

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 AI-202
Лобко Данііл Віталійович

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

.Перелік завдань:

2 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

2 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.

```
lobko_daniil@vpsj3leQ:~$ Type "help" for help.

lobko_daniil=> START TRANSACTION;
START TRANSACTION
lobko_daniil=> SELECT xmin,xmax,e_id FROM employer;
xmin | xmax | e_id
-----+-----+-----
2410 | 3261 | 2
3262 | 0 | 1
(2 rows)

lobko_daniil=> INSERT INTO employer VALUES (3,'Stepanov',501);
INSERT 0 1
lobko_daniil=> SELECT * FROM employer;
e_id | name | salary
-----+-----+-----
2 | NotIvanov | 400
1 | Ivanov | 2003441
3 | Stepanov | 501
(3 rows)

lobko_daniil=> COMMIT;
COMMIT
lobko_daniil=>

lobko_daniil@vpsj3leQ:~$ e_id | name | salary
-----+-----+-----
2 | NotIvanov | 400
1 | Ivanov | 2003441
3 | Stepanov | 501
(3 rows)

lobko_daniil=> SELECT * FROM employer;
e_id | name | salary
-----+-----+-----
2 | NotIvanov | 400
1 | Ivanov | 2003441
3 | Stepanov | 501
(3 rows)

lobko_daniil=> SELECT * FROM employer;
e_id | name | salary
-----+-----+-----
2 | NotIvanov | 400
1 | Ivanov | 2003441
3 | Stepanov | 502
(3 rows)

lobko_daniil=>

lobko_daniil@vpsj3leQ:~$ 2410 | 3261 | 2
3262 | 0 | 1
(2 rows)

lobko_daniil=> ROLLBACK;
lobko_daniil-> ROLLBACK;
ERROR: syntax error at or near "ROLLBACK"
LINE 2: ROLLBACK;
^

lobko_daniil=> \q
[lobko_daniil@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

lobko_daniil=> START TRANSACTION;
START TRANSACTION
lobko_daniil=> DELETE FROM employer WHERE e_id=3;
DELETE 1
lobko_daniil=> ROLLBACK;
ROLLBACK
lobko_daniil=> COMMIT;
WARNING: there is no transaction in progress
COMMIT
lobko_daniil=>
```

```
lobko_daniil@vpsj3leQ:~$ Type "help" for help.

lobko_daniil=> START TRANSACTION;
START TRANSACTION
lobko_daniil=> SELECT xmin,xmax,e_id FROM employer;
xmin | xmax | e_id
-----+-----+-----
2410 | 3261 | 2
3262 | 0 | 1
(2 rows)

lobko_daniil=> INSERT INTO employer VALUES (3,'Stepanov',501);
INSERT 0 1
lobko_daniil=> SELECT * FROM employer;
e_id | name | salary
-----+-----+-----
2 | NotIvanov | 400
1 | Ivanov | 2003441
3 | Stepanov | 501
(3 rows)

lobko_daniil=> COMMIT;
COMMIT
lobko_daniil=>

lobko_daniil@vpsj3leQ:~$ SELECT xmin,xmax,e_id FROM employer;
xmin | xmax | e_id
-----+-----+-----
2410 | 3261 | 2
3262 | 0 | 1
(2 rows)

lobko_daniil=> SELECT xmin,xmax,e_id FROM employer;
xmin | xmax | e_id
-----+-----+-----
2410 | 3261 | 2
3262 | 0 | 1
3263 | 0 | 3
(3 rows)

lobko_daniil=> SELECT * FROM employer;
e_id | name | salary
-----+-----+-----
2 | NotIvanov | 400
1 | Ivanov | 2003441

lobko_daniil@vpsj3leQ:~$ 2410 | 3261 | 2
3262 | 0 | 1
(2 rows)

lobko_daniil=> ROLLBACK;
lobko_daniil-> ROLLBACK;
ERROR: syntax error at or near "ROLLBACK"
LINE 2: ROLLBACK;
^

lobko_daniil=> \q
[lobko_daniil@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

lobko_daniil=> START TRANSACTION;
START TRANSACTION
lobko_daniil=> DELETE FROM employer WHERE e_id=3;
DELETE 1
lobko_daniil=> ROLLBACK;
ROLLBACK
lobko_daniil=> COMMIT;
WARNING: there is no transaction in progress
COMMIT
lobko_daniil=> UPDATE 1
lobko_daniil=> SELECT xmin,xmax,e_id FROM employer;
xmin | xmax | e_id
-----+-----+-----
3261 | 0 | 2
3262 | 0 | 1
(2 rows)

lobko_daniil=> ROLLBACK;
ROLLBACK
lobko_daniil=> \q
[lobko_daniil@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

lobko_daniil=> START TRANSACTION;
START TRANSACTION
lobko_daniil=> UPDATE employer SET salary=502 WHERE e_id=3;
UPDATE 1
lobko_daniil=> SELECT * FROM employer;
e_id | name | salary
-----+-----+-----
2 | NotIvanov | 400
1 | Ivanov | 2003441
```

Постійно дивлячись xmin та xmax ми можемо бачити номер транзакції, що додала або видалила рядок. Це корисно при роботі з декількома терміналами.

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.

IX-IS

```
lobko_danil@vpj3leQ~  
(lobko_danil@vpj3leQ ~)$ psql  
psql (9.5.25)  
Type "help" for help.  
  
lobko_danil=> START TRANSACTION;  
START TRANSACTION  
lobko_danil=> LOCK TABLE employer in row exclusive mode;  
LOCK TABLE  
lobko_danil=>
```

```
lobko_danil@vpj3leQ~  
(lobko_danil@vpj3leQ ~)$ psql  
psql (9.5.25)  
Type "help" for help.  
  
lobko_danil=> START TRANSACTION;  
START TRANSACTION  
lobko_danil=> LOCK TABLE employer in row share mode;  
LOCK TABLE  
lobko_danil=>
```

```
lobko_danil@vpj3leQ~  
-----  
relation | locktype | virtualtransaction | pid | mode | granted  
-----  
11673 | relation | 23/1941 | 4837 | AccessShareLock | t  
11673 | relation | 8/27158 | 30304 | AccessShareLock | t  
11673 | relation | 15/7259 | 2289 | AccessShareLock | t  
16639 | relation | 17/3324 | 30269 | AccessShareLock | t  
16639 | relation | 17/3324 | 30269 | RowExclusiveLock | t  
16714 | relation | 19/3379 | 4481 | ShareRowExclusiveLock | f  
16714 | relation | 18/7381 | 4534 | RowExclusiveLock | f  
16714 | relation | 15/7259 | 2289 | RowExclusiveLock | t  
16714 | relation | 3/86763 | 330 | AccessShareLock | t  
(9 rows)  
  
lobko_danil=> 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  
16714 | relation | 14/5281 | 5101 | RowExclusiveLock | t  
11673 | relation | 8/27158 | 30304 | AccessShareLock | t  
11673 | relation | 15/7259 | 2289 | AccessShareLock | t  
16639 | relation | 17/3324 | 30269 | AccessShareLock | t  
16639 | relation | 17/3324 | 30269 | RowExclusiveLock | t  
16714 | relation | 10/11251 | 5094 | RowShareLock | t  
16714 | relation | 15/7259 | 2289 | RowExclusiveLock | t  
16714 | relation | 3/86763 | 330 | AccessShareLock | t  
(9 rows)
```

SIX-IX

```
lobko_danil@vpj3leQ~  
(lobko_danil@vpj3leQ ~)$ psql  
psql (9.5.25)  
Type "help" for help.  
  
lobko_danil=> START TRANSACTION;  
START TRANSACTION  
lobko_danil=> LOCK TABLE employer in share row exclusive mode;  
^C  
ERROR: canceling statement due to user request  
lobko_danil=>
```

```
lobko_danil@vpj3leQ~  
(lobko_danil@vpj3leQ ~)$ psql  
psql (9.5.25)  
Type "help" for help.  
  
lobko_danil=> START TRANSACTION;  
START TRANSACTION  
lobko_danil=> LOCK TABLE employer in row exclusive mode;  
LOCK TABLE  
lobko_danil=> ^C  
lobko_danil=>
```

```
lobko_danil@vpj3leQ~  
Type "help" for help.  
  
lobko_danil=> START TRANSACTION;  
START TRANSACTION  
lobko_danil=> SELECT relation,locktype,virtualtransaction,pid,mode,granted FROM  
pg_locks WHERE locktype='relation';  
lobko_danil=> ^C  
lobko_danil=> 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 | 8/27158 | 30304 | AccessShareLock | t  
11673 | relation | 15/7259 | 2289 | AccessShareLock | t  
16639 | relation | 17/3324 | 30269 | AccessShareLock | t  
16639 | relation | 17/3324 | 30269 | RowExclusiveLock | t  
16714 | relation | 19/3379 | 4481 | ShareRowExclusiveLock | f  
16714 | relation | 18/7381 | 4534 | RowExclusiveLock | f  
16714 | relation | 15/7259 | 2289 | RowExclusiveLock | t  
16714 | relation | 3/86763 | 330 | AccessShareLock | t  
(9 rows)  
  
lobko_danil=>
```

SIX-IS

```

[lobko_daniil@vps33ieQ ~]$ psql
psql (9.5.25)
Type "help" for help.

lobko_daniil=> START TRANSACTION;
START TRANSACTION
lobko_daniil=> LOCK TABLE employer IN share row exclusive mode;
[

lobko_daniil@vps33ieQ ~]$ psql
psql (9.5.25)
Type "help" for help.

lobko_daniil=> START TRANSACTION;
START TRANSACTION
lobko_daniil=> LOCK TABLE employer in row share mode;
lobko_daniil=> [

lobko_daniil@vps33ieQ ~]$ psql
psql (9.5.25)
Type "help" for help.

lobko_daniil=> SELECT relation,locktype,virtualtransaction,pid,mode,granted FROM
pg_locks WHERE 1
locktype='relation';
relation | locktype | virtualtransaction | pid | mode | granted
-----
11673 | relation | 23/1941 | 4837 | AccessShareLock | t
16714 | relation | 14/5281 | 5101 | RowExclusiveLock | t
11673 | relation | 8/27158 | 30304 | AccessShareLock | t
11673 | relation | 15/7259 | 2289 | AccessShareLock | t
16639 | relation | 17/3324 | 30269 | AccessShareLock | t
16639 | relation | 17/3324 | 30269 | RowExclusiveLock | t
16714 | relation | 10/11251 | 5094 | RowShareLock | t
16714 | relation | 15/7259 | 2289 | RowExclusiveLock | t
16714 | relation | 3/86763 | 330 | AccessShareLock | t
(9 rows)

```

Таблиця 3 - Матриця сумісності гранульованих блокувань.

| | <i>X</i> | <i>SIX</i> | <i>IX</i> | <i>S</i> | <i>IS</i> |
|------------|----------|------------|-----------|----------|-----------|
| <i>X</i> | - | - | - | - | - |
| <i>SIX</i> | - | + | + | - | + |
| <i>IX</i> | - | - | + | - | + |
| <i>S</i> | - | - | - | + | + |
| <i>IS</i> | - | + | + | + | + |

Згідно з матрицею сумісності, ми бачимо, які блокування несумісні. Шляхом практики ми в цьому переконалися.

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

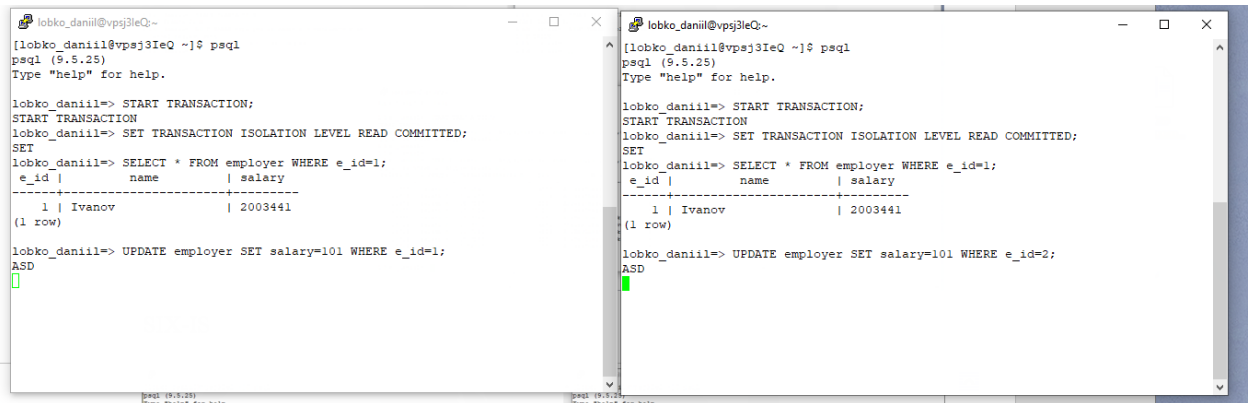
Підготуйте транзакції, які було створено у завданні 3.1 рішення попередньої лабораторної роботи, а саме, створіть дві транзакції, кожна з яких повинна включати такі

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.



```
lobko_daniil@vpsj3IeQ:~$ psql
psql (9.5.25)
Type "help" for help.

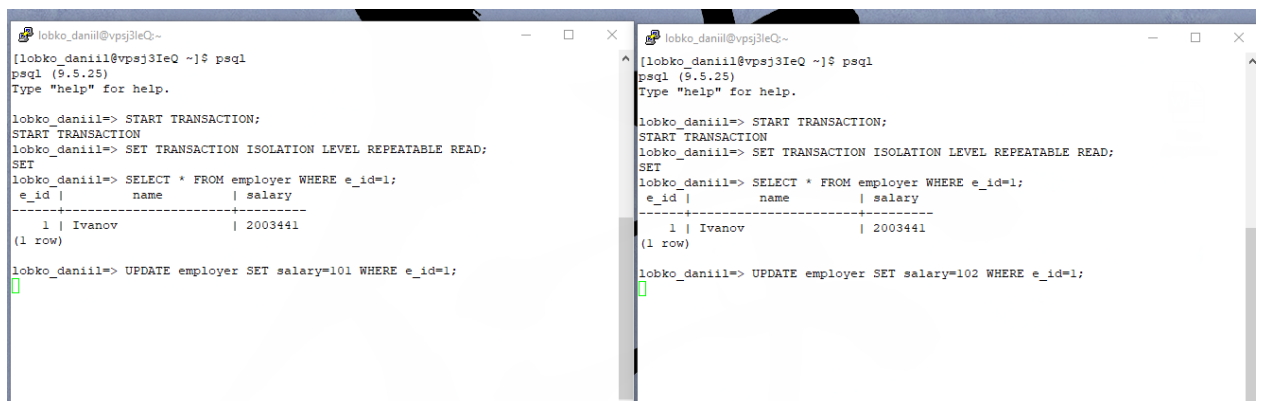
lobko_daniil=> START TRANSACTION;
START TRANSACTION
lobko_daniil=> SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
SET
lobko_daniil=> SELECT * FROM employer WHERE e_id=1;
 e_id |      name      | salary
-----+-----+-----
  1   | Ivanov         | 2003441
(1 row)

lobko_daniil=> UPDATE employer SET salary=101 WHERE e_id=1;
ASD
```

Як ми бачимо, після апдейту першої транзакції вона потрапила у сон.

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.



```
lobko_daniil@vpsj3IeQ:~$ psql
psql (9.5.25)
Type "help" for help.

lobko_daniil=> START TRANSACTION;
START TRANSACTION
lobko_daniil=> SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SET
lobko_daniil=> SELECT * FROM employer WHERE e_id=1;
 e_id |      name      | salary
-----+-----+-----
  1   | Ivanov         | 2003441
(1 row)

lobko_daniil=> UPDATE employer SET salary=101 WHERE e_id=1;
ASD
```

Транзакції знов чекають один одного.

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.


```
lobko_daniil@vpsj3leQ:~
[lobko_daniil@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

lobko_daniil=> START TRANSACTION
lobko_daniil-> ^C
lobko_daniil=> START TRANSACTION;
START TRANSACTION
lobko_daniil=> SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
SET
lobko_daniil=> SELECT * FROM employer WHERE e_id=1;
 e_id |      name      | salary
-----+-----+-----
  1   | Ivanov         | 2003441
(1 row)

lobko_daniil=> UPDATE employer SET salary=101 WHERE e_id=1;
```

```
lobko_daniil@vpsj3leQ:~
[lobko_daniil@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

lobko_daniil=> START TRANSACTION;
START TRANSACTION
lobko_daniil=> SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
SET
lobko_daniil=> SELECT * FROM employer WHERE e_id=1;
 e_id |      name      | salary
-----+-----+-----
  1   | Ivanov         | 2003441
(1 row)

lobko_daniil=> UPDATE employer SET salary=102 WHERE e_id=1;
```

Знов очікування.

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

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

```
lobko_daniil@vpsj3leQ:~
login slobko_daniil
lobko_daniil@91.219.60.189's password:
last login: Wed May 5 04:03:06 2021 from 78.26.152.178
[lobko_daniil@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

lobko_daniil=> START TRANSACTION;
START TRANSACTION
lobko_daniil=> SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SET
lobko_daniil=> SELECT * FROM employer WHERE e_id=1;
 e_id |      name      | salary
-----+-----+-----
  1   | Ivanov         | 2003441
(1 row)

lobko_daniil=> UPDATE employer SET salary=1 WHERE e_id=1;
```

```
lobko_daniil@vpsj3leQ:~
[lobko_daniil@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

lobko_daniil=> START TRANSACTION;
START TRANSACTION
lobko_daniil=> SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SET
lobko_daniil=> SELECT * FROM employer WHERE e_id=1;
 e_id |      name      | salary
-----+-----+-----
  1   | Ivanov         | 2003441
(1 row)

lobko_daniil=> UPDATE employer SET salary=2 WHERE e_id=1;
```

```
lobko_daniil@vpsj3leQ:~
psql (9.5.25)
Type "help" for help.

lobko_daniil=> START TRANSACTION;
START TRANSACTION
lobko_daniil=> SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
SET
lobko_daniil=> ^C
lobko_daniil=> ^C
lobko_daniil=> SELECT * FROM employer;
 e_id |      name      | salary
-----+-----+-----
  2   | NotIvanov       |    400
  1   | Ivanov          | 2003441
  3   | Stepanov        |    502
(3 rows)

lobko_daniil=> UPDATE employer SET salary=1 WHERE e_id=1;
COMMIT;
^CCancel request sent
ERROR: canceling statement due to user request
lobko_daniil=> COMMIT;
ROLLBACK
lobko_daniil=>
```

```
lobko_daniil@vpsj3leQ:~
[lobko_daniil@vpsj3leQ ~]$ psql
psql (9.5.25)
Type "help" for help.

lobko_daniil=> START TRANSACTION;
START TRANSACTION
lobko_daniil=>
lobko_daniil=>
lobko_daniil=> SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
SET
lobko_daniil=> SELECT * FROM employer;
 e_id |      name      | salary
-----+-----+-----
  2   | NotIvanov       |    400
  1   | Ivanov          | 2003441
  3   | Stepanov        |    502
(3 rows)

lobko_daniil=> UPDATE employer SET salary=2 WHERE e_id=2;
```

Як ми бачимо, вони призводять до тупика, (було застосовано блокування REPEATABLE READ и SERIALIZABLE);

4.2 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.

Висновки: Виконуючи цю лабораторну роботи ми закріпили навички роботи з керування процесами-транзакціями в базах даних

Найскладнішим було завдання 1 через незручне використання одразу 4 терміналів.