Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчёт о лабораторной работе №7**

**Дисциплина**: Базы данных

**Тема**: Изучение работы транзакций

Выполнил студент гр. 43501/1 Балясников А.Е.

(подпись)

Руководитель Мяснов А.В.

(подпись)

“ ” 2015 г.

Санкт-Петербург

2015

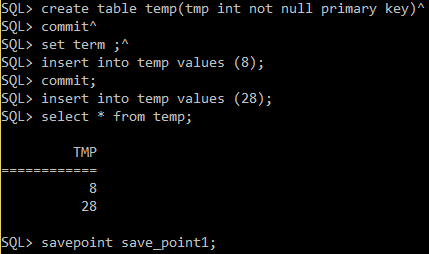
1. **Цель работы**

Познакомить студентов с механизмом транзакций, возможностями ручного управления транзакциями, уровнями изоляции транзакций.

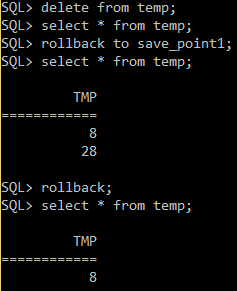
1. **Программа работы**
2. Изучить основные принципы работы транзакций.
3. Провести эксперименты по запуску, подтверждению и откату транзакций.
4. Разобраться с уровнями изоляции транзакций в Firebird.
5. Спланировать и провести эксперименты, показывающие основные возможности транзакций с различным уровнем изоляции.
6. Продемонстрировать результаты преподавателю, ответить на контрольные вопросы.
7. **Выполнение работы**

**А. Были проведены тесты по запуску, подтверждению и откату транзакций:**

Для тестирования была создана таблица temp с одним полем “tmp”, в это поле были добавлены два значения («8» и «28») после которого была создана точка сохранения save\_point1.



Были удалены все данные из таблицы, затем, вернувшись к точке сохранения, просмотрев данные в таблице мы увидели, что все данные вернулись. Потом, вернувшись к последнему подтверждению транзакции, в таблице осталось только одно значение «8».

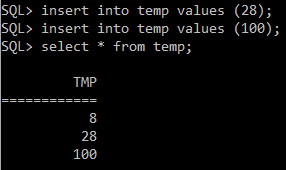


**B. Были проведены эксперименты, показывающие основные возможности транзакций с различным уровнем изоляции:**

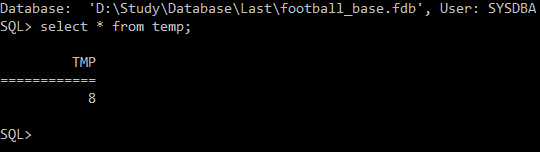
**Snapshot**

Это уровень изоляции по умолчанию. Он позволяет видеть неизменное состояние базы данных на момент старта транзакции. Изменения, выполненные другими транзакциями, в этой транзакции не видны. Свои изменения транзакция, разумеется, видит.

Один терминал:



Второй терминал:



По результатам работы видно, что при подключении двух клиентов один занес в таблицу два значения: 28 и 100, но второй клиент не видит этих изменений, пока транзакция не была завершена.

**Snapshot table stability**

Уровень изоляции транзакции SNAPSHOT TABLE STABILITY аналогичен уровню SNAPSHOT с той лишь разницей, что в данном случае другие транзакции независимо от их уровня изоляции могут только читать данные таблиц, включенных в операции этой транзакции, но не могут их изменять.

После того как уровень изоляции установлен, для того чтобы завершить транзакцию все клиентские транзакции должны быть завершены (commit;).

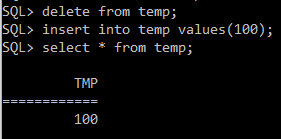
На первом терминале установим уровень изоляции:



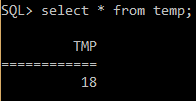
После данного действия, если у одного из клиентов есть незавершенные транзакции, мы не сможем завершить следующую транзакцию.

Просмотреть внесенные изменения, можно только после того как все другой клиент завершит транзакции:

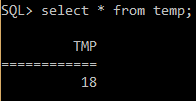
Первый терминал:



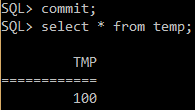
Второй терминал:



После *commit;* на первом терминале, на втором увидим те же данные:



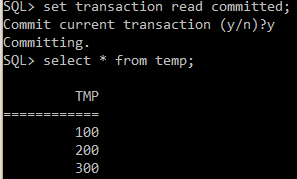
Чтобы увидеть изменения, завершение транзакции нужно и на втором терминале (после *commit;* на втором терминале):



**Read commited**

Уровень изолированности READ COMMITTED позволяет в транзакции без её перезапуска видеть все подтверждённые изменения данных базы данных, выполненные в других параллельных транзакциях. Неподтверждённые изменения не видны в транзакции и этого уровня изоляции.

Первый терминал:



Второй терминал:



После установления уровня изоляции **read committed**, на первом терминале возможно увидеть изменения данных сразу после подтверждения транзакции на втором терминале.

**Record\_Version**

Для этого уровня изоляции можно указать один из двух значений дополнительной характеристики в зависимости от желаемого способа разрешения конфликтов: RECORD\_VERSION и NO RECORD\_VERSION.

• NO RECORD\_VERSION (значение по умолчанию) является в некотором роде механизмом двухфазной блокировки. В этом случае транзакция не может прочитать любую запись, которая была изменена параллельной активной (неподтвержденной) транзакцией.

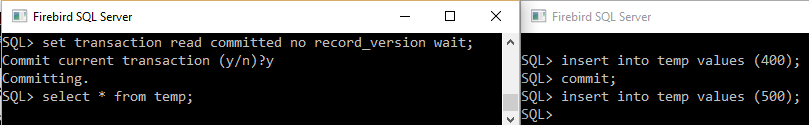
Если указана стратегия разрешения блокировок NO WAIT, то будет немедленно выдано соответствующее исключение. Если указана стратегия разрешения блокировок WAIT, то это приведёт к ожиданию завершения или откату конкурирующей транзакции.

• При задании RECORD\_VERSION транзакция всегда читает последнюю подтверждённую версию записей таблиц, независимо от того, существуют ли изменённые и ещё не подтверждённые версии этих записей. В этом случае режим разрешения блокировок (WAIT или NO WAIT) никак не влияет на поведение транзакции при её старте.

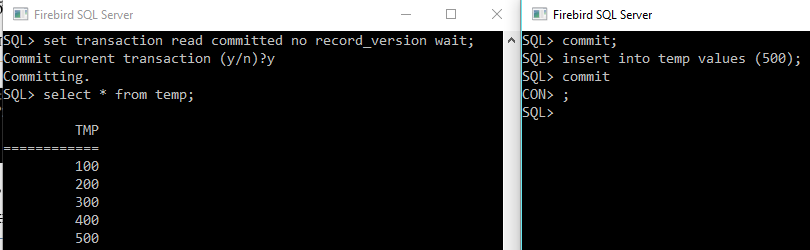
Изменим на одном терминале уровень изолированности:



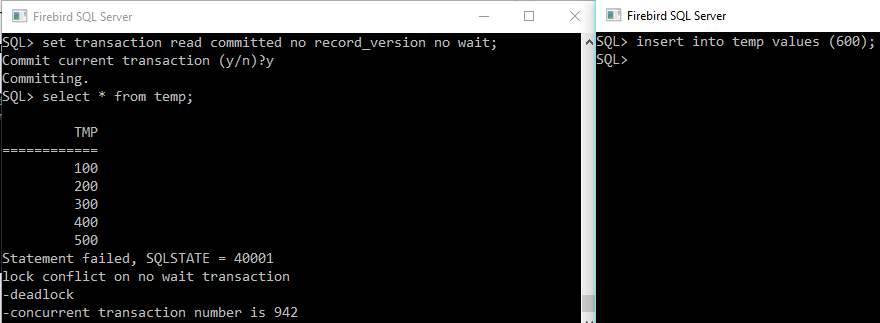
До того момента, когда на первом терминале не будет завершена транзакция, второй будет ожидать момента ее завершения и не увидит запрашиваемые данные:



Сразу после того, как на первом терминале будет завершена транзакция, второй увидит измененные данные, завершив команду select:



Изменим на первом терминале уровень изолированности:



При обращении первого терминала к данным таблицы, у него немедленно возникает соответствующее исключение.

1. **Вывод**

В данной лабораторной работе были изучены основные принципы работы транзакций и уровни их изоляции.

Дело в том, что при параллельном выполнении нескольких транзакций между ними могут возникать различные конфликты, такие как потерянное обновление, грязное чтение, неповторяющееся чтение и фантомное чтение. Для того, чтобы избежать этих конфликтов и существуют различные уровни изоляции транзакций.

Чем больше уровень изоляции транзакции – тем более изолированы транзакции друг от друга. При самом высоком уровне изоляции параллелизм в выполнении транзакций становится невозможен, так как любое параллельное выполнение транзакций может вызвать конфликт.

При работе с базой данных необходимо выбирать самый подходящий уровень изоляции транзакций, который будет золотой серединой между производительностью и надёжностью транзакций.