Лабораторная работа. Транзакции.

***Цель работы***

Используя данные базы данных, подготовленной в предыдущей лабораторной работе, подготовить и реализовать серию запросов, связанных с выборкой информации и модификацией данных таблиц.

***Теоретические сведения***

Изменения БД часто требуют выполнения нескольких запросов, например при покупке в электронном магазине требуется добавить запись в таблицу заказов и уменьшить число товарных позиций на складе. В промышленных БД одно событие может затрагивать большее число таблиц и требовать многочисленных запросов.

Если на этапе выполнения одного из запросов происходит сбой, это может нарушить целостность БД (товар может быть продан, а число товарных позиций на складе не обновлено). Чтобы сохранить целостность БД, все изменения должны выполняться как единое целое. Либо все изменения успешно выполняются, либо, в случае сбоя, БД принимает состояние, которое было до начала изменений. Это обеспечивается средствами обработки транзакций.

# ***Задание на оценку 4***

Используя базу, полученную в лабораторной 2, создать транзакцию, произвести ее откат и фиксацию. Показать, что данные существовали до отката, удалились после отката, снова были добавлены, и затем были успешно зафиксированы. При необходимости используйте точки сохранения и вложенные транзакции.

# ***Задание на оценку 5***

Подготовить SQL-скрипты для выполнения проверок изолированности транзакций. Ваши скрипты должны работать с одной из таблиц, созданных в лабораторной работе №2.

***Выполнение работы***

1. Запустить клиента и соединиться с базой данных. Открыть второе окно для ввода текста запросов (Ctrl+N в первом окне).
2. Установить в обоих сеансах уровень изоляции READ UNCOMMITTED. Выполнить сценарии проверки:

* потерянных изменений,
* грязного чтения.

1. Записать протокол выполнения сценариев.
2. Установить в обоих сеансах уровень изоляции READ COMMITTED. Выполнить сценарии проверки:

* грязного чтения.
* неповторяющегося чтения.

1. Записать протокол выполнения сценариев.
2. Установить в обоих сеансах уровень изоляции REPEATABLE READ. Выполнить сценарии проверки:

* неповторяющегося чтения,
* фантома.

1. Записать протокол выполнения сценариев.
2. Установить в обоих сеансах уровень изоляции SERIALIZABLE. Выполнить сценария проверки

* фантома.

1. Записать протокол выполнения сценария.

***Содержание отчета:***

Сценарий и протокол его выполнения.

Краткие выводы о навыках, приобретенных в ходе выполнения работы.

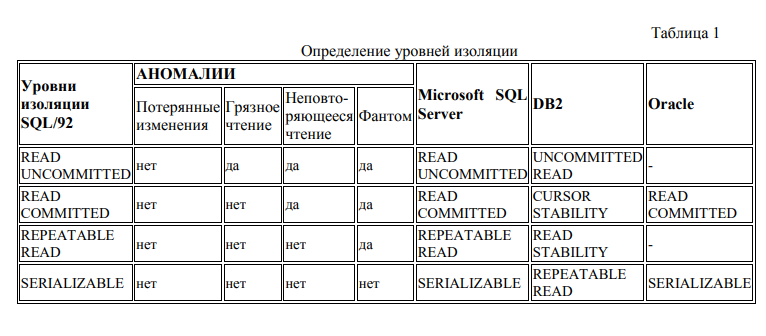
***Теоретические сведения***

*Уровни изоляции*

Стандарт SQL/92 определяет уровни изоляции транзакций в многопользовательской системе через отсутствие таких аномалий доступа к базе данных, которые могут в конечном итоге угрожать целостности данных. В стандарте различаются следующие аномалии:

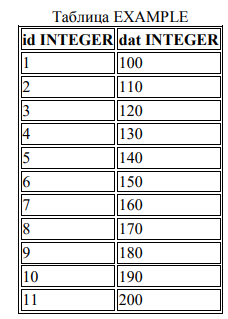
* ***Потерянные изменения.*** Транзакция Т1 читает данные. Транзакция Т2 читает те же данные. Транзакция T1 на основании прочитанного значения вычисляет новое значение данных, записывает его в базу данных и завершается. Транзакция T2 на основании прочитанного значения вычисляет новое значение данных, записывает его в базу данных и завершается. В результате значение, записанное транзакцией Т2, "затрет" значение, записанное транзакцией Т1.
* ***Грязное чтение.*** Транзакция Т1 изменяет некоторые данные, но еще не завершается. Транзакция Т2 читает эти же данные (с изменениями, внесенными транзакцией Т1) и принимает на их основе какие-то 3 решения. Транзакция Т1 выполняет откат. В результате решение, принятое транзакцией Т2 основано на неверных данных.
* ***Неповторяющееся чтение.*** Транзакция Т1 в ходе своего выполнения несколько раз читает одни и те же данные. Транзакция Т2 в интервалах между чтениями транзакцией Т1 изменяет эти данные и фиксируется. В результате оказывается, что чтения одних и тех же данных в транзакции Т1 дает разные результаты.
* ***Фантом.*** Транзакция Т1 в ходе своего выполнения несколько раз выбирает множество строк по одним и тем же критериям. Транзакция Т2 в интервалах между выборками транзакции Т1 добавляет или удаляет строки или изменяет столбцы некоторых строк, используемых в критерии выборки, и фиксируется. В результате оказывается, что одни и те же выборки в транзакции Т1 выбирают разные множество строк.

Промышленные СУБД в том или ином объеме выполняют требования стандарта по дифференциации уровней изоляции, но при формально одном и том же уровне изоляции поведение транзакций может существенно различаться в разных СУБД. Определение уровней изоляции в стандарте и в рассматриваемых нами СУБД сведено в табл. 1:

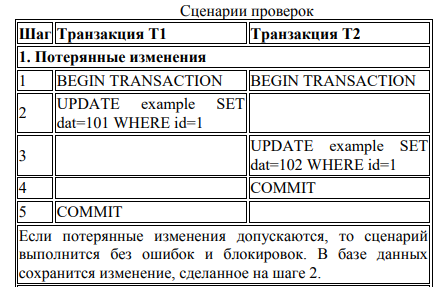


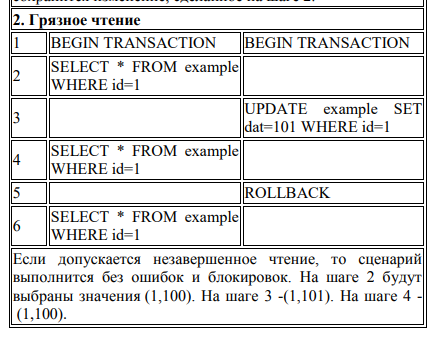
***Сценарии***

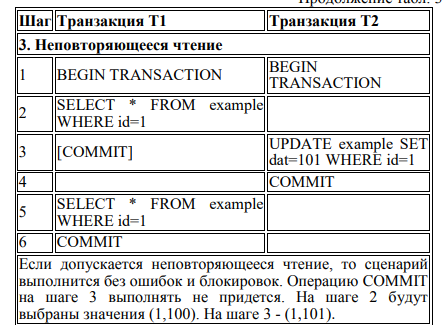
Рассмотрим сценарии проверки нежелательных ситуаций на примере таблицы EXAMPLE, структура и содержимое которой приведены ниже. Вам предстоит разработать подобные сценарии, работающие с одной из таблиц, созданных Вами в работе № 2.

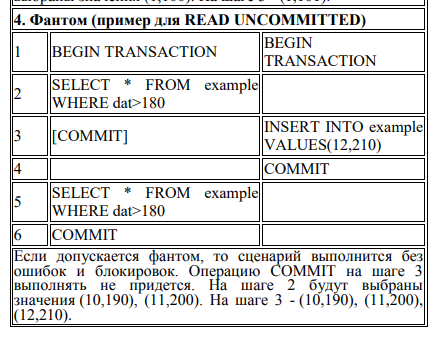


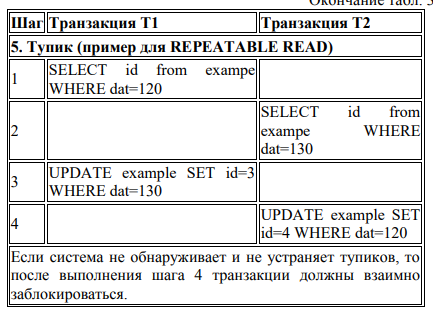
Ниже приводятся сценарии проверок . Сценарии должны выполняться пошагово, что приводит к тому, что транзакции Т1 и Т2 выполняются параллельно в разных сеансах. Мы подразумеваем, что после выполнения каждого сценария мы восстанавливаем исходное содержимое таблицы EXAMPLE.











***Инструментальные средства Microsoft SQL Server***

Для выполнения сценариев проверки изолированности следует открыть два окна Для того чтобы набор операторов выполнялся внутри транзакции, следует заключить их между строчками

BEGIN TRANSACTION и COMMIT:

BEGIN TRANSACTION

...

...

COMMIT

Установка уровня изоляции выполняется командой:

SET TRANSACTION ISOLATION LEVEL *уровень\_изоляции*

уровень изоляции может принимать значения: READ UNCOMMITTED / READ COMMITTED / REPEATABLE READ / SERIALIZABLE

Команду, выполняющую изменение уровня изоляции, следует разместить первой в скрипте. В процессе выполнения задания в каждом из окон следует выделять только строчки, присутствующие на данном шаге, и нажимать на кнопку «выполнить». В процессе работы после нажатия на эту кнопку выполнение дальнейших команд может быть заблокировано (становится активна только кнопка «остановить текущую операцию»). В этом случае результат выполнения зависит от параллельной транзакции, и может быть вычислен только при её завершении (COMMIT/ROLLBACK).