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.

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

- грязного чтения.
- неповторяющегося чтения.

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

- неповторяющегося чтения,

- фантома.

7. Записать протокол выполнения сценариев.
8. Установить в обоих сеансах уровень изоляции SERIALIZABLE. Выполнить сценария проверки
   - фантома.
9. 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 T1 читает данные. Транзакция T2 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 T2, "затрет" значение, записанное транзакцией T1.
- ***Грязное чтение.*** Транзакция T1 изменяет некоторые данные, но еще не завершается. Транзакция T2 читает эти же данные (с изменениями, внесенными транзакцией T1) и принимает на их основе какие-то 3 решения. Транзакция T1 выполняет откат. В результате решение, принятое транзакцией T2 основано на неверных данных.
- ***Неповторяющееся чтение.*** Транзакция T1 в ходе своего выполнения несколько раз читает одни и те же данные. Транзакция T2 в интервалах между чтениями транзакцией T1 изменяет эти данные и фиксируется. В результате оказывается, что чтения одних и тех же данных в транзакции T1 дает разные результаты.
- ***Фантом.*** Транзакция T1 в ходе своего выполнения несколько раз выбирает множество строк по одним и тем же критериям. Транзакция T2 в интервалах между выборками транзакции T1 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 T1 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:

Таблица 1

Определение уровней изоляции

Уровни изоляции SQL/92	**АНОМАЛИИ**				**Microsoft SQL Server**	**DB2**	**Oracle**
	Потерянные изменения	Грязное чтение	Неповто-ряющееся чтение	Фантом			
READ UNCOMMITTED	нет	да	да	да	READ UNCOMMITTED	UNCOMMITTED READ	-
READ COMMITTED	нет	нет	да	да	READ COMMITTED	CURSOR STABILITY	READ COMMITTED
REPEATABLE READ	нет	нет	нет	да	REPEATABLE READ	READ STABILITY	-
SERIALIZABLE	нет	нет	нет	нет	SERIALIZABLE	REPEATABLE READ	SERIALIZABLE

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.

Таблица EXAMPLE

id INTEGER	**dat INTEGER**
1	100
2	110
3	120
4	130
5	140
6	150
7	160
8	170
9	180
10	190
11	200

Ниже приводятся сценарии проверок . Сценарии должны выполняться пошагово, что приводит к тому, что транзакции T1 и T2 выполняются параллельно в разных сеансах. Мы подразумеваем, что после выполнения каждого сценария мы восстанавливаем исходное содержимое таблицы EXAMPLE.

Сценарии проверок

Шаг	Транзакция Т1	Транзакция Т2
1. Потерянные изменения		
1	BEGIN TRANSACTION	BEGIN TRANSACTION
2	UPDATE example SET dat=101 WHERE id=1	
3		UPDATE example SET dat=102 WHERE id=1
4		COMMIT
5	COMMIT	
Если потерянные изменения допускаются, то сценарий выполнится без ошибок и блокировок. В базе данных сохранится изменение, сделанное на шаге 2.		

2. Грязное чтение		
1	BEGIN TRANSACTION	BEGIN TRANSACTION
2	SELECT * FROM example WHERE id=1	
3		UPDATE example SET dat=101 WHERE id=1
4	SELECT * FROM example WHERE id=1	
5		ROLLBACK
6	SELECT * FROM example WHERE id=1	
Если допускается незавершенное чтение, то сценарий выполнится без ошибок и блокировок. На шаге 2 будут выбраны значения (1,100). На шаге 3 -(1,101). На шаге 4 - (1,100).		

Продолжение табл. 3

Шаг	Транзакция T1	Транзакция T2
3. Неповторяющееся чтение		
1	BEGIN TRANSACTION	BEGIN TRANSACTION
2	SELECT * FROM example WHERE id=1	
3	[COMMIT]	UPDATE example SET dat=101 WHERE id=1
4		COMMIT
5	SELECT * FROM example WHERE id=1	
6	COMMIT	
Если допускается неповторяющееся чтение, то сценарий выполнится без ошибок и блокировок. Операцию COMMIT на шаге 3 выполнять не придется. На шаге 2 будут выбраны значения (1,100). На шаге 3 - (1,101).		

4. Фантом (пример для READ UNCOMMITTED)		
1	BEGIN TRANSACTION	BEGIN TRANSACTION
2	SELECT * FROM example WHERE dat>180	
3	[COMMIT]	INSERT INTO example VALUES(12,210)
4		COMMIT
5	SELECT * FROM example WHERE dat>180	
6	COMMIT	
Если допускается фантом, то сценарий выполнится без ошибок и блокировок. Операцию COMMIT на шаге 3 выполнять не придется. На шаге 2 будут выбраны значения (10,190), (11,200). На шаге 3 - (10,190), (11,200), (12,210).		

Окончание табл. 3

Шаг	Транзакция T1	Транзакция T2
5. Тупик (пример для REPEATABLE READ)		
1	SELECT id from exampe WHERE dat=120	
2		SELECT id from exampe WHERE dat=130
3	UPDATE example SET id=3 WHERE dat=130	
4		UPDATE example SET id=4 WHERE dat=120
Если система не обнаруживает и не устраняет тупиков, то после выполнения шага 4 транзакции должны взаимно заблокироваться.		

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).