Санкт-Петербургский государственный политехнический университет

Кафедра компьютерных систем и программных технологий

**Отчёт по лабораторной работе №7**

**Дисциплина**: Базы данных

# Изучение работы транзакций

Выполнил студент гр. 43501/3 Муравьев Ф.Э.

Преподаватель: Мяснов А. В.

Санкт-Петербург

2015

**Цели работы**

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

**Программа работы**

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

Работа проводится в IBExpert. Для проведения экспериментов параллельно запускается несколько сессий связи с БД, в каждой сессии настраивается уровень изоляции транзакций. Выполняются конкурентные операции чтения/изменения данных в различных сессиях, а том числе приводящие к конфликтам.

**Транзакция** группа последовательных операций с базой данных, которая представляет собой логическую единицу работы с данными. Транзакция может быть выполнена либо целиком 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 приведет 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 неподтвержденные 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.

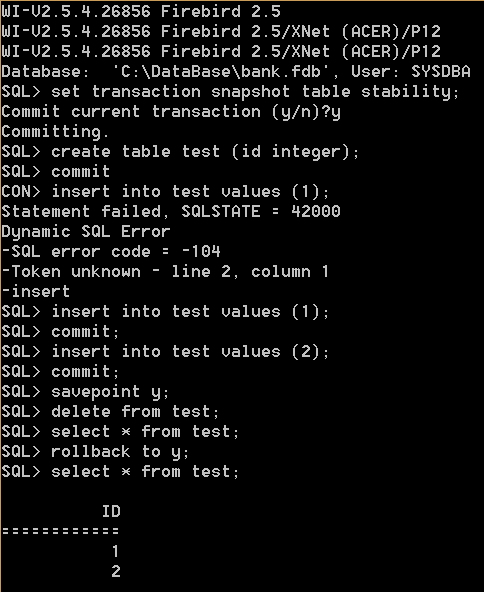
В режиме WAIT (режим по умолчанию) при появлении конфликта с параллельными транзакциями, выполняющими конкурирующие обновления данных в той же базе данных, такая транзакция будет ожидать завершения конкурирующей транзакции путем ее подтверждения (COMMIT) или отката (ROLLBACK). Иными словами, клиентское 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, внесенные некоторым оператором, будут видны всем последующим операторам, запущенным в рамках этой же транзакции, независимо от ее уровня изолированности. Изменения произведенные в рамках другой транзакции остаются невидимыми для текущей транзакции до тех пор пока они не подтверждены. Уровень изолированности, а иногда, другие атрибуты, определяют, как транзакции будут взаимодействовать с другой транзакцией, которая хочет подтвердить изменения.

**Выполнение работы**

1. Были изучены основные принципы работы транзакций;
2. Были проведены эксперименты по запуску, подтверждению и откату транзакций:

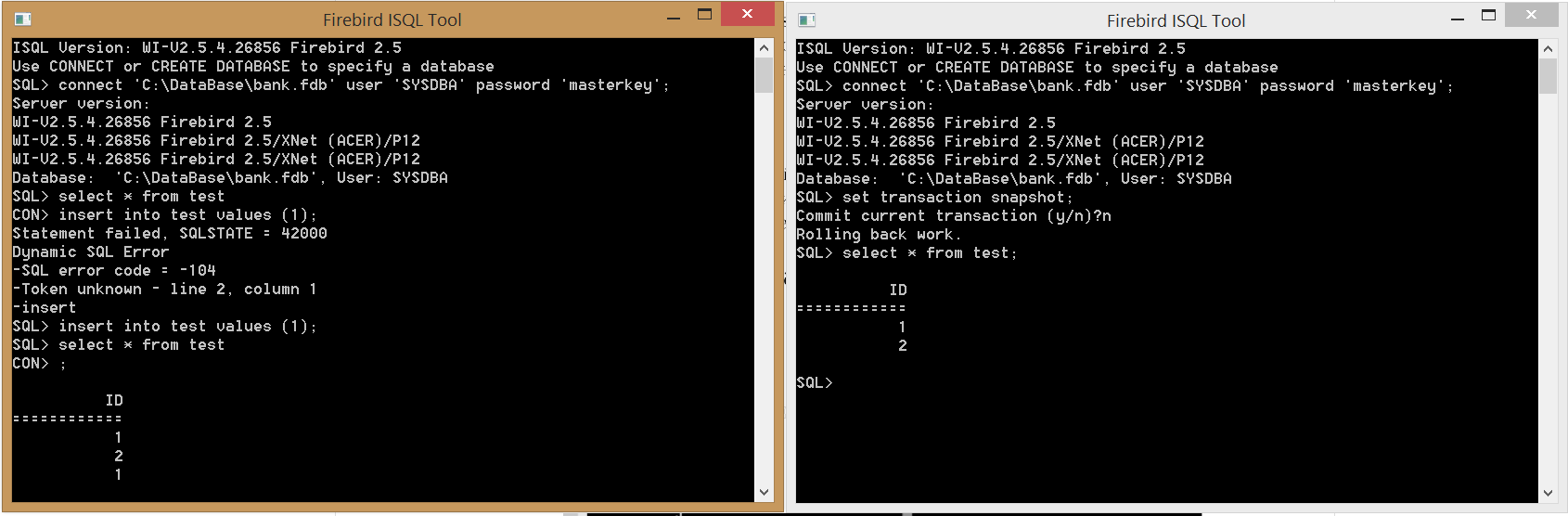


1. Были изучены уровни изоляции транзакций в Firebird;
2. Были проведены эксперименты с различными уровнями изоляции транзакций.

**Уровень изоляции snapshot:**

Позволяет видеть только те изменения, фиксация которых произошла не позднее момента старта этой транзакции. Любые подтвержденные изменения, сделанные другими конкурирующими транзакциями, не будут видны в такой транзакции в процессе ее активности без ее перезапуска.

Опыты с уровнем изоляции snapshot:



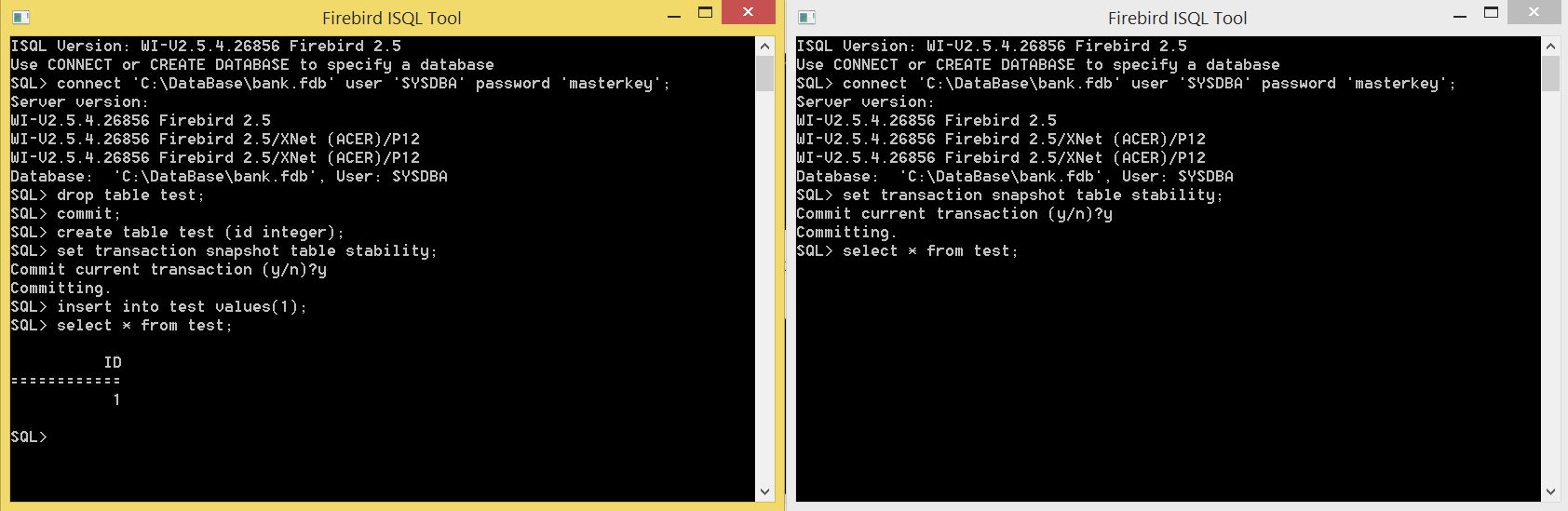
Два клиента: один выполнил вставку в таблицу, второй при этом не видит произведенных изменений.

**Уровень изоляции snapshot table stability:**

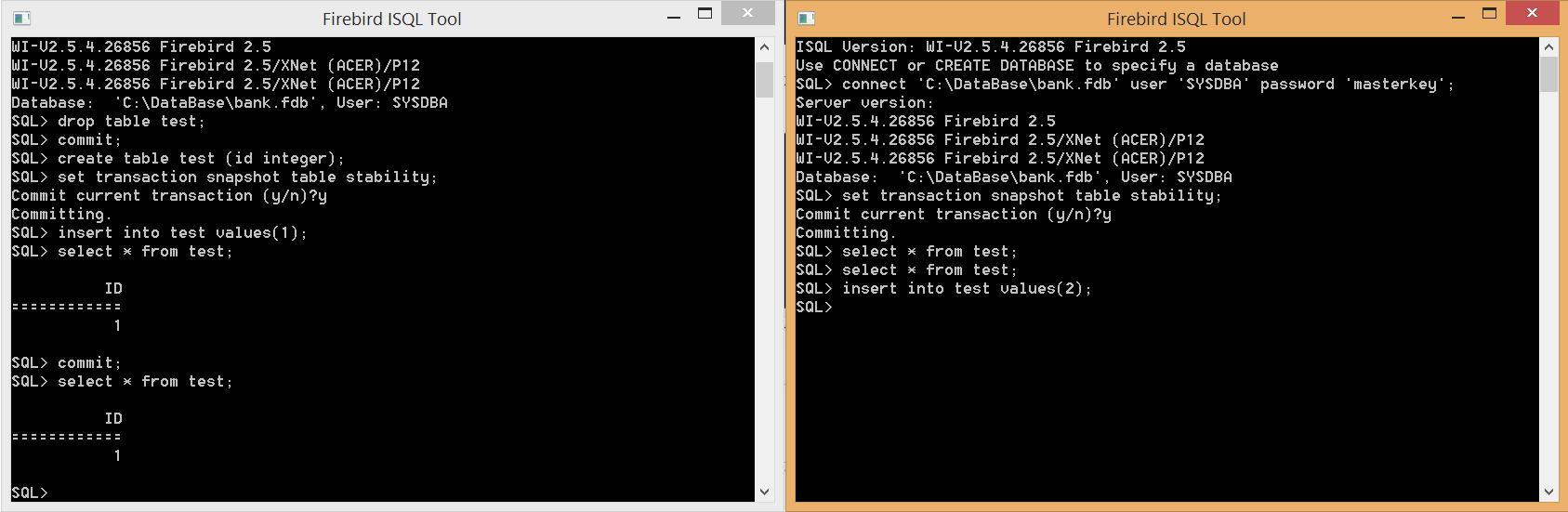
Позволяет видеть только те изменения, фиксация которых произошла не позднее момента старта этой транзакции. При этом после старта такой транзакции в других клиентских транзакциях невозможно выполнение изменений ни в каких таблицах этой базы данных, уже каким-либо образом измененных первой транзакцией.

Опыты с уровенем изоляции snapshot table stability:

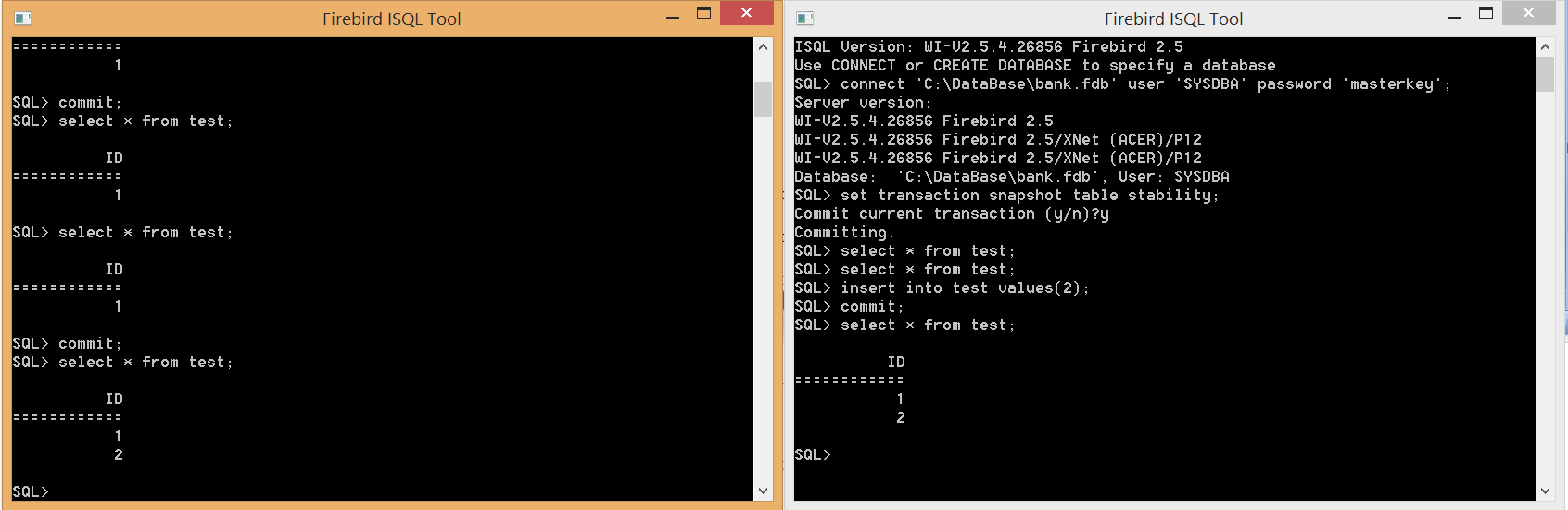
* Два клиента: один выполнил вставку в таблицу, второй при этом не может завершить операцию выборки:



* Два клиента: после того, как первый клиент зафиксировал изменения, второй смог выполнить операцию выборки:



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

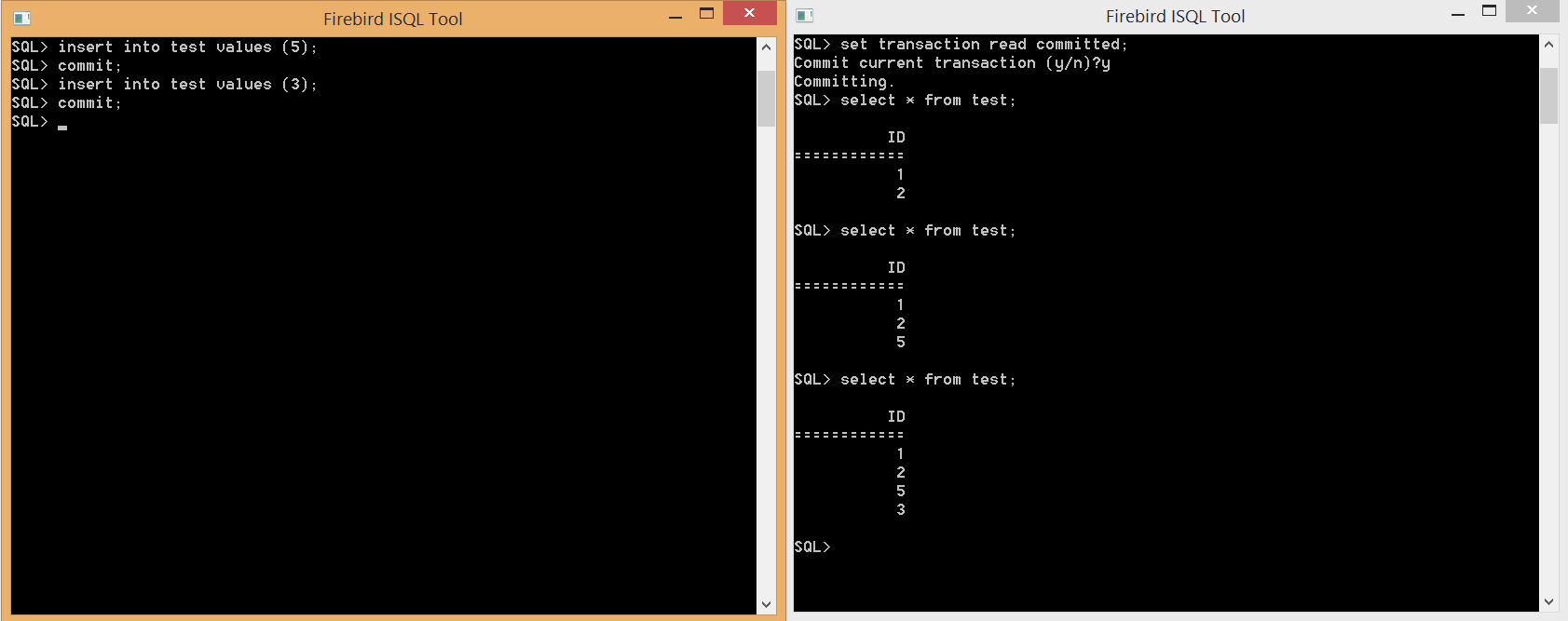


**Уровень изоляции read commited:**

Позволяет в транзакции без ее перезапуска видеть все подтвержденные изменения данных базы данных, выполненные в других параллельных транзакциях. Неподтвержденные изменения не видны в транзакции и этого уровня изоляции.

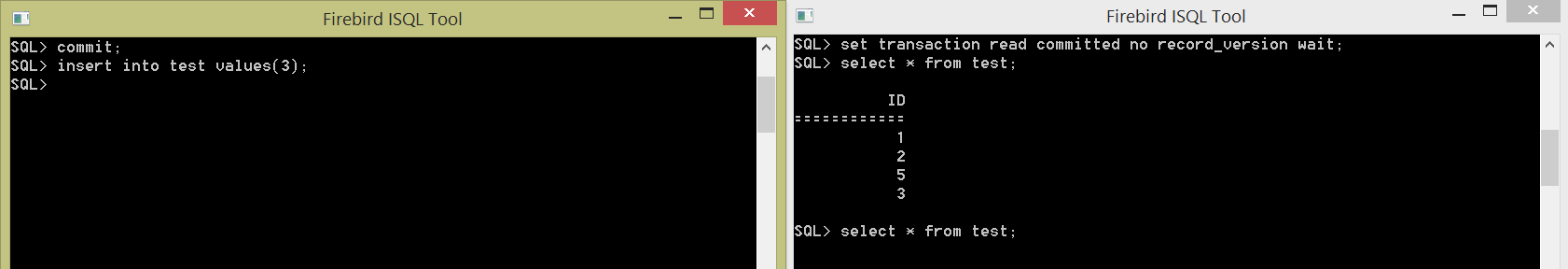
**С опцией record\_version:** транзакция всегда читает последнюю подтвержденную версию записей таблиц, независимо от того, существуют ли измененные и еще не подтвержденные версии этих записей. В этом случае режим разрешения блокировок (WAIT или NO WAIT) никак не влияет на поведение транзакции при ее старте.

Два клиента: один выполнил вставку в таблицу, второй видит ее сразу после commit:

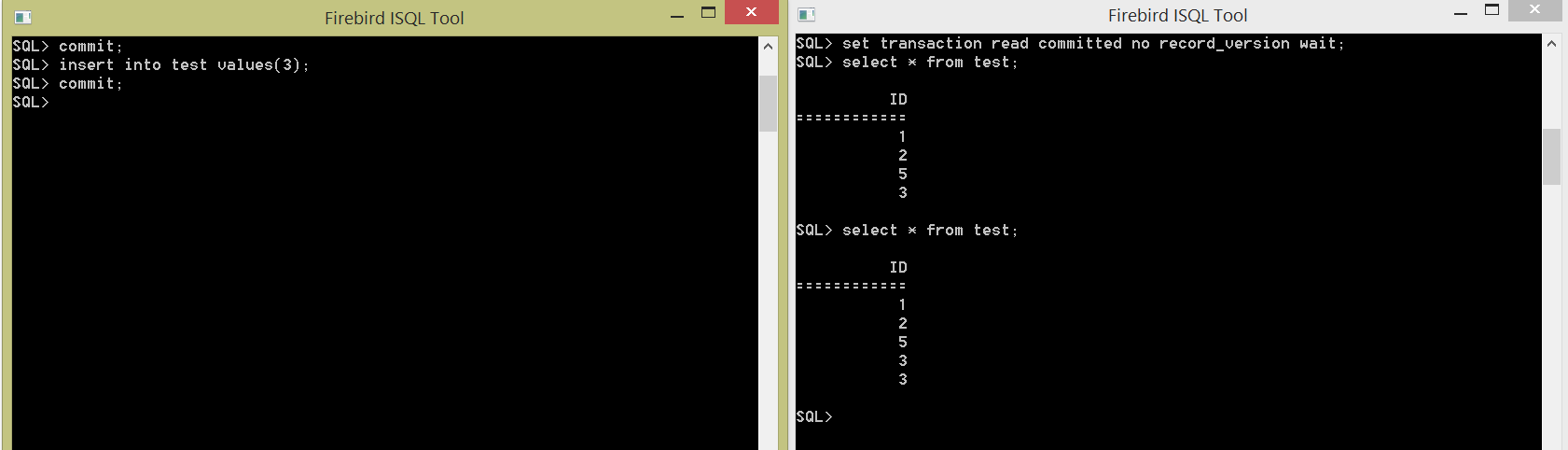


**С опцией no\_record\_version\_wait:** транзакция всегда читает последнюю подтвержденную версию записей таблиц, независимо от того, существуют ли измененные и еще не подтвержденные версии этих записей. В этом случае режим разрешения блокировок (WAIT или NO WAIT) нмкак не влияет на поведение транзакции при ее старте.

Два клиента: один выполнил вставку в таблицу, второй не может выполнить select пока первый не сделает commit:

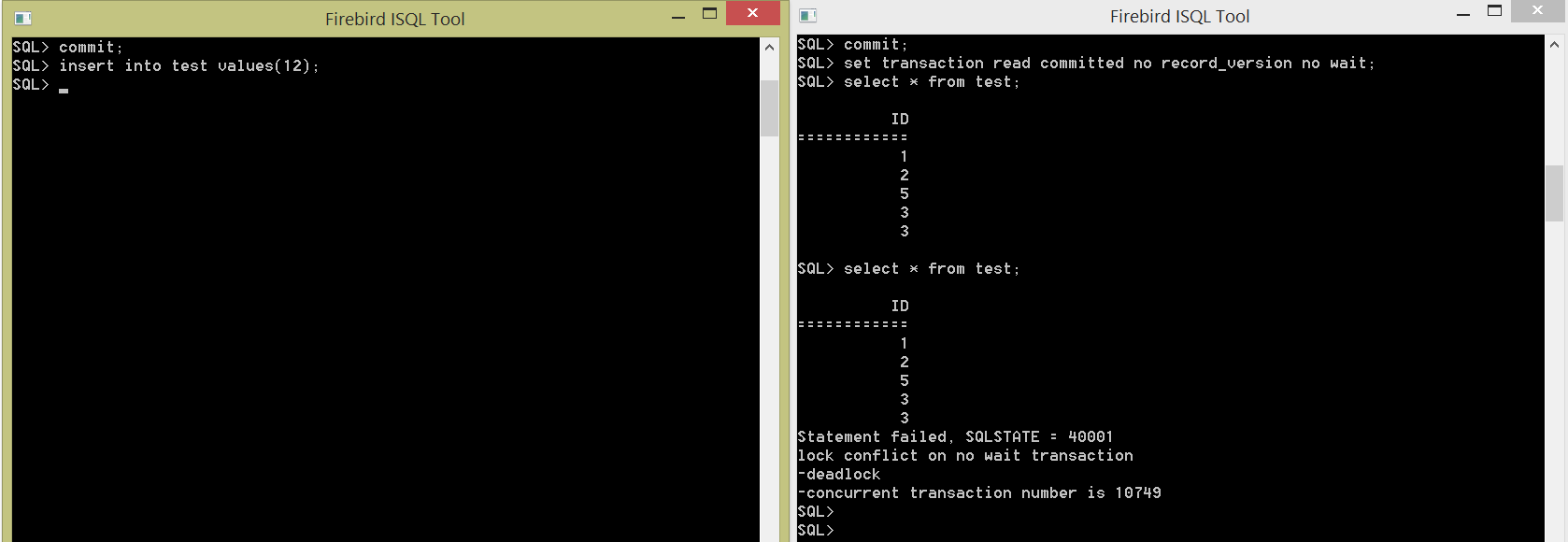


Первый зафиксировал изменения, второй сразу завершил выполнение select:



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

Два клиента: один выполнил вставку в таблицу, у вторго появилось исключение при обращении к этой таблице:



**Выводы.**

В данной работе был изучен механизм транзакций, возможности управления транзакциями и уровни изоляции транзакций в Firebird. Транзакции позволяют сохранять целостность БД при подключении к ней нескольких клиентов. Задавая различные типы разрешения конфликтов и уровни изоляции можно управлять видимостью изменений, произошедших в базе, для разных пользователей.