Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчёт по лабораторной работе**

**Дисциплина**: Базы данных

**Тема**: Изучение работы транзакций

Выполнил студент гр. 43501/4 Хрусталева М.С.

(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**

Транзакция (англ. transaction) — группа последовательных операций с базой данных, которая представляет собой логическую единицу работы с данными. Транзакция может быть выполнена либо целиком и успешно, соблюдая целостность данных и независимо от параллельно идущих других транзакций, либо не выполнена вообще и тогда она не должна произвести никакого эффекта. Для транзакций существует два режима доступа к данным базы данных: READ WRITE и READ ONLY.

• При режиме доступа READ WRITE операции в контексте данной транзакции могут быть как операциями чтения, так и операциями изменения данных. Это режим по умолчанию.

• В режиме READ ONLY в контексте данной транзакции могут выполняться только операции выборки данных SELECT. Любая попытка изменения данных в контексте такой транзакции приведёт к исключениям базы данных. Однако это не относится к глобальным временным таблицам (GTT), которые разрешено модифицировать в READ ONLY транзакциях.

При работе с одной и той же базой данных нескольких клиентских приложений могут возникать блокировки. Блокировки могут возникать, когда одна транзакция вносит неподтверждённые изменения в строку таблицы или удаляет строку, а другая транзакция пытается изменять или удалять эту же строку. Такие блокировки называются конфликтом обновления. Блокировки также могут возникнуть и в других ситуациях при использовании некоторых уровней изоляции транзакций.

Существуют два режима разрешения блокировок: WAIT и NO WAIT.

В режиме WAIT (режим по умолчанию) при появлении конфликта с параллельными транзакциями, выполняющими конкурирующие обновления данных в той же базе данных, такая транзакция будет ожидать завершения конкурирующей транзакции путём её подтверждения (COMMIT) или отката (ROLLBACK). Иными словами, клиентское приложение будет переведено в режим ожидания до момента разрешения конфликта.

Если установлен режим разрешения блокировок NO WAIT, то при появлении конфликта блокировки данная транзакция немедленно вызовет исключение базы данных.

Уровень изолированности транзакций — значение, определяющее уровень, при котором в транзакции допускаются несогласованные данные, то есть степень изолированности одной транзакции от другой. Изменения, внесённые некоторым оператором, будут видны всем последующим операторам, запущенным в рамках этой же транзакции, независимо от её уровня изолированности. Изменения произведённые в рамках другой транзакции остаются невидимыми для текущей транзакции до тех пор пока они не подтверждены. Уровень изолированности, а иногда, другие атрибуты, определяет, как транзакции будут взаимодействовать с другой транзакцией, которая хочет подтвердить изменения.

## **4. Ход работы**

1. Были изучены основные принципы работы транзакций;

2. Были проведены эксперименты по запуску, подтверждению и откату транзакций:

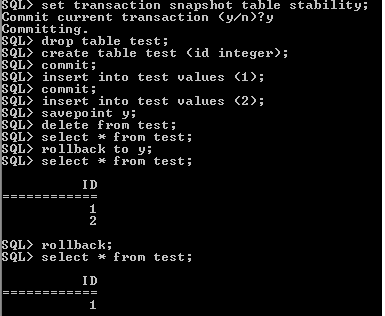


Рисунок 1: Опыты с подтверждением и запуском транзакций

3. Были изучены уровни изоляции транзакций в Firebird;

4. Были проведены эксперименты с различными уровнями изоляции транзакций.

Опыты с уровнем изоляции snapshot: позволяет видеть только те изменения, фиксация которых произошла не позднее момента старта этой транзакции. Любые подтверждённые изменения, сделанные другими конкурирующими транзакциями, не будут видны в такой транзакции в процессе ее активности без её перезапуска (рисунок 2).

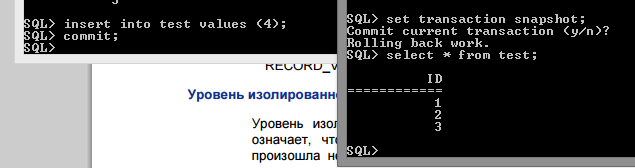


Рисунок 2: Два клиента: один выполнил вставку в таблицу, второй при этом не видит произведенных изменений

Опыты с уровнем изоляции snapshot table stability: позволяет видеть только те изменения, фиксация которых произошла не позднее момента старта этой транзакции. При этом после старта такой транзакции в других клиентских транзакциях невозможно выполнение изменений ни в каких таблицах этой базы данных, уже каким-либо образом измененных первой транзакцией (рисунки 3 – 5).

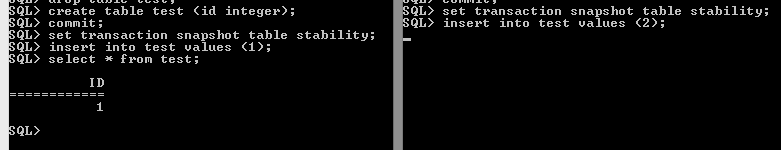


Рисунок 3: Два клиента: один выполнил вставку в таблицу, второй при этом не может завершить операцию вставки

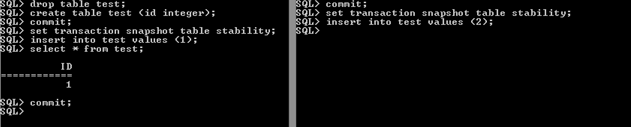


Рисунок 4: Два клиента: после того, как первый клиент зафиксировал изменения, второй смог выполнить операцию вставки

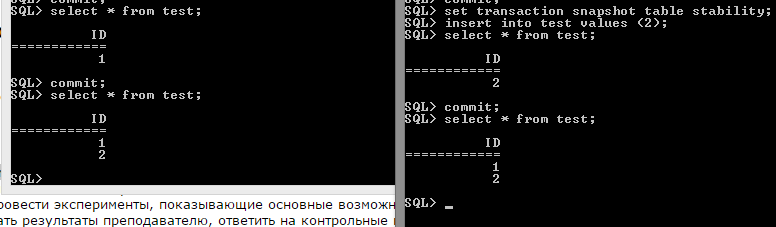


Рисунок 5: Два клиента: изменения, произведенные в других транзакциях видны только после того, как изменения были зафиксированы

Опыты с уровнем изоляции 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.

С опцией 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 (рисунок 6).

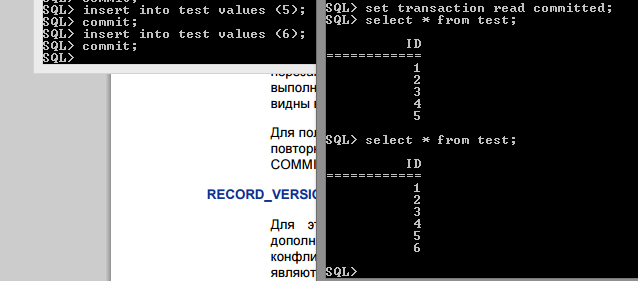


Рисунок 6: Два клиента: один выполнил вставку в таблицу, второй видит ее сразу после коммита

С опцией no record\_version wait: 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 (рисунки 7, 8).

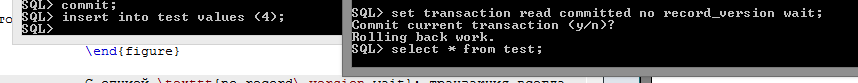


Рисунок 7: Два клиента: один выполнил вставку в таблицу, второй не может выполнить select пока первый не закоммитит

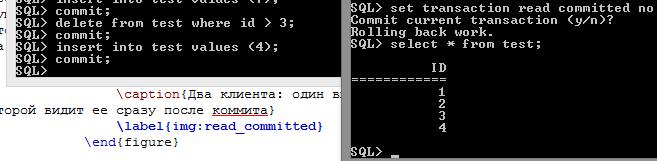


Рисунок 8: Два клиента: первый зафиксировал изменения, второй сразу завершил выполнение select

С опцией no record\_version no wait: при обращении к таблице, измененной в другой неподтвержденной транзакции, база выбросит исключение (рисунок 9).

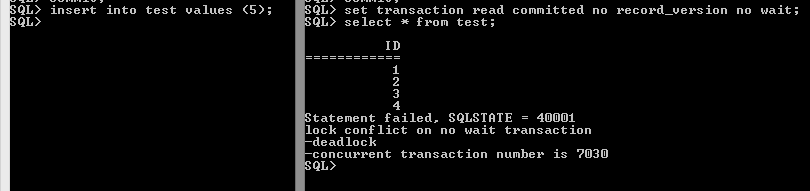


Рисунок 9: Два клиента: один выполнил вставку в таблицу, у второго вылетело исключение при обращении к этой таблице

**Вывод**

В ходе лабораторной работе мы изучили механизм транзакций, возможности ручного управления транзакциями, уровни изоляции транзакций.

Транзакция — логическая единица изолированной работы группы последовательных операций над базой данных.

Транзакция – неделимая с точки зрения воздействия на БД последовательность операторов манипулирования данными (чтения, удаления, вставки, модификации), такая, что:  
1) либо результаты всех операторов, входящих в транзакцию, отображаются в БД;  
2)либо воздействие всех операторов полностью отсутствует.  
При этом для поддержания ограничений целостности на уровне БД допускается их нарушение внутри транзакции так, чтобы к моменту завершения транзакции условия целостности были соблюдены.  
Для обеспечения контроля целостности каждая транзакция должна начинаться при целостном состоянии БД и должна сохранить это состояние целостным после своего завершения. Если операторы, объединенные в транзакцию, выполняются, то происходит нормальное завершение транзакции, и БД переходит в обновленное (целостное) состояние. Если же происходит сбой при выполнении транзакции, то происходит так называемый откат к исходному состоянию БД.

Транзакция обладает четырьмя важными свойствами, названными **АСИД**:

* **(А) Атомарность.** Транзакция выполняется как атомарная операция - либо выполняется вся транзакция целиком, либо она целиком не выполняется.
* **(С) Согласованность.** Транзакция переводит базу данных из одного согласованного (целостного) состояния в другое согласованное (целостное) состояние. Внутри транзакции согласованность базы данных может нарушаться.
* **(И) Изоляция.** Транзакции разных пользователей не должны мешать друг другу (например, как если бы они выполнялись строго по очереди).
* **(Д) Долговечность.** Если транзакция выполнена, то результаты ее работы должны сохраниться в базе данных, даже если в следующий момент произойдет сбой системы.

Ограничение целостности - это некоторое утверждение, которое может быть истинным или ложным в зависимости от состояния базы данных.

Ограничения целостности классифицируются несколькими способами:

* По способам реализации.
* По времени проверки.
* По области действия