Федеральное государственное автономное

образовательное учреждение

Высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

|  |
| --- |
| Институт космических и информационных технологий |
| институт |
| Информатика |
| кафедра |

**ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ**

|  |
| --- |
| Транзакции |
| тема |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Преподаватель | |  |  |  | Е. П. Моргунов |
|  | |  | подпись, дата |  | инициалы, фамилия |
| Студент | КИ21-17/2Б, 032156940 |  |  |  | Н. А. Самарин |
|  | номер группы, зачётной книжки |  | подпись, дата |  | инициалы, фамилия |

Красноярск 2023

**СОДЕРЖАНИЕ**

1 Задание............................................................................................................... 3

2 Ход работы........................................................................................................ 3

**1 Задание**

- Изучить материал главы 9 учебного пособия: Моргунов, Е. П.  
PostgreSQL. Основы языка SQL [Текст] : учеб. пособие / Е. П. Моргунов ; под  
ред. Е. В. Рогова, П. В. Лузанова. – СПб. : БХВ-Петербург, 2018. – 336 с. : ил. –  
ISBN 978-5-9775-4022-3. Можно дополнительно использовать презентации и  
видеоуроки: Моргунов, Е. П. Язык SQL [Электронный ресурс] : практический  
курс. – М. : Постгрес Профессиональный, 2019;

- Выполнить задания, приведенные в разделе "Контрольные вопросы и  
задания";

- Подготовить отчет. Включить в него тексты заданий из пособия. Для  
каждого задания сделать снимки экрана с введенными командами языка SQL и  
результатами их выполнения в среде утилиты psql.

**2 Ход работы**

1. По умолчанию каждая SQL-команда, выполняемая в среде psql,  
образует отдельную транзакцию с уровнем изоляции Read Committed. Поэтому  
в тех экспериментах, когда одна из транзакций состоит только из единственной  
SQLкоманды, можно не выполнять команды BEGIN и END. Конечно, если  
каждая из параллельных транзакций состоит из единственной SQL-команды, то  
хотя бы для одной из транзакций придется все же выполнить и команду BEGIN,  
иначе эксперимент не получится. В тексте главы были приведены примеры  
транзакций, в которых рассматривались команды SELECT ... FOR UPDATE и  
LOCK TABLE. Попробуйте повторить эти эксперименты с учетом описанного  
поведения PostgreSQL.

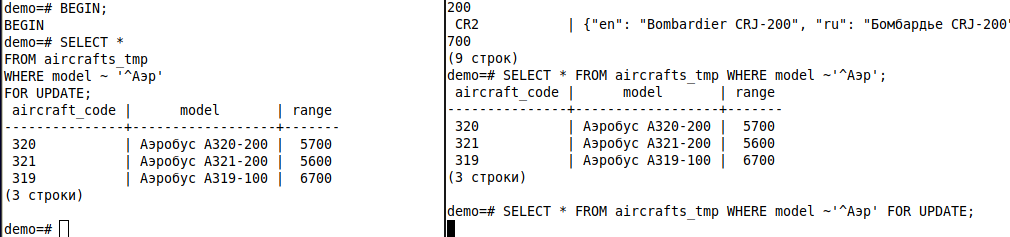


Рисунок 1 – Выполнение задания 1 часть 1

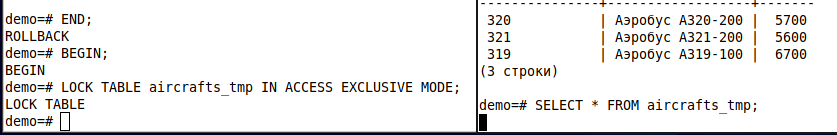


Рисунок 2 – Выполнение задания 1 часть 2

3.\* Предположим, что в одной транзакции будет просто присваиваться новое значение, например, так:UPDATE aircrafts\_tmp SET range = 2100 WHERE aircraft\_code = 'CR2'; А в параллельной транзакции будет выполняться аналогичная команда: UPDATE aircrafts\_tmp SET range = 2500 WHERE aircraft\_code = 'CR2'; Очевидно, что сохранится только одно из значений атрибута range. Можно ли говорить, что в такой ситуации имеет место потерянное обновление? Если оно имеет место, то что можно предпринять для его недопущения? Обоснуйте ваш ответ.

ОТВЕТ: С пользовательской стороны, произошедшее выглядит как  
потерянное обновление. Хотя на самом деле то, что происходит, эквивалентно  
последовательным транзакциям. Чтобы избежать потерь, можно использовать  
более высокий уровень изоляции.

5. В тексте главы была рассмотрена команда SELECT ... FOR UPDATE,  
выполняющая блокировку на уровне отдельных строк. Организуйте две  
параллельные транзакции с уровнем изоляции Read Committed и выполните с  
ними ряд экспериментов. В первой транзакции заблокируйте некоторое  
множество строк, отбираемых с помощью условия WHERE. А во второй  
транзакции изменяйте условие выборки таким образом, чтобы выбираемое  
множество строк: – являлось подмножеством множества строк, выбираемых в  
первой транзакции; – являлось надмножеством множества строк, выбираемых в  
первой транзакции; – пересекалось с множеством строк, выбираемых в первой  
транзакции; – не пересекалось с множеством строк, выбираемых в первой  
транзакции.

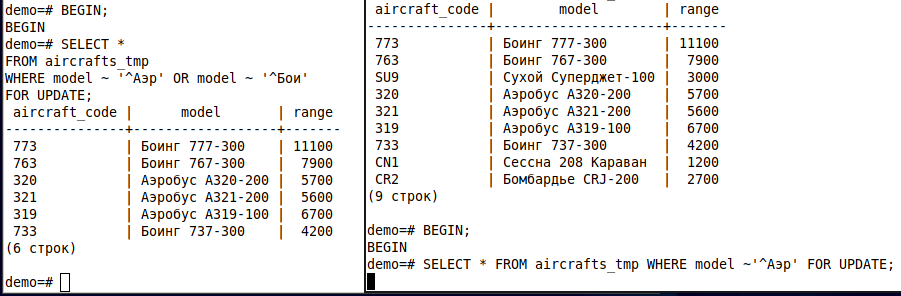


Рисунок 3 – Выполнение задания 5 часть 1

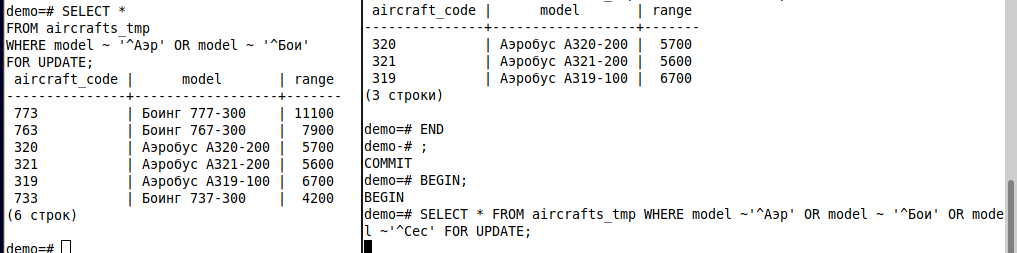


Рисунок 4 – Выполнение задания 5 часть 2

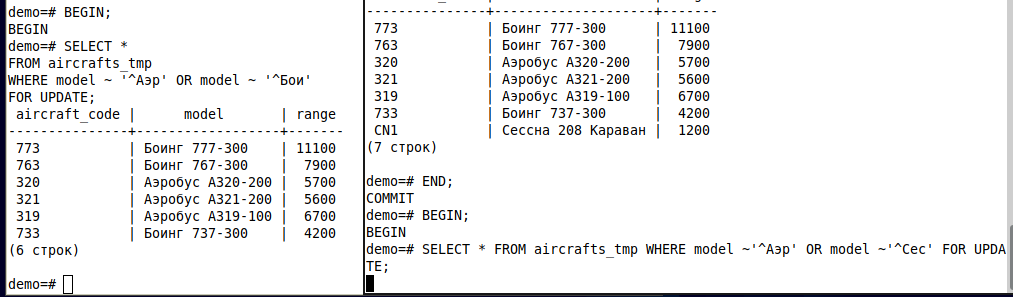


Рисунок 5 – Выполнение задания 5 часть 3

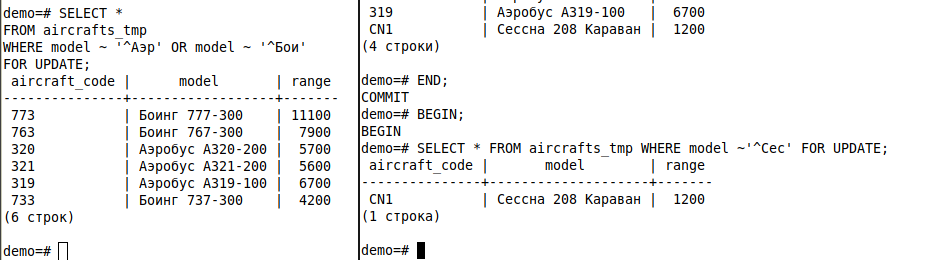


Рисунок 6 – Выполнение задания 5 часть 4

7. В тексте главы для иллюстрации изучаемых концепций мы создавали  
только две параллельные транзакции. Попробуйте воспроизвести  
представленные эксперименты, создав три или даже четыре параллельные  
транзакции.

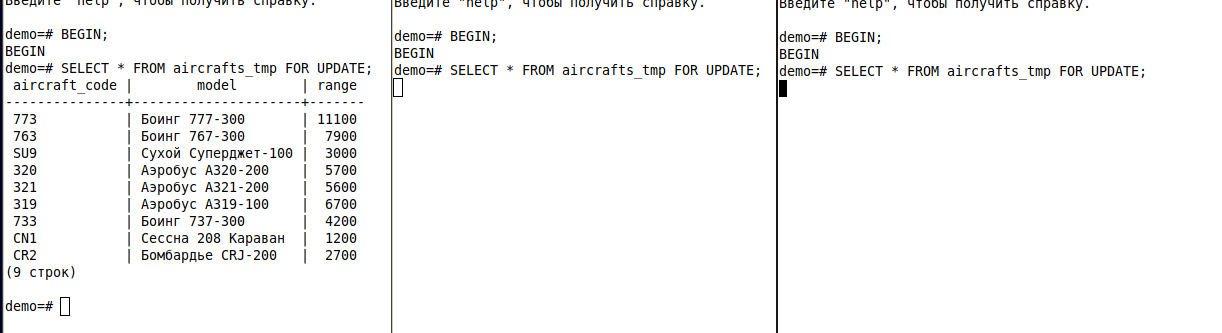


Рисунок 7 – Выполнение задания 7

9.\* В разделе документации 13.2.3 «Уровень изоляции Serializable»  
сказано, что если поиск в таблице осуществляется последовательно, без  
использования индекса, тогда на всю таблицу накладывается так называемая  
предикатная блокировка. Такой подход приводит к увеличению числа сбоев  
сериализации. В качестве контрмеры можно попытаться использовать индексы.  
Конечно, если таблица совсем небольшая, то может и не получиться заставить  
PostgreSQL использовать поиск по индексу. Тем не менее давайте выполним  
следующий эксперимент... ...Теперь система смогла сериализовать  
параллельные транзакции и зафиксировать их обе. Как вы думаете, почему это  
удалось?

ОТВЕТ: При использовании индекса, при поиске в таблице запрос  
обращается не напрямую к значениям строк, а к значениям сгенерированным  
индексом. Следовательно, предикатная блокировка не накладывается и сбоев  
сериализации не происходит.