05:06:26 2021 from 46.149.53.142  
[grebenik_anzhelika@vpsj3leQ ~]$ psql  
psql (9.5.25)  
Type "help" for help.  
  
grebenik_anzhelika=> start transaction;  
START TRANSACTION  
grebenik_anzhelika=> lock table auto in share row exclusive mode;  
LOCK TABLE  
grebenik_anzhelika=>   
  
grebenik_anzhelika@vpsj3leQ:~  
login as: grebenik_anzhelika  
grebenik_anzhelika@91.219.60.189's password:  
Last login: Wed May 5 05:08:37 2021 from 46.149.53.142  
[grebenik_anzhelika@vpsj3leQ ~]$ psql  
psql (9.5.25)  
Type "help" for help.  
  
grebenik_anzhelika=> start transaction;  
START TRANSACTION  
grebenik_anzhelika=> lock table auto in row share mode;  
LOCK TABLE  
grebenik_anzhelika=>   
  
relation | locktype | virtualtransaction | pid | mode | gran  
-----  
11673 | relation | 23/1941 | 4837 | AccessShareLock | t  
11673 | relation | 8/27158 | 30304 | AccessShareLock | t  
11673 | relation | 15/7259 | 2289 | AccessShareLock | t  
11673 | relation | 22/1852 | 16163 | AccessShareLock | t  
16639 | relation | 17/3324 | 30269 | AccessShareLock | t  
16639 | relation | 17/3324 | 30269 | RowExclusiveLock | t  
16792 | relation | 27/1304 | 11361 | AccessShareLock | t  
16792 | relation | 27/1304 | 11361 | RowExclusiveLock | t  
16792 | relation | 6/68480 | 11329 | AccessShareLock | t  
16792 | relation | 6/68480 | 11329 | RowExclusiveLock | t  
3455 | relation | 9/20521 | 12511 | AccessShareLock | t  
2663 | relation | 9/20521 | 12511 | AccessShareLock | t  
2662 | relation | 9/20521 | 12511 | AccessShareLock | t  
2685 | relation | 9/20521 | 12511 | AccessShareLock | t  
2684 | relation | 9/20521 | 12511 | AccessShareLock | t  
2615 | relation | 9/20521 | 12511 | AccessShareLock | t  
1259 | relation | 9/20521 | 12511 | AccessShareLock | t  
16714 | relation | 18/8325 | 8045 | AccessShareLock | t  
16714 | relation | 18/8325 | 8045 | RowExclusiveLock | f
```

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.

```
grebenik_anzhelika@vpsj3leQ:~  
n block  
grebenik_anzhelika=> update auto set year=2011 where a_id=1;  
ERROR: current transaction is aborted, commands ignored until end of transaction block  
n block  
grebenik_anzhelika=> \q  
[grebenik_anzhelika@vpsj3leQ ~]$ psql  
psql (9.5.25)  
Type "help" for help.  
  
grebenik_anzhelika=> start transaction;  
START TRANSACTION  
grebenik_anzhelika=> set transaction isolation level read committed;  
SET  
grebenik_anzhelika=> select * from auto where a_id=1;  
a_id | name | year  
-----  
1 | BMW 5 | 2008  
(1 row)  
  
grebenik_anzhelika=> update auto set year=2010 where a_id=1;  
UPDATE 1  
grebenik_anzhelika=> commit;  
COMMIT  
grebenik_anzhelika=>   
  
grebenik_anzhelika@vpsj3leQ:~  
grebenik_anzhelika=> update auto set year=2010 where a_id=1;  
UPDATE 1  
grebenik_anzhelika=> \q  
[grebenik_anzhelika@vpsj3leQ ~]$ psql  
psql (9.5.25)  
Type "help" for help.  
  
grebenik_anzhelika=> start transaction;  
START TRANSACTION  
grebenik_anzhelika=> set transaction isolation level read committed;  
SET  
grebenik_anzhelika=> select * from auto where a_id=1;  
a_id | name | year  
-----  
1 | BMW 5 | 2008  
(1 row)  
  
grebenik_anzhelika=> update auto set year=2011 where a_id=1;  
UPDATE 1  
grebenik_anzhelika=>   

```

Відбувається зависання під час операції та завершення після команди commit

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.

```
grebenik_anzhelika@vpsj3leQ:~
login as: grebenik_anzhelika
grebenik_anzhelika@91.219.60.189's password:
Last login: Wed May 5 05:38:35 2021 from 46.149.53.142
[grebenik_anzhelika@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

grebenik_anzhelika=> start transaction;
START TRANSACTION
grebenik_anzhelika=> set transaction isolation level repeatable read;
SET
grebenik_anzhelika=> select * from auto where a_id=1;
 a_id |      name      | year
-----+-----+-----
  1   | BMW 5          | 2008
(1 row)

grebenik_anzhelika=> update auto set year=2010 where a_id=1;
UPDATE 1
grebenik_anzhelika=>
```

```
grebenik_anzhelika@vpsj3leQ:~
login as: grebenik_anzhelika
grebenik_anzhelika@91.219.60.189's password:
Last login: Wed May 5 05:47:13 2021 from 46.149.53.142
[grebenik_anzhelika@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

grebenik_anzhelika=> start transaction;
START TRANSACTION
grebenik_anzhelika=> set transaction isolation level repeatable read;
SET
grebenik_anzhelika=> select * from auto where a_id=1;
 a_id |      name      | year
-----+-----+-----
  1   | BMW 5          | 2008
(1 row)

grebenik_anzhelika=> update auto set year=2011 where a_id=1;
```

Другий термінал буде виводити помилку, тому що на такому рівні ізоляції заборонено виконувати паралельно зміни одних даних.

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.

```
grebenik_anzhelika@vpsj3leQ:~
(1 row)

grebenik_anzhelika=> update auto set year=2010 where a_id=1;
UPDATE 1
grebenik_anzhelika=> \q
[grebenik_anzhelika@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

grebenik_anzhelika=> start transaction;
START TRANSACTION
grebenik_anzhelika=> set transaction isolation level serializable;
SET
grebenik_anzhelika=> select * from auto where a_id=1;
 a_id |      name      | year
-----+-----+-----
  1   | BMW 5          | 2008
(1 row)

grebenik_anzhelika=> update auto set year=2008 where a_id=1;
UPDATE 1
grebenik_anzhelika=> commit;
COMMIT
grebenik_anzhelika=>
```

```
grebenik_anzhelika@vpsj3leQ:~
(1 row)

grebenik_anzhelika=> update auto set year=2011 where a_id=1;
UPDATE 1
grebenik_anzhelika=> \q
[grebenik_anzhelika@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

grebenik_anzhelika=> start transaction;
START TRANSACTION
grebenik_anzhelika=> set transaction isolation level serializable;
SET
grebenik_anzhelika=> select * from auto where a_id=1;
 a_id |      name      | year
-----+-----+-----
  1   | BMW 5          | 2008
(1 row)

grebenik_anzhelika=> update auto set year=2010 where a_id=1;
ERROR:  could not serialize access due to concurrent update
grebenik_anzhelika=>
```

*Після зміни значення під час другої транзакції відбувається зависання, виходимо з цього стану командою **commit** в першому терміналі та виводить помилку*

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

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

```
grebenik_anzhelika@vpsj3leQ:~
grebenik_anzhelika=> start transaction;
START TRANSACTION
grebenik_anzhelika=> set transaction isolation level repeatable read;
SET
grebenik_anzhelika=> select * from auto where a_id=1;
 a_id |      name      | year
-----+-----+-----
  1   | BMW 5          | 2010
(1 row)

grebenik_anzhelika=> update auto set year=2011 where a_id=1;
UPDATE 1
grebenik_anzhelika=> \q
[grebenik_anzhelika@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

grebenik_anzhelika=> start transaction;
START TRANSACTION
grebenik_anzhelika=> update auto set year=2020 where a_id=1;
UPDATE 1
grebenik_anzhelika=> update auto set year=2020 where a_id=2;
UPDATE 1
grebenik_anzhelika=>
```

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

grebenik_anzhelika=> start transaction;
START TRANSACTION
grebenik_anzhelika=> set transaction isolation level repeatable read;
SET
grebenik_anzhelika=> \q
[grebenik_anzhelika@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

grebenik_anzhelika=> start transaction;
START TRANSACTION
grebenik_anzhelika=> update auto set year=2020 where a_id=2;
UPDATE 1
grebenik_anzhelika=> update auto set year=2020 where a_id=1;
ERROR:  deadlock detected
DETAIL:  Process 26177 waits for ShareLock on transaction 3297; blocked by process 26170.
Process 26170 waits for ShareLock on transaction 3298; blocked by process 26177.
HINT:  See server log for query details.
CONTEXT:  while updating tuple (0,26) in relation "auto"
grebenik_anzhelika=>
```


3.2 Виконайте дві модифіковані транзакції.

Проаналізуйте реакцію СКБД на операцію UPDATE 2-ї транзакції (яка виконується пізніше) та яка призвела до тупику.

```
grebenik_anzhelika@vpsj3leQ:~  
grebenik_anzhelika=> update auto set year=2020 where a_id=2;  
UPDATE 1  
grebenik_anzhelika=> \q  
[grebenik_anzhelika@vpsj3leQ ~]$ psql  
psql (9.5.25)  
Type "help" for help.  
  
grebenik_anzhelika=> start transaction;  
START TRANSACTION  
grebenik_anzhelika=> set transaction isolation level serializable;  
SET  
grebenik_anzhelika=> select * from auto;  
a_id | name | year  
-----+-----+-----  
2 | Audi | 2006  
3 | Cadillac | 2019  
1 | BMW 5 | 2010  
(3 rows)  
  
grebenik_anzhelika=> update auto set year=2020 where a_id=1;  
UPDATE 1  
grebenik_anzhelika=> commit;  
COMMIT  
grebenik_anzhelika=>   
  
grebenik_anzhelika@vpsj3leQ:~  
^C  
Process 26170 waits for ShareLock on transaction 3298; blocked by process 261  
HINT: See server log for query details.  
CONTEXT: while updating tuple (0,26) in relation "auto"  
grebenik_anzhelika=> \q  
[grebenik_anzhelika@vpsj3leQ ~]$ psql  
psql (9.5.25)  
Type "help" for help.  
  
grebenik_anzhelika=> start transaction;  
START TRANSACTION  
grebenik_anzhelika=> set transaction isolation level serializable;  
SET  
grebenik_anzhelika=> select * from auto;  
a_id | name | year  
-----+-----+-----  
2 | Audi | 2006  
3 | Cadillac | 2019  
1 | BMW 5 | 2010  
(3 rows)  
  
grebenik_anzhelika=> update auto set year=2021 where a_id=1;  
ERROR: could not serialize access due to concurrent update  
grebenik_anzhelika=>   
  
grebenik_anzhelika@vpsj3leQ:~  
login as: grebenik_anzhelika  
grebenik_anzhelika@91.219.60.189's password:  
Last login: Wed May 5 05:38:35 2021 from 46.149.53.142  
[grebenik_anzhelika@vpsj3leQ ~]$ psql  
psql (9.5.25)  
Type "help" for help.  
  
grebenik_anzhelika=> start transaction;  
START TRANSACTION  
grebenik_anzhelika=> set transaction isolation level repeatable read;  
SET  
grebenik_anzhelika=> select * from auto where a_id=1;  
a_id | name | year  
-----+-----+-----  
1 | BMW 5 | 2008  
(1 row)  
  
grebenik_anzhelika=> update auto set year=2010 where a_id=1;  
UPDATE 1  
grebenik_anzhelika=>   
  
grebenik_anzhelika@vpsj3leQ:~  
login as: grebenik_anzhelika  
grebenik_anzhelika@91.219.60.189's password:  
Last login: Wed May 5 05:47:13 2021 from 46.149.53.142  
[grebenik_anzhelika@vpsj3leQ ~]$ psql  
psql (9.5.25)  
Type "help" for help.  
  
grebenik_anzhelika=> start transaction;  
START TRANSACTION  
grebenik_anzhelika=> set transaction isolation level repeatable read;  
SET  
grebenik_anzhelika=> select * from auto where a_id=1;  
a_id | name | year  
-----+-----+-----  
1 | BMW 5 | 2008  
(1 row)  
  
grebenik_anzhelika=> update auto set year=2011 where a_id=1;  
grebenik_anzhelika=>   

```

Висновок: під час лабораторної роботи ми перевірили поведінку процесів-транзакцій в базах даних та засоби керування ними через механізм блокування з використанням сучасних систем керування базами даних.