**Лабораторная работа 2. Транзакции и блокировки. Уровни изолированности транзакций. Распределённые транзакции.**

Предметная область БД, на которой необходимо производить упражнения, может быть любой. Рекомендуется использовать БД, которая была разработана в ходе лабораторного практикума по дисциплине «Управление данными».

⦁ Наглядно продемонстрировать влияние параметра xact\_abort.

--SET XACT\_ABORT OFF

USE PetStore

SET XACT\_ABORT OFF

BEGIN TRANSACTION

UPDATE Заказ

SET [Итоговая стоимость] = 10000

WHERE Id\_Заказа = 1

UPDATE Заказ

SET [Итоговая стоимость] = -100

WHERE Id\_Заказа = 1

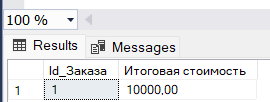
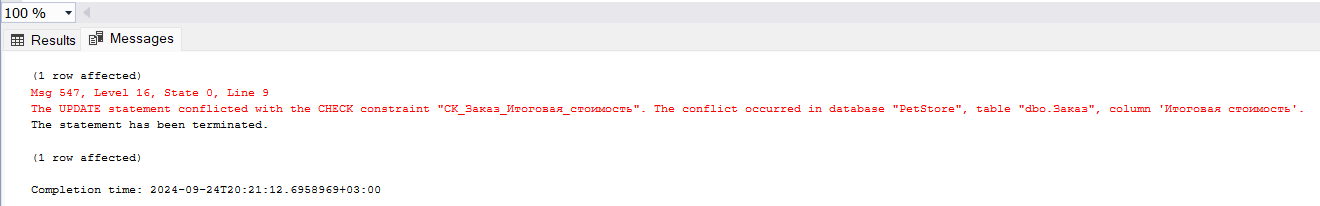
COMMIT

SELECT Id\_Заказа, [Итоговая стоимость]

FROM Заказ

WHERE Id\_Заказа = 1

GO



--SET XACT\_ABORT ON

USE PetStore

SET XACT\_ABORT ON

BEGIN TRANSACTION

UPDATE Заказ

SET [Итоговая стоимость] = 500

WHERE Id\_Заказа = 1

UPDATE Заказ

SET [Итоговая стоимость] = -100

WHERE Id\_Заказа = 1

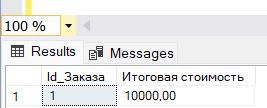
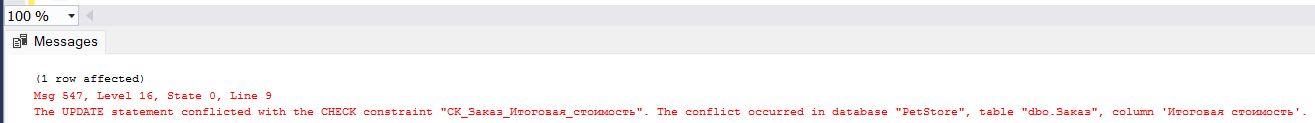
COMMIT

SELECT Id\_Заказа, [Итоговая стоимость]

FROM Заказ

WHERE Id\_Заказа = 1

GO



Автоматический откат транзакции и всех изменений. Инструкция была прервана, и до SELECT не дошло

⦁ Наглядно продемонстрировать явные, неявные и автоподтверждённые транзакции в обоих режимах работы сервера (режим неявного подтверждения и автоматического подтверждения).

Явные в явном:

SET IMPLICIT\_TRANSACTIONS OFF

PRINT @@TRANCOUNT

BEGIN TRANSACTION

PRINT @@TRANCOUNT

UPDATE Заказ

SET [Итоговая стоимость] = 600

WHERE Id\_Заказа = 1

PRINT @@TRANCOUNT

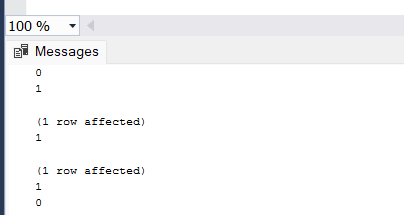
DELETE FROM Заказ

WHERE [Итоговая стоимость] = 600

PRINT @@TRANCOUNT

COMMIT TRANSACTION

PRINT @@TRANCOUNT



Явные в неявном:

SET IMPLICIT\_TRANSACTIONS ON

PRINT @@TRANCOUNT

BEGIN TRANSACTION

PRINT @@TRANCOUNT

UPDATE Заказ

SET [Итоговая стоимость] = 1000

WHERE Id\_Заказа < 5

PRINT @@TRANCOUNT

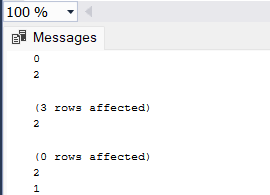
DELETE FROM Заказ

WHERE Id\_Заказа = 1

PRINT @@TRANCOUNT

COMMIT TRANSACTION

PRINT @@TRANCOUNT



Неявные в явном:

SET IMPLICIT\_TRANSACTIONS OFF

PRINT @@TRANCOUNT

UPDATE Заказ

SET [Итоговая стоимость] = 1000

WHERE Id\_Заказа < 5

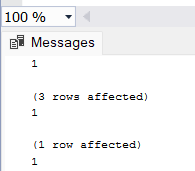
PRINT @@TRANCOUNT

UPDATE Заказ

SET [Итоговая стоимость] = 500

WHERE Id\_Заказа = 5

PRINT @@TRANCOUNT



Неявные в неявном:

SET IMPLICIT\_TRANSACTIONS ON

PRINT @@TRANCOUNT

UPDATE Заказ

SET [Итоговая стоимость] = 1000

WHERE Id\_Заказа < 5

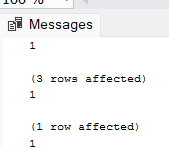
PRINT @@TRANCOUNT

UPDATE Заказ

SET [Итоговая стоимость] = 500

WHERE Id\_Заказа = 5

PRINT @@TRANCOUNT



Явные в авто:

PRINT @@TRANCOUNT

BEGIN TRANSACTION

PRINT @@TRANCOUNT

UPDATE Заказ

SET [Итоговая стоимость] = 1000

WHERE Id\_Заказа < 5

PRINT @@TRANCOUNT

UPDATE Заказ

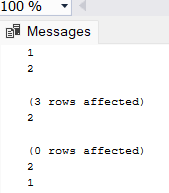
SET [Итоговая стоимость] = 1000

WHERE Id\_Заказа = 1

PRINT @@TRANCOUNT

COMMIT TRANSACTION

PRINT @@TRANCOUNT



Неявные в авто:

PRINT @@TRANCOUNT

UPDATE Заказ

SET [Итоговая стоимость] = 1000

WHERE Id\_Заказа = 1

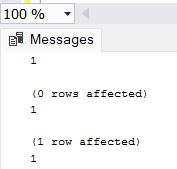
PRINT @@TRANCOUNT

UPDATE Заказ

SET [Итоговая стоимость] = 5000

WHERE Id\_Заказа = 3

PRINT @@TRANCOUNT



⦁ Создать несколько подключений и наглядно продемонстрировать на них разницу между различными уровнями изолированности транзакций (5 уровней).

⦁ READ UNCOMMITTED

Read Uncommitted – самый низкий уровень изоляции. Позволяет читать "грязные" данные незафиксированых транзакций, отсюда и название феномена – «грязное чтение» (Dirty Read). Суть феномена в том, что если первая транзакция запишет какие-то данные, вторая их прочитает, а потом первая транзакция будет отменена, то получится, что вторая транзакция прочитала данные, которые никогда не существовали.

SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED

BEGIN TRANSACTION

UPDATE Заказ

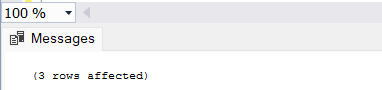
SET [Итоговая стоимость] = 1000

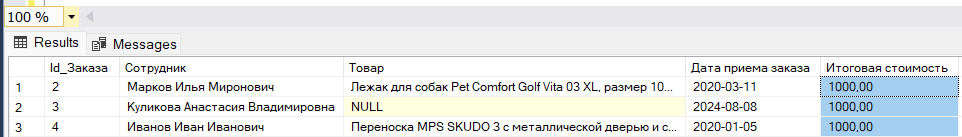
WHERE Id\_Заказа < 5

COMMIT

SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED

BEGIN TRANSACTION





Если в одной транзакции считать данные — другая транзакции не будет ждать окончания первой и обновит данные.

--Первая транзакция

SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED

BEGIN TRANSACTION

SELECT \* FROM Заказ

--Вторая

SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED

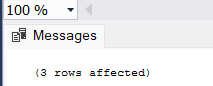
BEGIN TRANSACTION

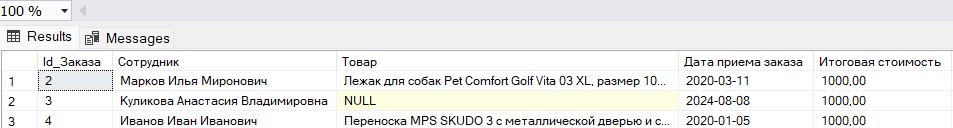
UPDATE Заказ

SET [Итоговая стоимость] = 1000

WHERE Id\_Заказа < 5

COMMIT





READ COMMITTED

Read Committed – при этом уровне изоляции грязное чтение невозможно, то есть второй транзакции не дадут прочитать данные первой до тех пор, пока первая транзакция не зафиксируется. Но при этом уровне изоляции все еще возможна аномалия неповторяющегося чтения. Суть этого феномена в том, что если первая транзакция один раз прочитала данные, а потом вторая их изменила и зафиксировалась, то повторное чтение тех же данных первой транзакцией вернет уже измененные данные.

⦁ Если в одной транзакции поменять данные , то другая транзакция будет ждать окончания первой, чтобы прочитать эти данные.

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN TRANSACTION

UPDATE Заказ

SET [Итоговая стоимость] = 1000

WHERE Id\_Заказа < 5

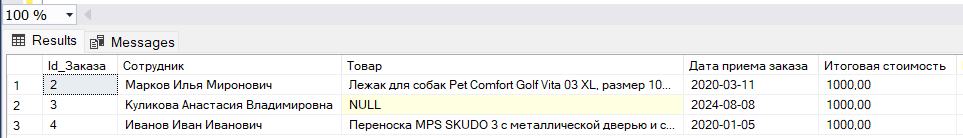
COMMIT

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN TRANSACTION

SELECT \* FROM Заказ

WHERE Id\_Заказа < 5



⦁ Если в одной транзакции считать данные — другая транзакции не будет ждать окончания первой и обновит данные.

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

BEGIN TRANSACTION

SELECT \* FROM Заказ

WHERE Id\_Заказа < 5

SET TRANSACTION ISOLATION LEVEL READ COMMITTED

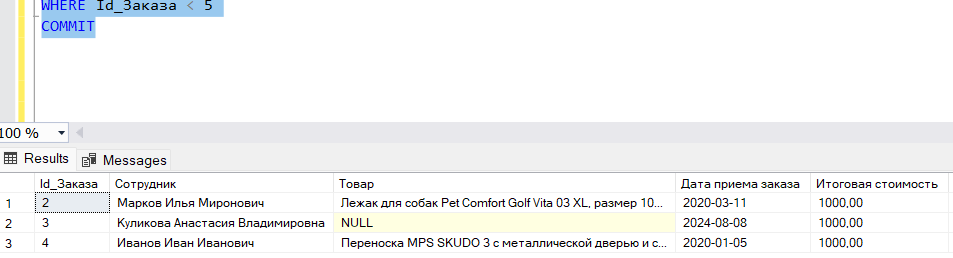
BEGIN TRANSACTION

UPDATE Заказ

SET [Итоговая стоимость] = 1000

WHERE Id\_Заказа < 5

COMMIT



⦁ REPEATABLE READ

Repeatable Read – этот уровень решает предыдущую проблему, но при этом возможно появление фантомов. Изменение однажды прочитанных первой транзакцией данных другими транзакциями (до фиксации первой) невозможно. Однако если первая транзакция сделала выборку по какому-то условию, а потом вторая транзакция добавила новые данные, этому условию удовлетворяющие, и зафиксировалась, то повторная выборка первой транзакцией по тому же условию вернет в том числе и добавленные данные – фантомы.

⦁ Если в одной транзакции считать данные — другая транзакция будет ждать окончания первой, чтобы обновить данные.

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ

BEGIN TRANSACTION

SELECT \* FROM Заказ

WHERE Id\_Заказа < 5

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ

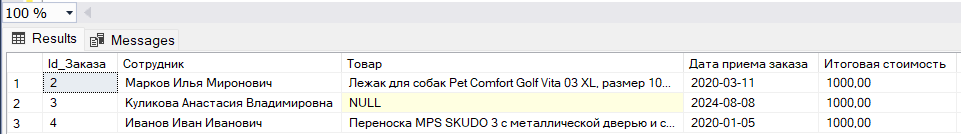
BEGIN TRANSACTION

UPDATE Заказ

SET [Итоговая стоимость] = 100

WHERE Id\_Заказа < 5

COMMIT



⦁ Транзакция не может считывать данные, которые были изменены, но еще не зафиксированы другой транзакцией.

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ

BEGIN TRANSACTION

UPDATE Заказ

SET [Итоговая стоимость] = 100

WHERE Id\_Заказа < 5

COMMIT

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ

BEGIN TRANSACTION

SELECT \* FROM Заказ



⦁ SERIALIZABLE

Serializable – при этом уровне изоляции никакие фантомы невозможны в принципе, равно как и другие феномены, даже такие, которых еще не придумали. Этот уровень изоляции ни на какие феномены не опирается, просто требуется, чтобы результат параллельного выполнения транзакций был таким же, как если бы они выполнялись последовательно.

⦁ Инструкции не могут считывать данные, которые были изменены другими транзакциями, но еще не были зафиксированы.

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE

BEGIN TRANSACTION

UPDATE Заказ

SET [Итоговая стоимость] = 100

WHERE Id\_Заказа < 5

COMMIT

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE

BEGIN TRANSACTION

SELECT \* FROM Заказ

⦁ Другие транзакции не могут изменять данные, считываемые текущей транзакцией, до ее завершения.

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE

BEGIN TRANSACTION

SELECT \* FROM Заказ

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE

BEGIN TRANSACTION

UPDATE Заказ

SET [Итоговая стоимость] = 100

WHERE Id\_Заказа < 5

COMMIT

⦁ SNAPSHOT

SNAPSHOT является отдельным уровнем изоляции. Указывает на то, что данные, считанные любой инструкцией транзакции, будут согласованы на уровне транзакции с версией данных, существовавших в ее начале. Транзакция распознает только те изменения, которые были зафиксированы до ее начала. Инструкции, выполняемые текущей транзакцией, не видят изменений данных, произведенных другими транзакциями после запуска текущей транзакции.

USE PetStore

GO

ALTER DATABASE PetStore SET ALLOW\_SNAPSHOT\_ISOLATION ON

SET TRANSACTION ISOLATION LEVEL SNAPSHOT

BEGIN TRANSACTION

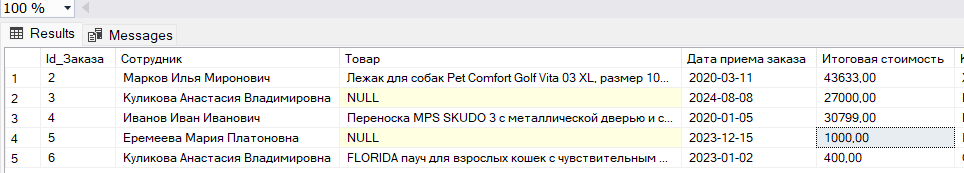
UPDATE Заказ

SET [Итоговая стоимость] = 1000

WHERE Id\_Заказа = 5

SELECT \* FROM Заказ

WHERE Id\_Заказа < 7



SET TRANSACTION ISOLATION LEVEL SNAPSHOT

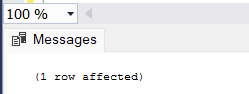
BEGIN TRANSACTION

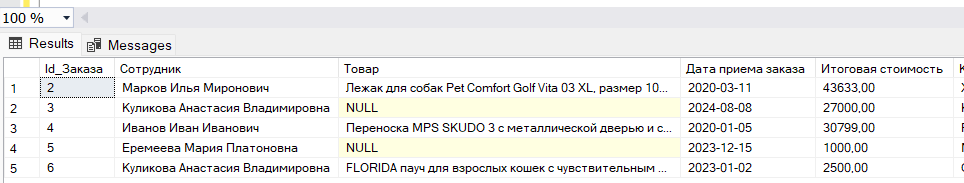
UPDATE Заказ

SET [Итоговая стоимость] = 2500

WHERE Id\_Заказа = 6

COMMIT





⦁ Промоделировать ситуацию взаимоблокировки транзакций. Объяснить полученные результаты.

Взаимоблокировка – это ситуация, когда транзакции блокируют друг друга таким образом, что дальнейшее выполнение невозможно.

BEGIN TRANSACTION

UPDATE Заказ

SET [Итоговая стоимость] = 123456

WHERE Id\_Заказа = 7

WAITFOR DELAY '00:00:05'

SELECT \* FROM Заказ

WHERE Id\_Заказа < 8

COMMIT TRANSACTION

BEGIN TRANSACTION

UPDATE Заказ

SET [Итоговая стоимость] = 3000

WHERE Id\_Заказа = 6

WAITFOR DELAY '00:00:05'

SELECT \* FROM Заказ

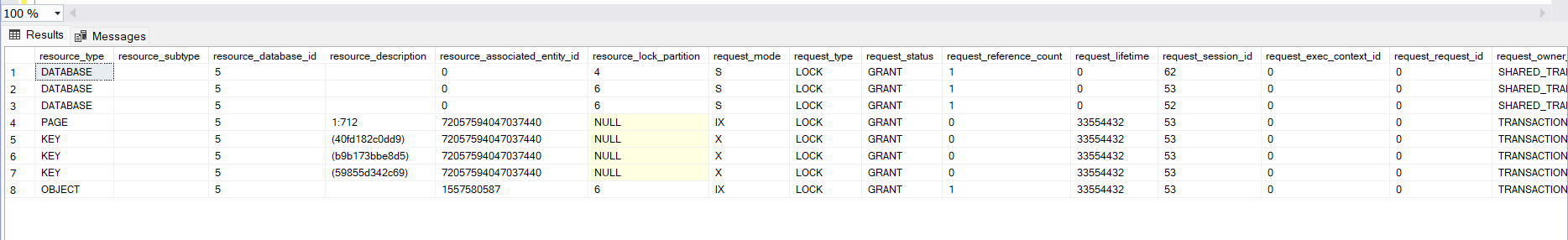
WHERE Id\_Заказа < 8

COMMIT TRANSACTION



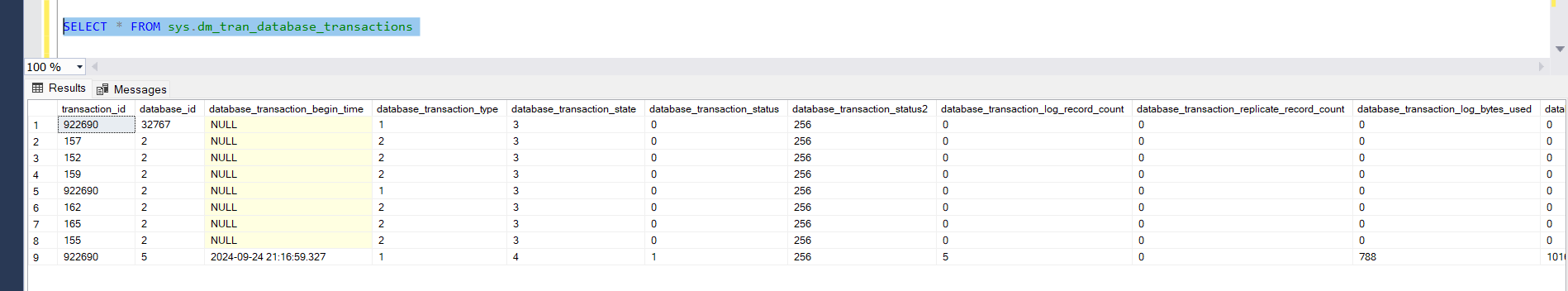
⦁ Наглядно продемонстрировать получение информации о текущих транзакциях и блокировках, исходя из данных системного словаря.

SELECT \* FROM sys.dm\_tran\_locks - Возвращает сведения об активных в данный момент в SQL Server ресурсах диспетчера блокировок:



SELECT \* FROM sys.dm\_tran\_database\_transactions - Возвращает сведения о транзакциях на уровне базы данных:

SELECT \* FROM sys.dm\_tran\_database\_transactions



Вопросы к отчету:

1. Что такое транзакция и для чего она используется?

Транзакция - это набор операций, который должен быть выполнен либо полностью, либо не выполнен вовсе. Она используется для обеспечения целостности данных в базе данных, позволяя совершать несколько операций как единое целое.

2. Какие базовые свойства имеют транзакции?

Основные свойства транзакций - атомарность (все операции либо выполняются полностью, либо не выполняются вовсе), согласованность (транзакция приводит базу данных из одной согласованной состояние в другое), изолированность (другие транзакции не могут изменять данные во время выполнения текущей транзакции), долговечность (изменения, внесенные транзакцией, сохраняются в базе данных).

3. Назовите типы транзакций и режимы подтверждения транзакций сервера.

Типы транзакций: локальные (выполняются только в пределах одной базы данных) и распределенные (включают операции в нескольких базах данных).

Режимы подтверждения транзакций сервера: автоматический (COMMIT или ROLLBACK происходит автоматически), ручной (необходимо явно указать COMMIT или ROLLBACK), явный (когда каждая операция COMMIT/ROLLBACK указывается явно).

4. Расскажите о уровнях изолированности транзакций. Что означает уровень Read uncommitted?

Уровни изолированности транзакций определяют, насколько одна транзакция видит изменения других транзакций. Read uncommitted - уровень, при котором транзакция может видеть изменения, сделанные другими транзакциями, до их фиксации.

5. Что означает уровень Read committed?

Read committed - уровень изолированности, при котором транзакция видит только фиксированные изменения других транзакций. Не видит изменения, которые еще не были зафиксированы.

6. Опишите уровень Repeatable read.

Repeatable read - уровень, при котором транзакция в процессе выполнения не видит изменения данных, сделанные другими транзакциями. Данные, прочитанные транзакцией, останутся неизменными в течение всей транзакции.

7. Что такое уровень Snapshot в изолированности транзакций?

Snapshot - уровень, при котором транзакция видит данные, как если бы она работала с отдельной версией базы данных на момент начала транзакции. Это позволяет избежать блокировок на чтение данных.

8. Что означает уровень Serializable?

Serializable - уровень, при котором транзакция видит только изменения данных, произведенные другими транзакциями после завершения текущей транзакции.

9. Что такое блокировки в контексте транзакций? Виды блокировок и концепция менеджера блокировок.

Блокировки - это механизм, позволяющий контролировать доступ к данным другим транзакциям. Виды блокировок могут быть различными - от простых до сложных, таких как блокировка на чтение, запись, обновление и т.д. Менеджер блокировок отвечает за управление блокировками в базе данных.

10. Что такое взаимоблокировка и приведите примеры.

Взаимоблокировка (или дедлок) - это ситуация, при которой две или более транзакции блокируют друг друга, ожидая освобождения ресурсов. Пример: транзакция A блокирует строку, которую ждет транзакция B, в то время как транзакция B блокирует строку, которую ждет транзакция A.

11. Что такое связанный сервер и распределенный запрос?

Связанный сервер - это сервер, к которому можно установить соединение и выполнять запросы из другого сервера. Распределенный запрос - это запрос, который выполняется на нескольких серверах с помощью связанных серверов.

12. Какова структура файлов журналов? Что такое логический и физический журналы? Понятия LSN, minLSN, контрольной точки.

Структура файлов журналов состоит из логического и физического журналов. Логический журнал содержит информацию об изменениях данных, а физический - информацию об операциях с блоками данных. LSN (Log Sequence Number) - это уникальный номер, присваиваемый каждой записи в журнале. minLSN - минимальный LSN, используемый для восстановления после сбоя. Контрольная точка - это момент времени, на который база данных должна быть восстановлена при сбое.

13.Распределённые транзакции. Двухфазная фиксация. Компонент DTC;

1. Что такое распределённые транзакции?

Распределённые транзакции - это транзакции, которые включают несколько участников или ресурсов, распределенных по различным вычислительным устройствам или системам.

2. Что такое двухфазная фиксация?

Двухфазная фиксация - это протокол, который используется для обеспечения атомарности распределенных транзакций. Он состоит из двух фаз: подготовки и фиксации.

3. Что такое компонент DTC?

DTC (Distributed Transaction Coordinator) - это компонент операционной системы Windows, который отвечает за управление распределёнными транзакциями. Он обеспечивает атомарность, согласованность, изолированность и долговечность (ACID) транзакций, выполняемых через несколько ресурсов или узлов в сети.