|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Инструментального и прикладного программного обеспечения

**ПРАКТИЧЕСКАЯ РАБОТА №2**

по дисциплине «Технологии обработки транзакций клиент-серверных приложений»

**Тема практической работы:**

**Транзакции. Уровни изоляции. Уровни изоляции в PostgreSQL Read Committed, Repeatable Read. Точки сохранения. ЭТУ РАБОТУ УКРАДЁТ ТОГО НОЧЬЮ СОЖРЁТ КРОКОДИЛ**

**Студент группы** ИКБО-20-19 Анваржонов Ж Т

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись студента)

**Руководитель практической работы** Белолипцев М.А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя)

Работа представлена «01» апреля 2022 г.

Работа зачтена «\_\_» \_\_\_\_\_\_\_ 2022 г.

Москва 2022

**Задание на практическую работу:**

1. Начните транзакцию (командой BEGIN) и создайте новый заказ в таблице sales\_order c сегодняшней датой. Добавьте два предмета в таблицу item, связанных с созданным заказом.

Представьте, что пользователь не подтвердил заказ и все введенные данные необходимо отменить. Выполните отмену транзакции и проверьте, что никакой добавленной вами информации действительно не осталось.

2. Теперь представьте сценарий, в котором нужно отменить не все данные, а только последний из добавленных предметов. Для этого повторите все действия из предыдущего упражнения, но перед добавлением каждого предмета создавайте точку сохранения (с одним и тем же именем). После ввода второго предмета выполните откат к точке сохранения. Проверьте, что заказ и первый предмет остались.

3. В рамках той же транзакции добавьте еще один предмет и зафиксируйте транзакцию. Обратите внимание на то, что после этой операции отменить внесенные транзакцией изменения будет уже невозможно.

**Уровень изоляции Read Committed**

4. Перед началом выполнения задания проверьте, что в таблице sales\_order нет заказов на сумму total 1 000 рублей.

4.1. В первом сеансе начните транзакцию (командой BEGIN). Выполните обновление таблицы sales\_order: увеличьте total в два раза в тех строках, где сумма равна 1 000 рублей.

4.2. Во втором сеансе (откройте новое окно psql) вставьте в таблицу sales\_order новый заказ на 1 000 рублей и зафиксируйте транзакцию.

4.3. В первом сеансе повторите обновление таблицы sales\_order и зафиксируйте транзакцию.

Осталась ли сумма добавленного заказа равной 1 000 рублей? Почему это не так?

**Уровень изоляции Repeatable Read**

5. Повторите предыдущее упражнение, но начните транзакцию в первом сеансе с уровнем изоляции транзакций Repeatable Read. Объясните различие полученных результатов.

6. Выполните указанные действия в двух сеансах:

6.1. В первом сеансе начните новую транзакцию с уровнем изоляции Repeatable Read. Вычислите количество заказов с суммой 20 000 рублей.

6.2. Во втором сеансе начните новую транзакцию с уровнем изоляции Repeatable Read. Вычислите количество заказов с суммой 30 000 рублей.

6.3. В первом сеансе добавьте новый заказ на 30 000 рублей и снова вычислите количество заказов с суммой 20 000 рублей.

6.4. Во втором сеансе добавьте новый заказ на 20 000 рублей и снова вычислите количество заказов с суммой 30 000 рублей.

6.5. Зафиксируйте транзакции в обоих сеансах.

Соответствует ли результат ожиданиями? Можно ли сериализовать эти транзакции (иными словами, можно ли представить такой порядок последовательного выполнения этих транзакций, при котором результат совпадет с тем, что получился при параллельном выполнении)?

**Выполнение работы:**

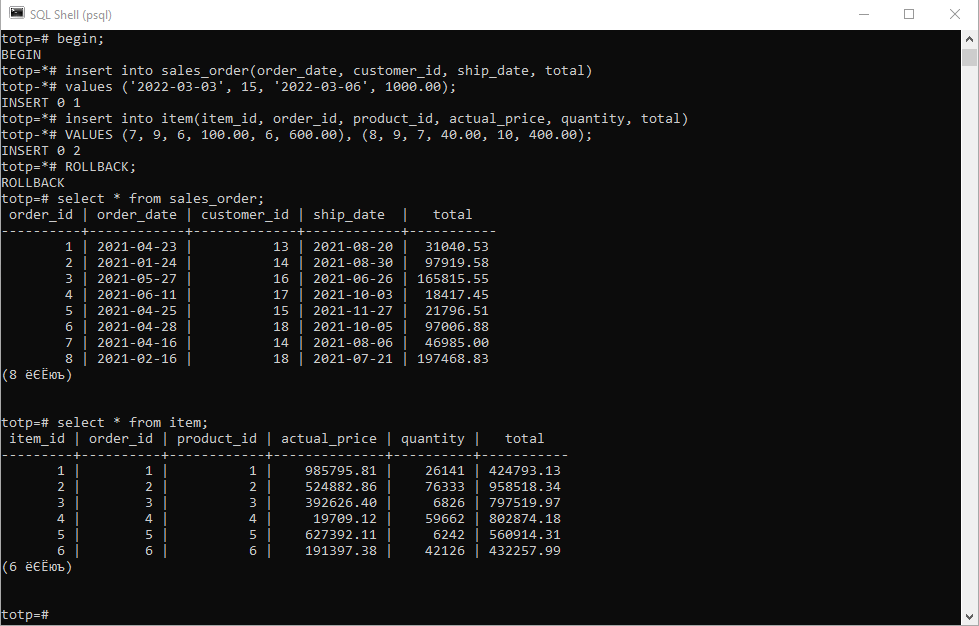


Рис. 1 ‒ Создание и отмена транзакции

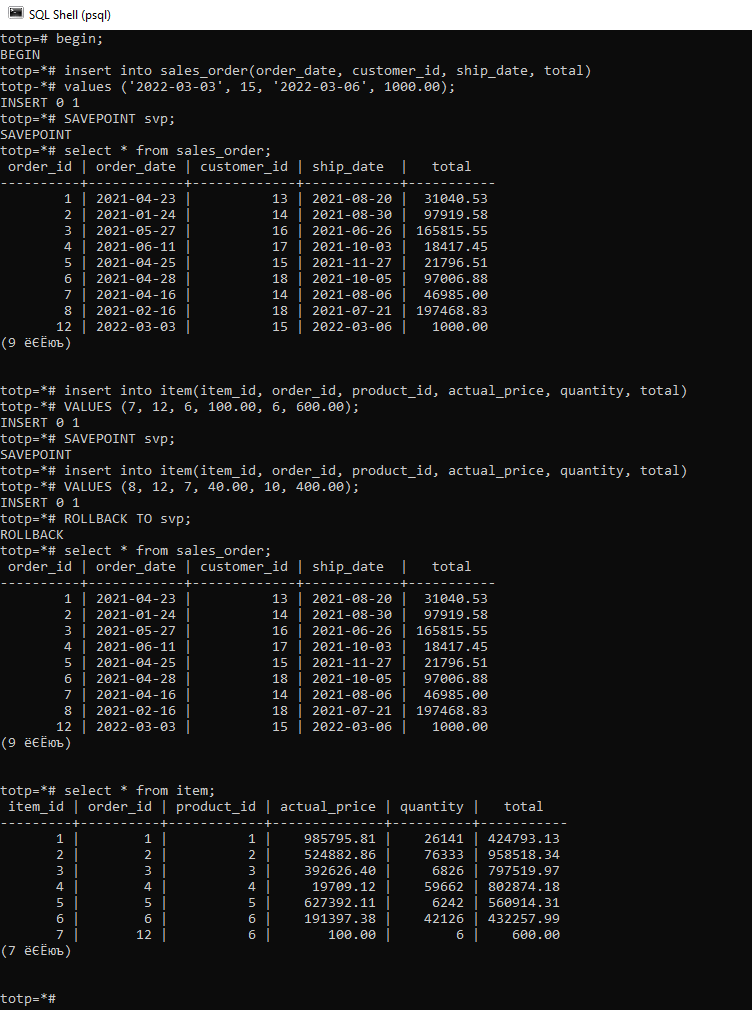


Рис. 2 ‒ Работа с точками сохранения в транзакции

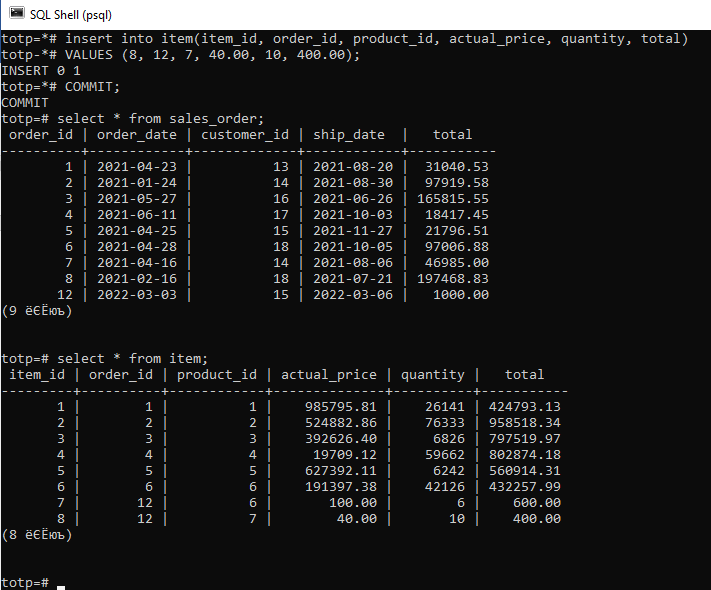


Рис. 3 ‒ Добавление третьего предмета и завершение транзакции

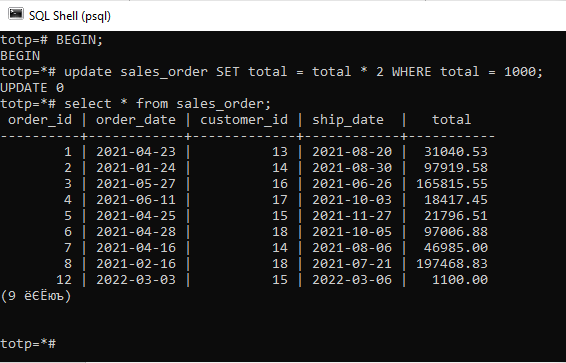


Рис. 4 ‒ Начало работы с первым сеансом транзакции

без заказов с total =1000(незавершённая)

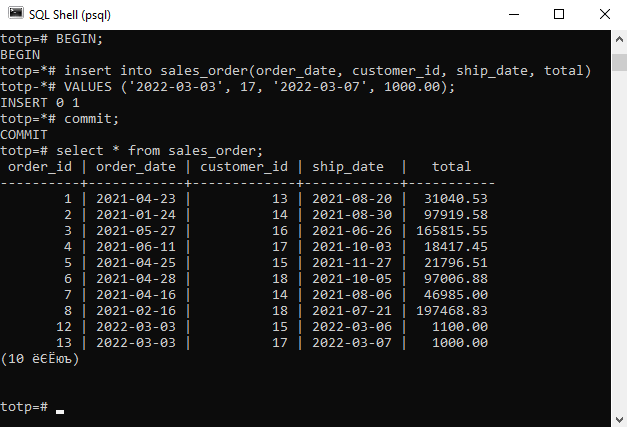


Рис. 5 ‒ Сеанс со второй транзакцией, где происходит

добавление заказа с total = 1000(завершённая)

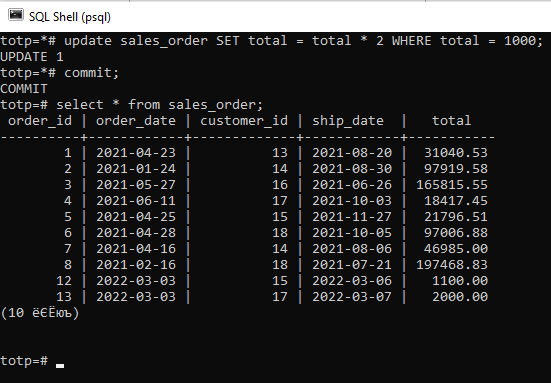


Рис. 6 ‒ Обновление значения у транзакций с total = 1000(завершенная)

Осталась ли сумма добавленного заказа равной 1 000 рублей? Почему это не так?

Нет, не осталась. Это связано с аномалией фантомного чтения возникшей из-за работы второй транзакции.

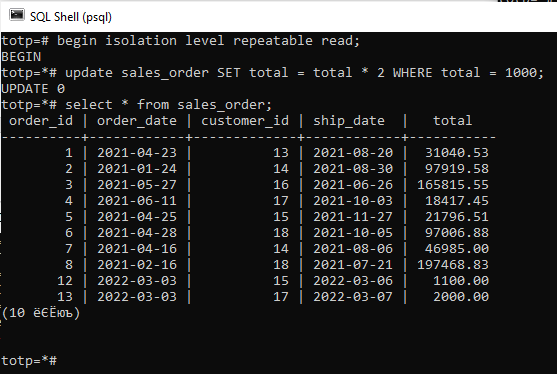


Рис. 7 ‒ Начало работы с первым сеансом транзакции на уровне изоляции Repeatable Read без заказов с total =1000(незавершённая)

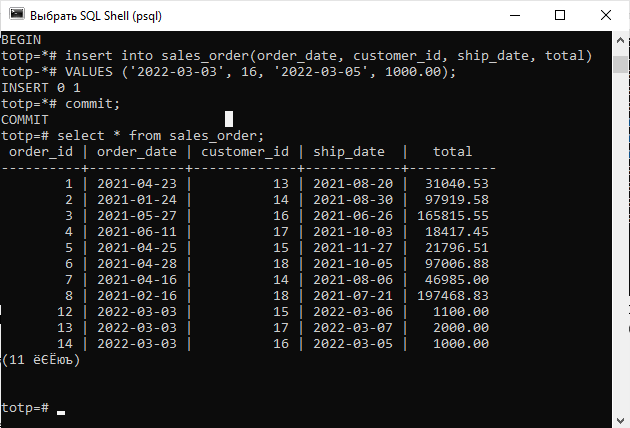


Рис. 8 ‒ Сеанс со второй транзакцией, где происходит

добавление заказа с total = 1000(завершённая)

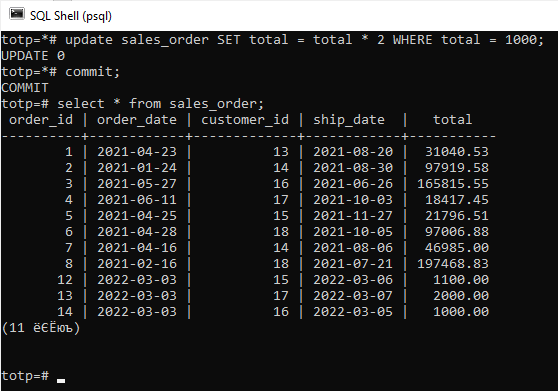


Рис. 9 ‒ Попытка обновление значения у транзакций с

total = 1000(завершенная)

Уровень Repeatable Read не допускает не только неповторяющегося, но и фантомного чтения (хотя и не обеспечивает полную изоляцию). В связи с этой особенностью значение не изменилось после добавления записи с total = 1000 в транзакции второго сеанса

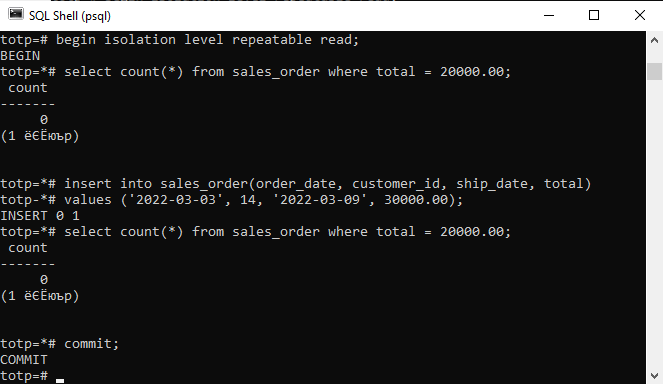


Рис. 10 ‒ Окончательный результат выполнения транзакции в первом сеансе

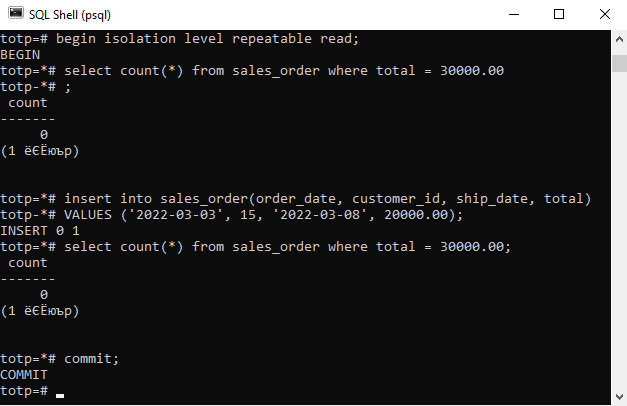


Рис. 11 ‒ Окончательный результат выполнения транзакции во втором сеансе

Результат выполнения транзакций не соответствует ожиданиям ‒ не произошло увеличение количества заказов, удовлетворяющих условиям. Добиться результата аналогичного параллельной обработке можно только при подтверждении одной из транзакций перед повторной проверкой.

**Вывод:**

В результате выполнения данной практической работы были получены начальные навыки работы с транзакциями в СУБД PostgreSQL, получено практическое понимание о различиях между разными уровнями изоляции и опыт работы с сэйв-поинтами.