ГУАП

КАФЕДРА № 42

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| канд. тех. наук, доцент |  |  |  | А. В. Бржезовский |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
| ТРАНЗАКЦИИ И БЛОКИРОВКИ |
| по курсу: МЕТОДЫ И СРЕДСТВА ПРОЕКТИРОВАНИЯ  ИНФОРМАЦИОННЫХ СИСТЕМ И ТЕХНОЛОГИЙ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ гр. № | 4121 |  |  |  | А. М. Антонов |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2024

**Задание**

Смоделировать в БД грязное чтение, неповторяемое чтение, фантомы, изменяя уровень изоляции транзакций продемонстрировать их исключение, сформировать отчеты о блокировках, пояснить их содержание.

Смоделировать в БД тупик (взаимную блокировку), получить с помощью приложения SQL Server Profiler отчет о тупике, пояснить его содержание.

Индивидуальный вариант задания №5: ВУЗ, студент, группа, факультет, конференция, тема доклада, программа конференции.

**Ход работы**

1. Грязное чтение

«Грязное» чтение происходит, когда транзакция получает доступ к незафиксированным данным, которые были изменены другой транзакцией. Это явление возможно только при самом низком уровне изоляции — read uncommitted.

Для проверки «грязного» чтения разработаны скрипты для двух транзакций, выполняющихся одновременно в двух разных окнах в базе данных:

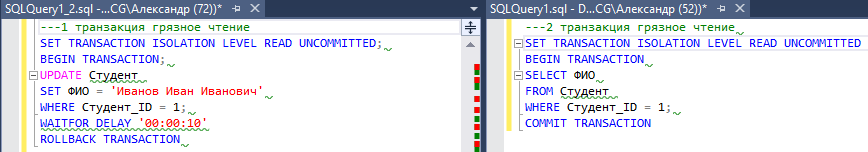


Рисунок 1 – «Грязное» чтение

На рисунке 2 представлен отчет о блокировках, на рисунке 3 – результат запроса, выдаваемый второй транзакцией. На рисунках 4-5 показаны отчет о блокировках и верный результат запроса.

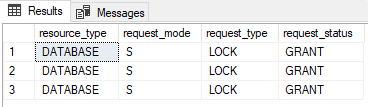


Рисунок 2 – Отчет о блокировках

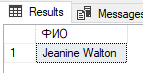


Рисунок 3 – Неверный результат запроса

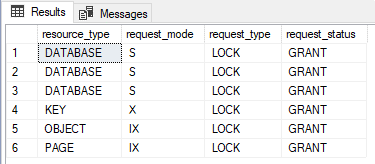


Рисунок 4 – Отчет о блокировках

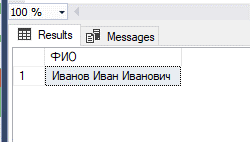


Рисунок 5 – Результат запроса

1. Неповторяемое чтение

При неповторяемом чтении одна транзакция читает данные, а затем другая транзакция изменяет эти данные до того, как первая транзакция завершится. Если первая транзакция попытается снова прочитать те же данные, она получит уже изменённое значение.

На рисунке 6 показаны две транзакции для симуляции неповторяемого чтения.

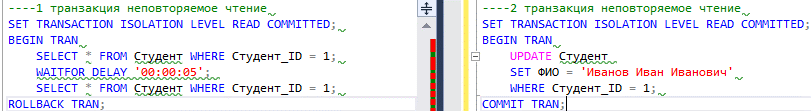


Рисунок 6 – Неповторяемое чтение

На рисунках 7-8 показаны отчет о блокировках и результат выполнения двух запросов.

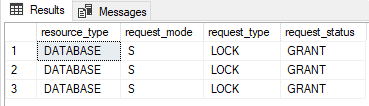


Рисунок 7 – Отчет о блокировках

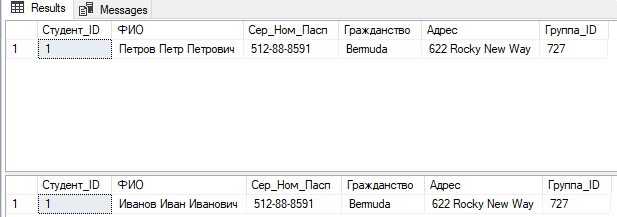


Рисунок 8 – Результат запроса

Проблему неповторяемого чтения можно решить с помощью повышения уровня изоляции до repeatable read. На рисунках 9-10 показаны отчет о блокировках после повышения уровня изоляции и результат запроса.

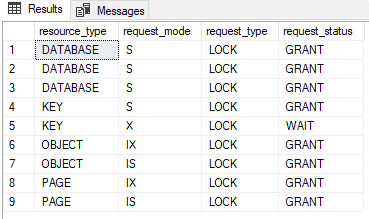


Рисунок 9 – Отчет о блокировках

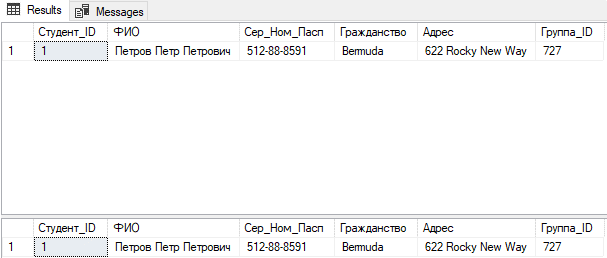


Рисунок 10 – Результат запроса

1. Фантомные строки

Фантомные строки — это тип аномалии, возникающей в многопользовательских системах баз данных при выполнении транзакций. Эта аномалия происходит, когда одна транзакция выполняет выборку данных, а другая транзакция добавляет или удаляет строки, которые соответствуют критериям выборки первой транзакции.

На рисунке 11 показаны две транзакции для симуляции возникновения фантомных строк.

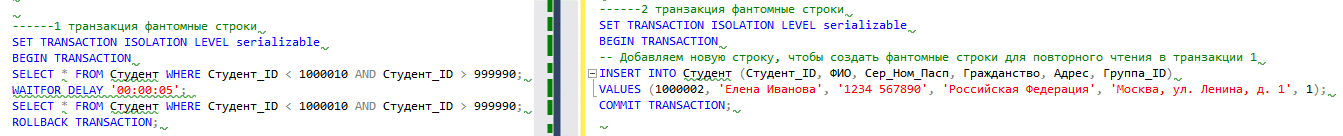


Рисунок 11 – Симуляция фантомных строк

На рисунках 12-13 показаны отчет о блокировках и результат выполнения двух запросов.

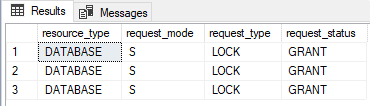


Рисунок 12 – Отчет о блокировках

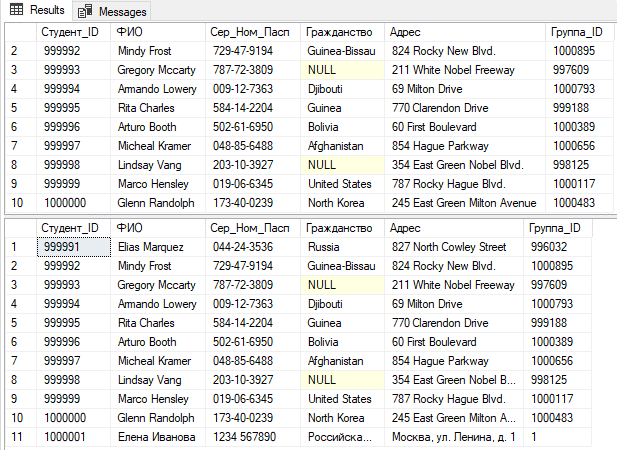


Рисунок 13 – Неверный результат запроса

Исключить появление фантомных строк можно с помощью уровня изоляции serializable. На рисунках 14-15 показаны отчет о блокировках и результат запроса с повышенным уровнем изоляции.

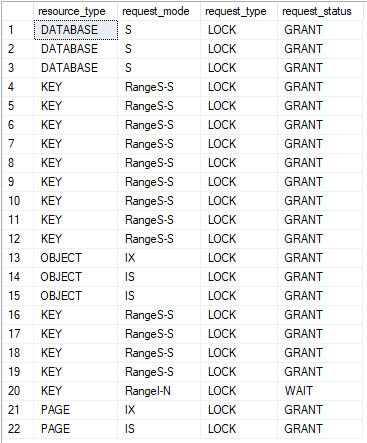


Рисунок 14 – Отчет о блокировках

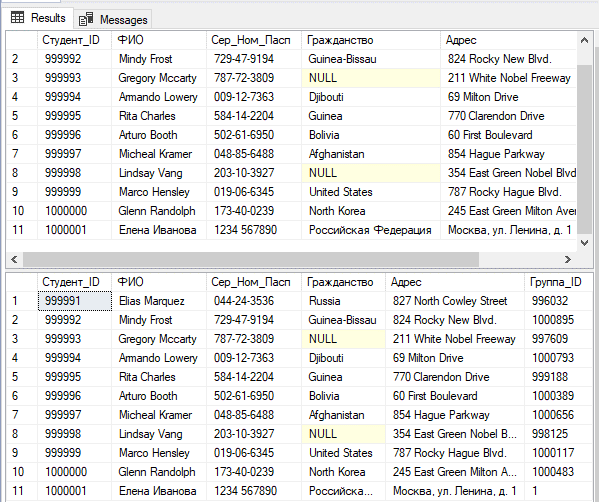


Рисунок 15 – Результат запроса

1. Взаимоблокировка

Взаимоблокировки (или дедлоки) возникают в многопользовательских системах, когда две или более транзакции ждут друг друга, чтобы завершить свои операции. Это приводит к ситуации, в которой ни одна из транзакций не может продолжить выполнение, так как каждая из них удерживает ресурсы, необходимые другой.

На рисунке 16 представлены две транзакции, симулирующие взаимоблокировку.

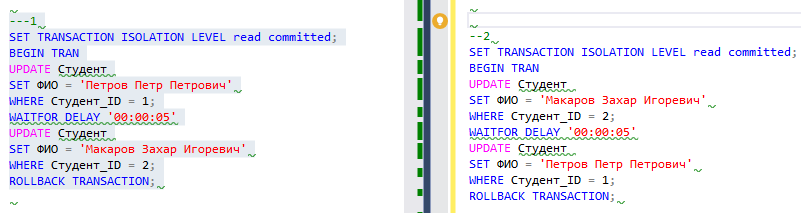


Рисунок 16 – Симуляция взаимоблокировки

На рисунке 17 представлены результаты двух транзакций.

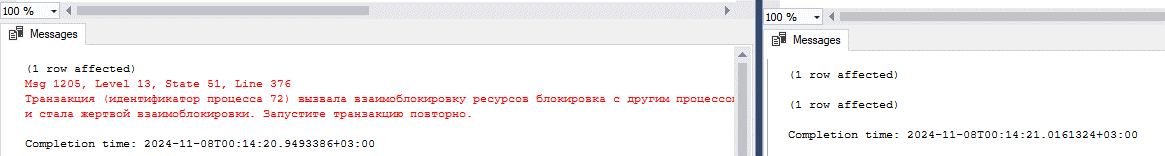


Рисунок 17 – Результат выполнения запросов

График зависимости транзакций показывает, как одна транзакция ожидает снятия блокировки другой. Если в графе обнаруживается цикл, одна из транзакций принудительно откатывается и перезапускается спустя некоторое время, либо возникает исключительная ситуация. Обычно жертвой становится та транзакция, которая внесла наименьшее количество изменений в базу данных.

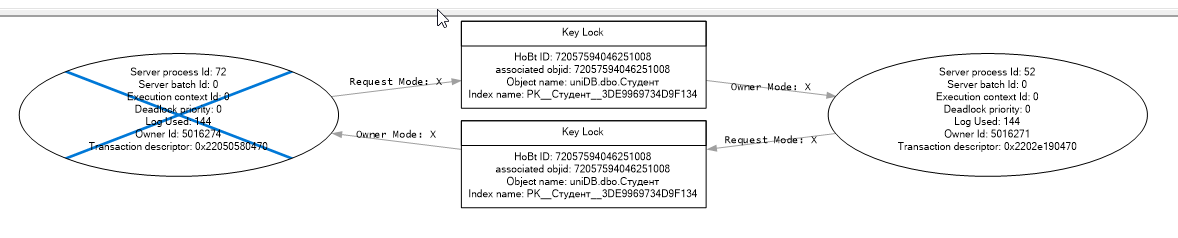


Рисунок 18 – Отчет о тупике

**Вывод**

В процессе выполнения лабораторной работы было исследовано, как используются транзакции и решаются проблемы, возникающие из-за многопользовательского доступа к данным с помощью механизмов блокировок. Это способствовало более глубокому пониманию организации транзакций в базах данных и методов предотвращения конфликтов при параллельном доступе к информации.

Приложение 1. Листинг файла transactions.sql

---1 транзакция грязное чтение

SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;

BEGIN TRANSACTION;

UPDATE Студент

SET ФИО = 'Иванов Иван Иванович'

WHERE Студент\_ID = 1;

WAITFOR DELAY '00:00:10'

ROLLBACK TRANSACTION

----1 транзакция неповторяемое чтение

SET TRANSACTION ISOLATION LEVEL repeatable read;

BEGIN TRAN

SELECT \* FROM Студент WHERE Студент\_ID = 1;

WAITFOR DELAY '00:00:05';

SELECT \* FROM Студент WHERE Студент\_ID = 1;

ROLLBACK TRAN;

------1 транзакция фантомные строки

SET TRANSACTION ISOLATION LEVEL serializable

BEGIN TRANSACTION

SELECT \* FROM Студент WHERE Студент\_ID < 1000010 AND Студент\_ID > 999990;

WAITFOR DELAY '00:00:05';

SELECT \* FROM Студент WHERE Студент\_ID < 1000010 AND Студент\_ID > 999990;

ROLLBACK TRANSACTION;

---1 транзакция Взаимоблокировка

SET TRANSACTION ISOLATION LEVEL read committed;

BEGIN TRAN

UPDATE Студент

SET ФИО = 'Петров Петр Петрович'

WHERE Студент\_ID = 1;

WAITFOR DELAY '00:00:05'

UPDATE Студент

SET ФИО = 'Макаров Захар Игоревич'

WHERE Студент\_ID = 2;

COMMIT TRAN

Приложение 2. Листинг файла transactions2.sql

---2 транзакция грязное чтение

SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED

BEGIN TRANSACTION

SELECT ФИО

FROM Студент

WHERE Студент\_ID = 1;

COMMIT TRANSACTION

----2 транзакция неповторяемое чтение

SET TRANSACTION ISOLATION LEVEL repeatable read;

BEGIN TRAN

UPDATE Студент

SET ФИО = 'Иванов Иван Иванович'

WHERE Студент\_ID = 1;

COMMIT TRAN;

------2 транзакция фантомные строки

SET TRANSACTION ISOLATION LEVEL serializable

BEGIN TRANSACTION

-- Добавляем новую строку, чтобы создать фантомные строки для повторного чтения в транзакции 1

INSERT INTO Студент (Студент\_ID, ФИО, Сер\_Ном\_Пасп, Гражданство, Адрес, Группа\_ID)

VALUES (1000004, 'Елена Иванова', '1239 567890', 'Российская Федерация', 'Москва, ул. Ленина, д. 1', 1);

COMMIT TRANSACTION;

----2 транзакция Взаимоблокировка

SET TRANSACTION ISOLATION LEVEL read committed;

BEGIN TRAN

UPDATE Студент

SET ФИО = 'Макаров Захар Игоревич'

WHERE Студент\_ID = 2;

WAITFOR DELAY '00:00:05'

UPDATE Студент

SET ФИО = 'Петров Петр Петрович'

WHERE Студент\_ID = 1;

COMMIT TRAN