Теоретический материал к выполнению лабораторной работы

Firebird, как и все другие СУБД семейства InterBase реализует многоверсионную архитектуру – Multi Generation Architecture (MGA).

В модели MGA каждая строка, сохраняемая в базе данных, отмечается уникальным идентификатором транзакции, которая ее сохранила. Если новая транзакция посылает (post) изменение этой же строки, она записывается с идентификатором этой новой транзакции. Посылки, пока они не зафиксированы оператором `COMMIT`, видны только той транзакции, которая их создала.

После того как транзакция зафиксировала изменение строки, эти изменения будут видны всем транзакциям, которые начнут выполняться после момента фиксации.

Будут ли видны зафиксированные изменения, в стартовавших ранее и продолжающих выполняться транзакциях, определяется уровнем изоляции каждой из этих транзакций.

При таком подходе конфликт возникает только в том случае, когда несколько транзакций выполнили посылку (изменение) одной и той же строки и производится фиксация изменений одной из этих транзакций.

В таблице 1 представлено описание допустимых проблем для каждого уровня изоляции транзакций в СУБД Firebird.

Таблица 1. Уровни изоляции транзакций СУБД Firebird.

Уровень изоляции	**Проблема**				
	потерянные обновления	«грязное» чтение	неповторяю-щееся чтение	фантомы	несовместный анализ
READ COMMITTED RECORD_VERSION	нет	нет	да	да	да
READ COMMITTED NO RECORD_VERSION	нет	нет	нет	да	да
SNAPSHOT	нет	нет	нет	нет	нет
SNAPSHOT TABLE STABILITY	нет	нет	нет	нет	нет

Определение уровня изоляции транзакции осуществляется в начале транзакции с помощью оператора запуска транзакции. После завершения транзакции оператором COMMIT или ROLLBACK, начинается следующая транзакция. Если в начале транзакции не указан оператор запуска транзакции, то уровень изоляции будет определяться по умолчанию.

Синтаксис оператора запуска транзакции:

```
SET TRANSACTION [NAME имя]
[READ WRITE | READ ONLY]
[WAIT | NO WAIT]
[ISOLATION LAVEL]
{SNAPSHOT [TABLE STABILITY] |
 READ COMMITTED [[NO] RECORD_VERSION]}
[RESERVING предложение резервирования|
 USING дескрипторы баз данных]
```

Транзакция завершается выполнением оператора COMMIT или ROLLBACK.

Всякая операция с базой данных всегда выполняется в рамках транзакции. Если транзакция не запущена явно оператором SET TRANSACTION, то она выполняется в рамках транзакции по умолчанию.

Конфигурация транзакции по умолчанию:

```
READ WRITE WAIT SNAPSHOT.
```

Лабораторная работа 6. Конфигурирование транзакций

Задание 1. Для выполнения данной лабораторной работы транзакции необходимо выполнять через утилиту isql. Подключение к базе данных выполняется командой CONNECT. Команды вводятся в режиме диалога. Каждая команда может занимать более одной строки. Признаком завершения ввода команды является символ « ; ».

Ниже приведен пример сеанса работы с утилитой isql. Предполагается, что имя сервера, на котором расположена БД serverbd, вместо имени

возможно указание ip-адреса, d:\Dbase\employee.fdb – местоположение базы данных на сервере, имя пользователя SYSDBA, пароль masterkey. При выполнении лабораторной работы эти параметры соединения должны быть заменены на реально используемые. Утилита isql может быть запущена из каталога инсталляции клиента Firebird или через команду меню Пуск.

```
C:\Firebird\bin>isql
Use CONNECT or CREATE DATABASE to specify a database
SQL>CONNECT 'serverbd:d:\Dbase\employee.fdb'
CON> user 'SYSDBA'
CON> password 'masterkey';
Database:serverbd:d:\Dbase\employee.fdb, USER: SYSDBA
```

Проверяем возможность выполнения запросов к БД

```
SQL>select count(*) from employee;

        COUNT
=============
        42
```

Завершаем транзакцию

```
SQL> commit;
```

Продолжаем работу. . .

Для завершения работы выполняем корректный выход (с автоматическим завершением последней транзакции)

```
SQL> quit;
C:\Firebird\bin>
```

Если, не выходя из утилиты, ввести новую команду CONNECT, будет завершено текущее соединение и открыто новое. При этом перед закрытием текущего соединения необходимо явно завершить последнюю транзакцию.

Чтобы в БД нормально отображались русские буквы **ПЕРЕД СОЕДИНЕНИЕМ** с БД ввести команду

```
set names dos866;
```

Для того чтобы промоделировать при выполнении заданий конфликт транзакций, можно запустить одновременно два сеанса работы через утилиту isql с одной и той же базой. В компьютерных классах можно работать одновременно с одной базой с двух (или) более разных компьютеров.

Внимание! Убедитесь, что при проведении тестов параллельного выполнения транзакций никто больше не работает с теми таблицами, на которых выполняется тестирование.

Перед началом новой транзакции, предыдущая выполняющаяся транзакция должна быть завершена. После того как с базой данных установлено соединение, автоматически запускается транзакция по умолчанию. Чтобы завершить ее необходимо выполнить оператор `COMMIT`.

Для запуска транзакции с определенным уровнем изоляции используется оператор запуска транзакции.

Например, чтобы запустить транзакцию с уровнем изоляции `READ COMMITED`, нужно выполнить такую последовательность команд:

```
SQL>commit;
SQL>set transaction
CON>read write wait
CON>isolation level read committed record_version;
```

Многие параметры оператора запуска транзакции имеют значения по умолчанию. Например, в приведенном выше примере, можно было опустить параметры `read write`, `wait` и `record_version`, так как они определены по умолчанию. Однако чтобы быть уверенным в точной конфигурации транзакции, лучше все параметры задавать явно.

Научитесь запускать транзакции с разными уровнями изоляции.

Убедитесь, что при конфигурировании транзакции с параметром `read only`, в ней нельзя выполнять команды модификации таблиц (`INSERT`, `UPDATE`, `DELETE`).

В последующих заданиях для иллюстрации конфликтов приведены достаточно простые примеры для условных таблиц БД. Выполняя задания нужно использовать примеры для своих таблиц в тестовой базе данных. Примеры могут быть изменены в сторону усложнения.

Задание 2. По требованиям стандарта SQL СУБД ни на одном из уровней изоляции не должны допускать возникновения проблемы утраченных изменений. Эта проблема возникает, когда имеет место конфликт «запись – запись».

Чтобы проверить это, выполните для любой таблицы из тестовой базы данных две параллельные операции изменения одного и того же поля.

Пример графика параллельного выполнения транзакций может быть, например, таким.

Первая транзакция	Вторая транзакция
`select val1 from T where id=1;`	
	`select val1 from T where id=1;`
`update T set val1=val1+1 where id=1;`	
	`update T set val1=val1*2 where id=1;`
`select val1 from T where id=1;`	
	`select val1 from T where id=1;`
	`commit;`
`commit;`	

Уровень изоляции транзакций установите `READ COMMITTED RECORD_VERSION` (самый слабый в Firebird). Обе транзакции должны быть «чтения – записи». Чтобы не возникали тупиковые ситуации, режим ожидания установите `NO WAIT`.

Попробуйте изменить режим ожидания на `WAIT`. Что произойдет. Попробуйте использовать другие уровни изоляции транзакций.

Попробуйте поменять график выполнения транзакций в смеси. Изменится ли что-нибудь.

Запустите транзакции последовательно. Какой получится результат? Будет ли он зависеть от порядка запуска транзакций?

Попробуйте провести аналогичный эксперимент с запуском двух параллельных транзакций через IBExpert. Учтите, что при открытии окна работы с таблицей стартует транзакция с параметрами по умолчанию. В окне выполнения SQL–скриптов можно вводить последовательность команд SQL, в том числе и стартовать транзакции.

Задание 3. СУБД Firebird, в отличие от стандарта SQL, не допускает ни на одном из уровней изоляции возможности чтения незафиксированных изменений (чтение «грязных» данных). Установите уровни изоляции двух транзакций, как и в предыдущей задаче.

График выполнения транзакций должен быть аналогичен следующему:

Первая транзакция	Вторая транзакция
`update T set val1=val1+1`	
`where id=1;;`	
	`select val1 from T where id=1;`
`select val1 from T where id=1;`	
	`select val1 from T where id=1;`
`commit;`	
	`commit;`

Какой вывод можно сделать, проанализировав выполнение транзакций по этому графику?

Изменится ли что-нибудь, если вторую транзакцию сделать «только для чтения»?

Проведите эксперименты, выполняя конфигурирование транзакций для других уровней изоляции.

Задание 4. Для рассмотрения проблемы неповторяющегося чтения необходимо создать следующий график выполнения транзакций.

Первая транзакция	Вторая транзакция
	`select val1 from T where id=1;`

```
update T set val1=val1+1
         where id=1;
commit;
                                    select val1 from T where id=1;
                                    commit;
```

Выполните тестирование приведенного графика выполнения транзакций для разных вариантов их конфигурации. Оформите результаты тестирования в виде таблицы и сделайте выводы.

Проверьте, будет ли влиять на выполнение транзакций в смеси то, что вторая транзакция не требует конфигурации в режиме «чтение – запись».

Изменится ли результат тестирования, если в первой транзакции заменить операцию UPDATE на DELETE?

Задание 5. При выполнении в записывающей транзакции операции INSERT возможно возникновение «фантомных» чтений. Поэтому изменим тестируемые транзакции.

Пишущая транзакция	Читающая транзакция
	`select count(*) from T;`
`insert into T(id, val1)`	
`values(1000,1);`	
`commit;`	
	`select count(*) from T;`
	`commit;`

Выполните тестирование приведенного графика выполнения транзакций для разных вариантов их конфигурации. Оформите результаты тестирования в виде таблицы и сделайте выводы.

Проверьте, будет ли влиять на выполнение транзакций в смеси то, что вторая транзакция не требует конфигурации в режиме «чтение – запись».

Изменится ли результат тестирования, если в первой транзакции заменить операцию INSERT на DELETE?

Задание 6. Проблему собственно несовместного анализа чаще всего демонстрируют на примере конфликта длиной читающей транзакции (определение суммы на всех счетах в банке) и короткой записывающей транзакции (перенос суммы с одного счета на другой). Придумайте самостоятельно вариант двух параллельно запускаемых транзакций и график их выполнения, чтобы увидеть конфликт. Протестируйте этот график при разных вариантах конфигурации транзакций. Оформите результаты тестирования в виде таблицы и сделайте выводы.