

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико–математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

Отчет по лабораторной работе № 3

Дисциплина: Системы управления базами данных

Студент: Логинов Сергей Андреевич

Группа: НФИбд-01-18

МОСКВА 2021г

Задание

Блокировка транзакций

1. Начните одну транзакцию и в ней измените определенную запись в таблице, затем приостановите выполнение на 20-30 секунд. В другой сессии попробуйте вначале сделать *select*, а затем *update* изменяемой записи, покажите в каком случае произойдет блокировка (или нет)
2. Начните одну транзакцию и в ней измените определенную запись в таблице, затем приостановите выполнение на 20-30 секунд. В другой сессии попробуйте вначале сделать *select*, а затем *update* другой записи, покажите в каком случае произойдет блокировка (или нет)

Выполнение

Для старта транзакции будем использовать *begin transaction*, для завершения транзакции и внесения изменения используем *commit*.

По сути, транзакция помогает решать те ситуации, когда бд находится в таком состоянии, что часть изменений в данных уже выполнена, но осталась еще, скажем, необработанная часть. Это может приводить к ошибкам и проблемам. Также транзакции предотвращают различные сбои, поскольку все изменения станут постоянными только после завершения транзакции. Предполагаю, что при использовании транзакции должно быть заблокировано любое параллельное изменение данных в таблице.

1. Начните одну транзакцию и в ней измените определенную запись в таблице, затем приостановите выполнение на 20-30 секунд. В другой сессии попробуйте вначале сделать *select*, а затем *update* изменяемой записи, покажите в каком случае произойдет блокировка (или нет)

Создадим базу данных для третьей лабораторной работы и таблицу *squares* в ней:

PostgreSQL 14

Databases (4)

lab01

lab02

lab03

Casts

Catalogs

Event Triggers

Extensions

Foreign Data Wrappers

Languages

Publications

Schemas (1)

public

Collations

Domains

2

3 -- DROP TABLE IF EXISTS public.squares;

4

5 CREATE TABLE IF NOT EXISTS public.squares

6 (

7 "number" integer NOT NULL,

8 square integer NOT NULL,

9 test character varying(100) COLLATE pg_catalog."default",

10 CONSTRAINT squares_pkey PRIMARY KEY ("number")

11)

12

13 TABLESPACE pg_default;

14

15 ALTER TABLE IF EXISTS public.squares

16 OWNER to postgres;

lab03/postgres@PostgreSQL 14

Query Editor

Query History

1

select * from squares

Data Output

Explain

Notifications

Messages

	number [PK] integer	square integer
1	1	1
2	2	4
3	3	9

Для начала попробуем не завершая транзакцию изменить используемое в транзакции значение в параллельной сессии:

1 сессия

lab03/postgres@PostgreSQL 14

Query EditorQuery History

1

begin transaction;

2

update squares set square = 0 where number = 2;

3

Data OutputExplainNotificationsMessages

	number [PK] integer	square integer	
1	2	0	

2 сессия

lab03/postgres@PostgreSQL 14

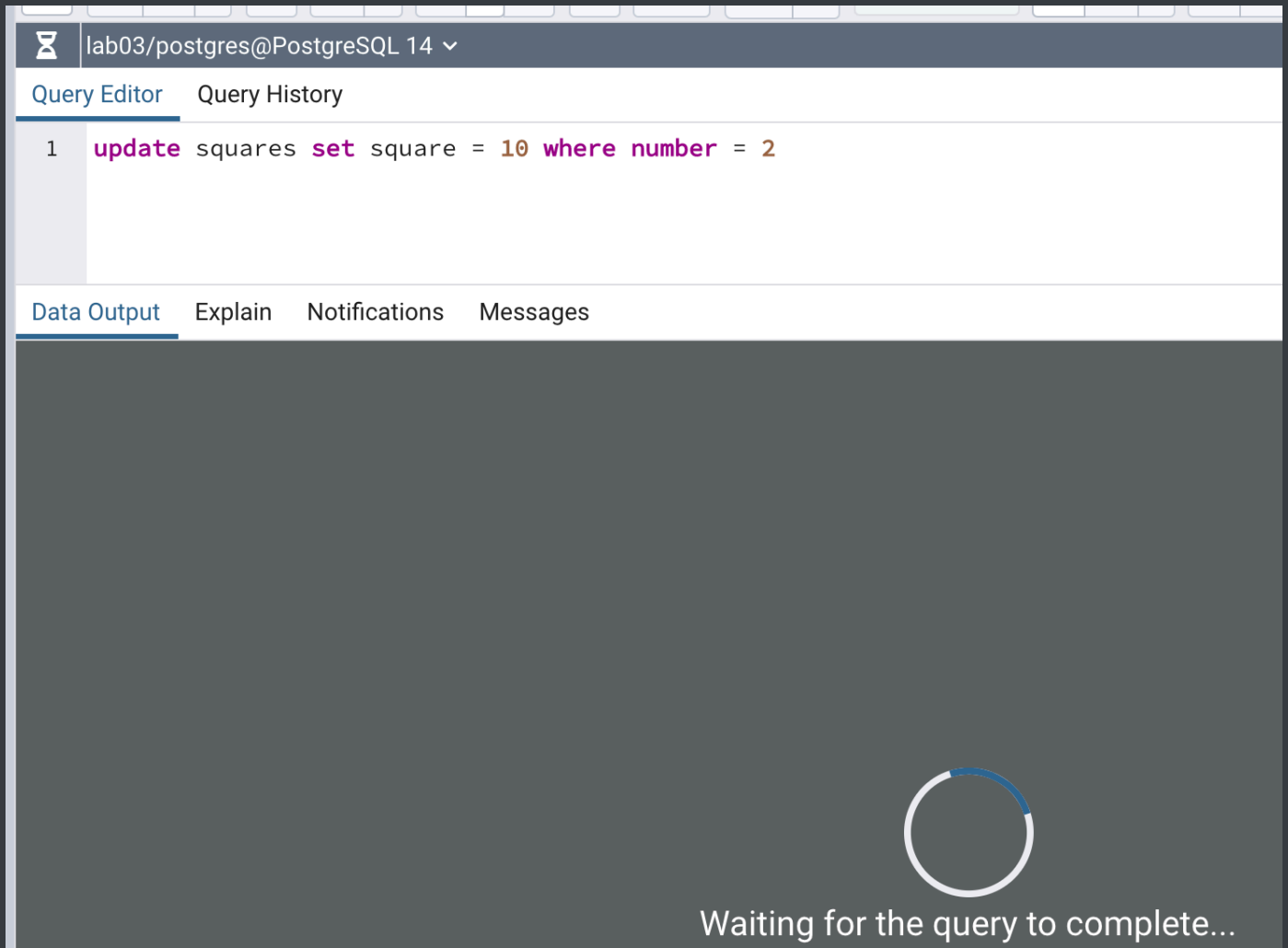
Query EditorQuery History

1

select * from squares where number = 2

Data OutputExplainNotificationsMessages

	number [PK] integer	square integer	
1	2	4	



Вывод: транзакция не препятствует чтению значения, но она заблокировала обновление значения в параллельной сессии, обновление состоится после команды commit или rollback в первой сессии

То же самое с завершением транзакции и командой pg_sleep:

1 сессия


lab03/postgres@PostgreSQL 14

Query EditorQuery History

```
1 begin transaction;
2 update squares set square = 0 where number = 2;
3 select pg_sleep(25);
4 commit;
5 select * from squares where number = 2
6
```

Data OutputExplainNotificationsMessages

	number [PK] integer	square integer
1	1	1
2	2	4
3	3	9



Waiting for the query to complete...

2 сессия

lab03/postgres@PostgreSQL 14

Query EditorQuery History

```
1 select * from squares where number = 2;
```

Data OutputExplainNotificationsMessages

	number [PK] integer	square integer
1	2	4


lab03/postgres@PostgreSQL 14

Query EditorQuery History

1 update squares set square = 100 where number = 2

Data OutputExplainNotificationsMessages

	number [PK] integer	square integer
1	2	4



Waiting for the query to complete...

Результаты аналогичные, значение можно прочесть и нельзя перезаписать.

- Начните одну транзакцию и в ней измените определенную запись в таблице, затем приостановите выполнение на 20-30 секунд. В другой сессии попробуйте вначале сделать *select*, а затем *update* другой записи, покажите в каком случае произойдет блокировка (или нет)

Сначала так же без завершения транзакции:

1 сессия

lab03/postgres@PostgreSQL 14

Query EditorQuery History

1begin transaction;

2update squares set square = 0 where number = 2

3

Data OutputExplainNotificationsMessages

	number [PK] integer	square integer	
1	1	1	
2	2	4	
3	3	9	

2 сессия, чтение

lab03/postgres@PostgreSQL 14

Query EditorQuery History

1select * from squares where number = 3

2

Data OutputExplainNotificationsMessages

	number [PK] integer	square integer	
1	3	9	

запись

lab03/postgres@PostgreSQL 14 ▾

Query Editor Query History

1 **update** squares **set** square = **100** **where** number = **3**;

2 **select** * **from** squares **where** number = **3**

Data Output Explain Notifications Messages

	number [PK] integer	square integer	
1	3	100	

Даже с незавершенной транзакцией нам удалось прочитать и изменить другое значение таблицы. В данном случае блокируются только изменения значения из транзакции.

С завершением транзакции и pg_sleep:

1 сессия

```
1 begin transaction;
2 update squares set square = 0 where number = 2;
3 select pg_sleep(30);
4 commit;
```

	<div>number</div> <div>[PK] integer</div>	<div>square</div> <div>integer</div>
1	1	1
2	2	4
3	3	9



Waiting for the query to complete...

lab03/postgres@PostgreSQL 14 ▾

Query Editor Query History

```
1 select * from squares where number = 3
```

Data Output Explain Notifications Messages

	number [PK] integer	square integer
1	3	9

запись

lab03/postgres@PostgreSQL 14 ▾

Query Editor Query History

```
1 update squares set square = 100 where number = 3;  
2 select * from squares where number = 3
```

Data Output Explain Notifications Messages

	number [PK] integer	square integer
1	3	100

В данном случае все аналогично, транзакция не препятствует изменению других значений

Вывод: в данной лабораторной работе на практике изучили принцип работы транзакций и их блокировки. Транзакция выступает как атомарная операция, которая либо будет завершена полностью, либо не будет выполнена вовсе. При этом данные обновляются только после успешного завершения (коммита). Также транзакция блокирует любые параллельные изменения значений, которые используются в транзакции, но не блокирует чтение этих значений.