**Лабораторная работа 3. Резервное копирование и восстановление**

Задание (выполняется посредством ввода команд на TransactSQL):

⦁ Создать тестовую базу данных. База данных должна состоять из 6 файлов данных и 2 файлов журнала. Файлы данных должны быть распределены между 3 файловыми группами. Установить полную модель восстановления базы данных;

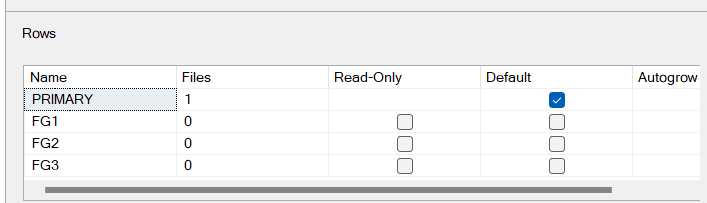
CREATE DATABASE TestDB2

-- Создание файловых групп

ALTER DATABASE TestDB2 ADD FILEGROUP FG1

ALTER DATABASE TestDB2 ADD FILEGROUP FG2

ALTER DATABASE TestDB2 ADD FILEGROUP FG3



-- Добавление файлов в файловые группы

ALTER DATABASE TestDB2 ADD FILE (NAME = 'DataFile1', FILENAME = 'C:\TestBD2\Files\DataFile1.ndf', SIZE = 100MB) TO FILEGROUP FG1

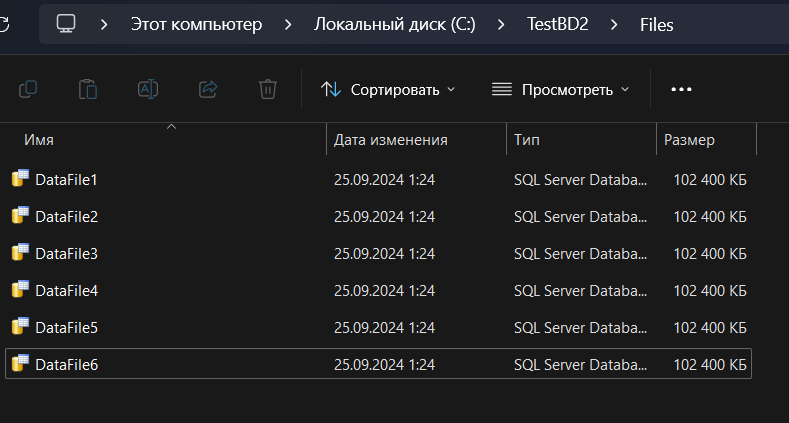
ALTER DATABASE TestDB2 ADD FILE (NAME = 'DataFile2', FILENAME = 'C:\TestBD2\Files\DataFile2.ndf', SIZE = 100MB) TO FILEGROUP FG1

ALTER DATABASE TestDB2 ADD FILE (NAME = 'DataFile3', FILENAME = 'C:\TestBD2\Files\DataFile3.ndf', SIZE = 100MB) TO FILEGROUP FG2

ALTER DATABASE TestDB2 ADD FILE (NAME = 'DataFile4', FILENAME = 'C:\TestBD2\Files\DataFile4.ndf', SIZE = 100MB) TO FILEGROUP FG2

ALTER DATABASE TestDB2 ADD FILE (NAME = 'DataFile5', FILENAME = 'C:\TestBD2\Files\DataFile5.ndf', SIZE = 100MB) TO FILEGROUP FG3

ALTER DATABASE TestDB2 ADD FILE (NAME = 'DataFile6', FILENAME = 'C:\TestBD2\Files\DataFile6.ndf', SIZE = 100MB) TO FILEGROUP FG3



-- Добавление журнальных файлов

USE TestDB2

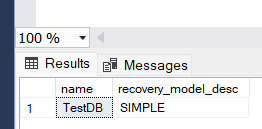
ALTER DATABASE TestDB2 ADD LOG FILE (NAME = 'LogFile1', FILENAME = 'C:\TestBD2\Logs\LogFile1.ldf', SIZE = 50MB)

ALTER DATABASE TestDB2 ADD LOG FILE (NAME = 'LogFile2', FILENAME = 'C:\TestBD2\Logs\LogFile2.ldf', SIZE = 50MB)

SELECT name, recovery\_model\_desc

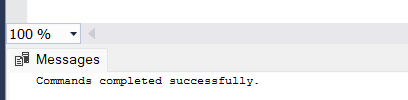
FROM sys.databases

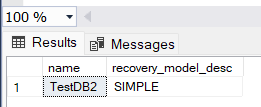
WHERE name = 'TestDB2'



-- Установка полной модели восстановления

ALTER DATABASE TestDB2 SET RECOVERY FULL





⦁ В базе данных создать тестовую таблицу с минимальным количеством столбцов (2-3). Таблица должна быть размещена в файлах только одной из файловых групп. Организовать циклическое заполнение таблицы 200000 записями с произвольными значениями полей.

CREATE TABLE Сотрудники

(

Id INT IDENTITY(1,1) PRIMARY KEY NOT NULL,

Фамилия VARCHAR(50) NOT NULL,

Дата\_рождения DATE NOT NULL,

) ON FG3

GO

INSERT INTO Сотрудники(Фамилия, Дата\_рождения)

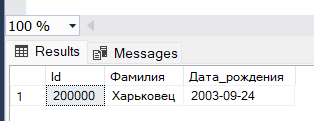
VALUES ('Харьковец', '2003-09-24')

GO 200000

SELECT \*

FROM Сотрудники

WHERE Id = 200000



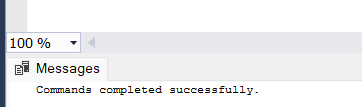
⦁ Создайте логическое устройство копирования на основе файла на жёстком диске; Название логического устройства и местоположения файла определите самостоятельно

USE TestDB2

GO

EXEC sp\_addumpdevice 'disk', 'LogicalDevice2', 'C:\backup\backup2.bak';

GO



⦁ Создайте снимок тестовой базы данных;

CREATE DATABASE TestDB2\_Snapshot

ON

(NAME = TestDB2, FILENAME = 'C:\Snapshots2\TestDB2.ss' ),

( NAME = 'DataFile1', FILENAME = 'C:\Snapshots2\DataFile1.ss' ),

( NAME = 'DataFile2', FILENAME = 'C:\Snapshots2\DataFile2.ss' ),

( NAME = 'DataFile3', FILENAME = 'C:\Snapshots2\DataFile3.ss' ),

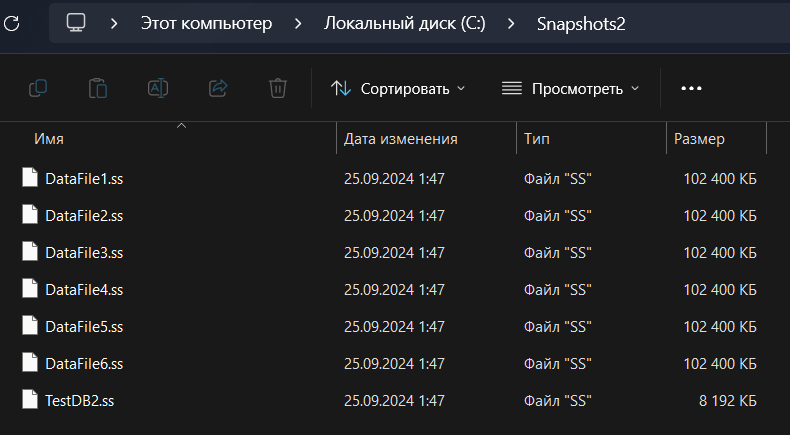
( NAME = 'DataFile4', FILENAME = 'C:\Snapshots2\DataFile4.ss' ),

( NAME = 'DataFile5', FILENAME = 'C:\Snapshots2\DataFile5.ss' ),

( NAME = 'DataFile6', FILENAME = 'C:\Snapshots2\DataFile6.ss' )

AS SNAPSHOT OF TestDB2;





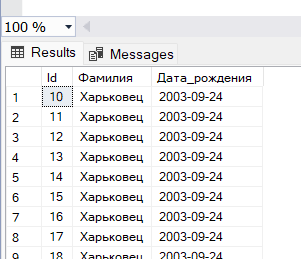
⦁ Смоделируйте порчу данных в тестовой таблице и восстановите БД из снимка; Убедитесь в успешности восстановления БД;

USE TestDB2

DELETE Сотрудники

WHERE Id < 10

SELECT \* FROM Сотрудники



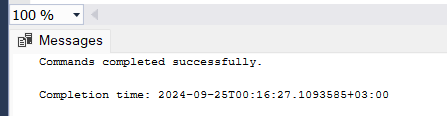
USE master

GO

RESTORE DATABASE TestDB2

FROM DATABASE\_SNAPSHOT = 'TestDB2\_Snapshot'

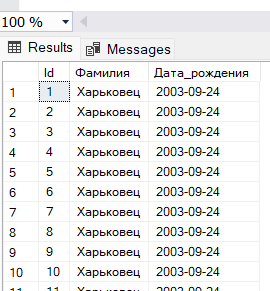
GO



USE TestDB2

GO

SELECT \* FROM Сотрудники



⦁ Создайте полную резервную копию тестовой БД в созданное логическое устройство копирования;

USE TestDB2

GO

EXEC sp\_addumpdevice 'disk', 'LogicalDevice2', 'C:\backup\backup2.bak';

GO

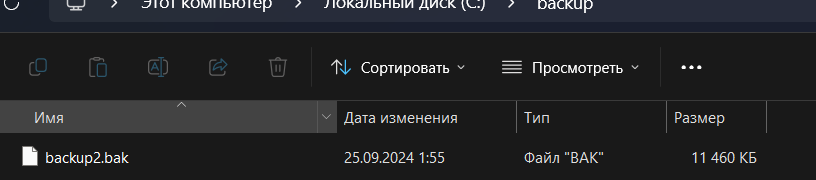
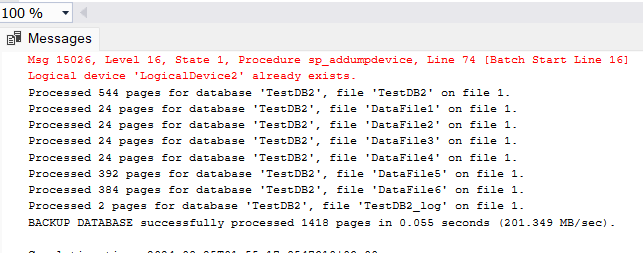
USE master

GO

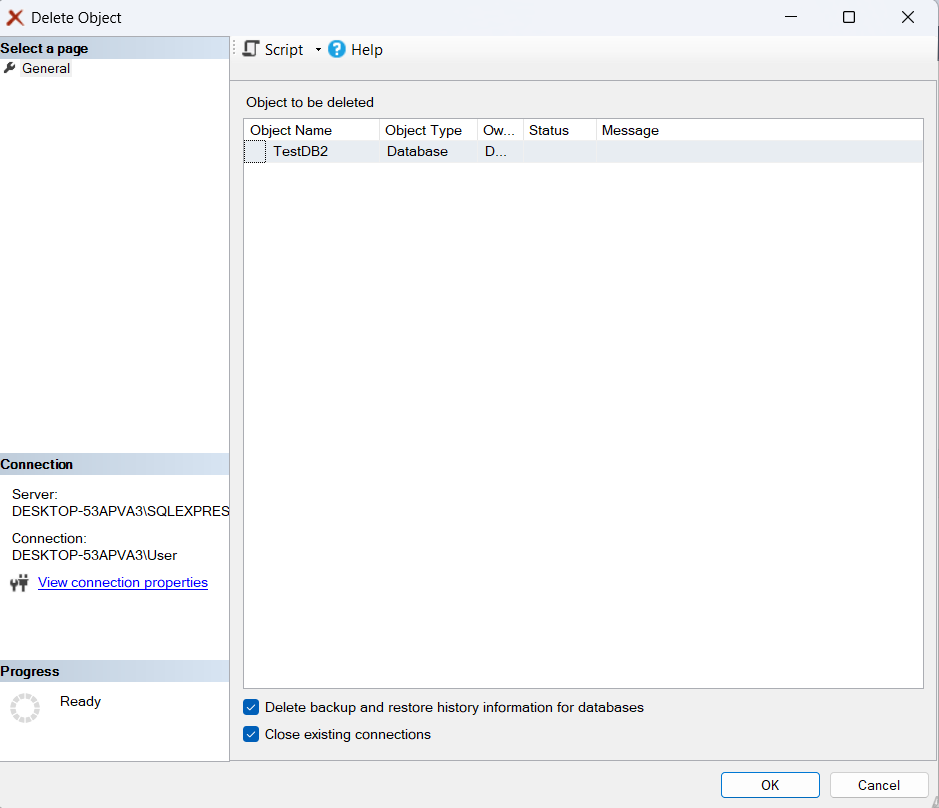
BACKUP DATABASE TestDB2

TO LogicalDevice2

GO



⦁ Смоделируйте потерю тестовой БД и запустите процесс восстановления из созданной полной резервной копии; Убедитесь в успешности восстановления БД;



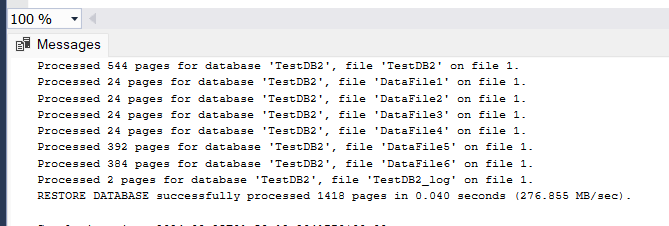
USE master

GO

RESTORE DATABASE TestDB2

FROM LogicalDevice2

GO



⦁ Внесите изменения в БД, добавив и изменив несколько строк (100-200). Создайте разностную резервную копию БД в созданном ранее логическом устройстве;

USE TestDB2

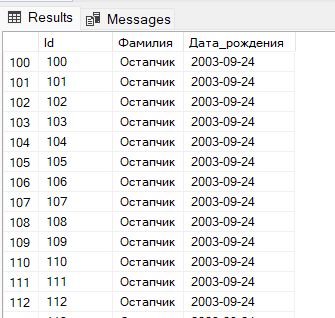
GO

UPDATE Сотрудники

SET Фамилия = 'Остапчик'

WHERE Id < 200

GO



USE TestDB2

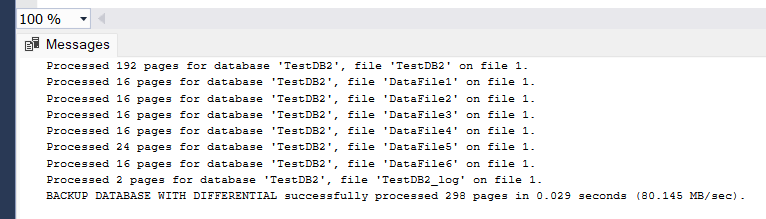
GO

BACKUP DATABASE TestDB2

TO DISK = 'LogicalDevice2'

WITH DIFFERENTIAL;

GO



⦁ Смоделируйте потерю тестовой БД и запустите процесс восстановления из созданной полной резервной копии и разностной копии; Убедитесь в успешности восстановления БД;

--Полное восстановление

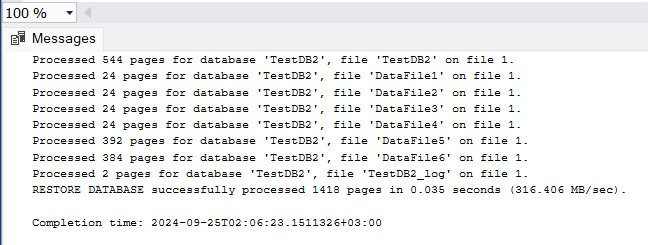
USE MASTER

GO

RESTORE DATABASE TestDB2

FROM DISK = 'C:\backup\backup2.bak'

GO



--Изменение данных

USE TestDB2

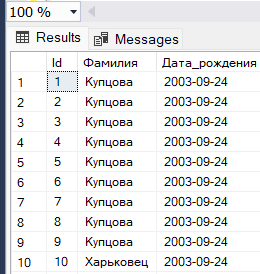
GO

UPDATE Сотрудники

SET Фамилия = 'Купцова'

WHERE ID < 10

GO



--Востановление данных с помощью разностной копии

USE master

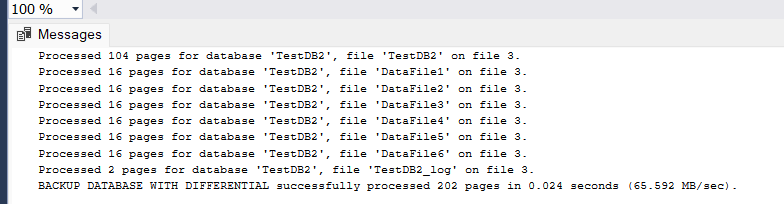
GO

BACKUP DATABASE TestDB2

TO LogicalDevice2

WITH DIFFERENTIAL

GO



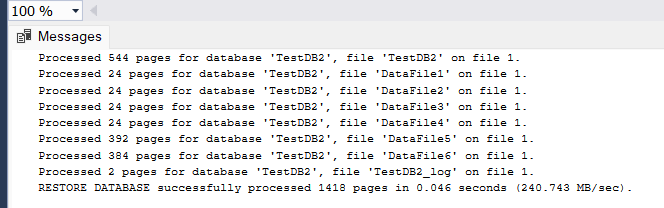
USE MASTER

GO

RESTORE DATABASE TestDB2

FROM DISK = 'C:\backup\backup2.bak'

GO



⦁ Убедитесь, что транзакции все зафиксированы. Создайте 2 резервных копий журнала транзакций тестовой БД на созданное ранее логическое устройство. Между двумя процессами резервного копирования выполните и зафиксируйте несколько простых транзакций, причём первую из них снабдите пометкой (with mark).

USE TestDB2

GO

BEGIN TRANSACTION ДобавитьСотрудника

WITH MARK 'Добавить сотрудника';

GO

INSERT INTO Сотрудники (Фамилия, Дата\_рождения)

VALUES ('Мосин', '1987-05-13')

GO

COMMIT TRANSACTION ДобавитьСотрудника

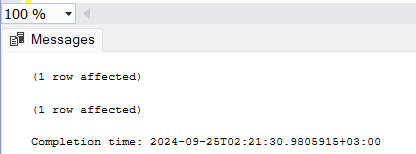
GO

BEGIN TRAN

INSERT INTO Сотрудники (Фамилия, Дата\_рождения) VALUES ('Купцова', '1999-12-31')

COMMIT TRAN

GO



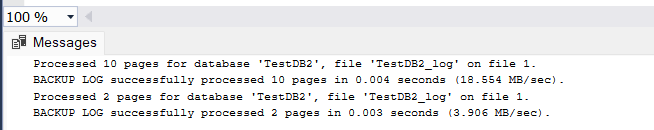
USE master

GO

BACKUP LOG TestDB2 TO DISK = 'C:\backup\TransactionLogBackup1.trn'

BACKUP LOG TestDB2 TO DISK = 'C:\backup\TransactionLogBackup2.trn'

GO



USE master

go

RESTORE DATABASE TestDB2 FROM DISK = 'C:\backup\backup2.bak' WITH REPLACE;

--from LogicalDevice2

--with norecovery

RESTORE DATABASE TestDB2 FROM DISK = 'C:\backup\backup2.bak' WITH REPLACE;

--from LogicalDevice2

--with file = 2, norecovery

Restore log TestDB2

RESTORE DATABASE TestDB2 FROM DISK = 'C:\backup\backup2.bak' WITH REPLACE;

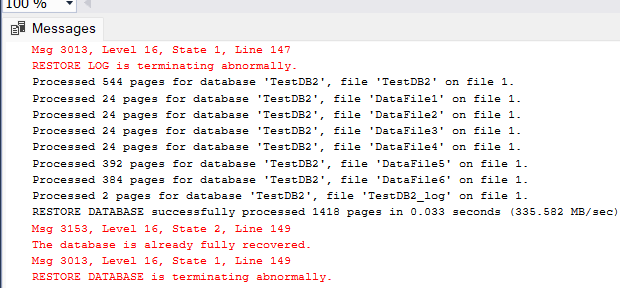
Restore log TestDB2

RESTORE DATABASE TestDB2 FROM DISK = 'C:\backup\backup2.bak' WITH REPLACE;

Restore database TestDB2

with recovery

GO



⦁ Смоделируйте потерю тестовой БД и запустите процесс восстановления данных на момент сбоя. Убедитесь в успешности восстановления БД;

USE MASTER;

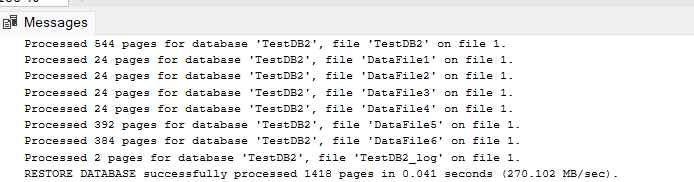
GO

RESTORE DATABASE TestDB2

FROM DISK = 'C:\backup\backup2.bak'

WITH REPLACE

GO



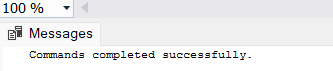


⦁ Смоделируйте потерю тестовой БД и запустите процесс восстановления данных на момент выполнения транзакции, снабжённой меткой. Убедитесь в успешности восстановления БД;

BACKUP DATABASE TestDB2

TO DISK = 'C:\backup\backup2.bak'

DROP DATABASE TestDB2



USE TestDB2

GO

BEGIN TRANSACTION ДобавитьСотрудника

WITH MARK 'Добавить сотрудника';

GO

INSERT INTO Сотрудники (Фамилия, Дата\_рождения)

VALUES ('Мосина', '1987-05-13')

GO

COMMIT TRANSACTION ДобавитьСотрудника

GO

BEGIN TRANSACTION

INSERT INTO Сотрудники (Фамилия, Дата\_рождения) VALUES ('Валуев', '2000-10-23')

COMMIT TRANSACTION

GO

use master

go

Restore database TestDB2

from disk = 'C:\backup\backup2.bak'

with norecovery;

Restore database TestDB2

from disk = 'C:\backup\backup2.bak'

with norecovery;

Restore log TestDB2

from disk = 'C:\backup\TransactionLogBackup1.trn'

with norecovery;

restore log TestDB2

from disk = 'C:\backup\TransactionLogBackup2.trn'

with norecovery, stopatmark ='Добавить сотрудника';

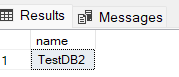
restore database TestDB2

with recovery;

Select name from sys.databases

where name = 'TestDB2'

GO



№1. Полное резервное копирование.

Включает в себя файлы данных и журнал транзакций. По сути является базой данных на момент создания резервной копии базы данный MS SQL Server.

Включает в себя:

⦁ Резервное копирование данных в базе.

⦁ Резервное копирование изменений, возникающих во время резервного копирования

⦁ Резервное копирование транзакций, не зафиксированных в журнале транзакций.

№2. Разностная резервная копия.

Включает в себя все изменения базы данных с момента последнего полного резервного копирования.

Нельзя восстановить без полной резервной копии. После каждого запуска разностного копирования, размер резервной копии возрастает из-за количества транзакций с момента полного резервного копирования.

При создании разностного резервного копирования выполняются следующие действия:

Создание резервных копий баз данных, которые изменились с момента полного резервного копирования.

Создание резервных копий всех операций, выполняющихся во время разностного резервного копирования и всех транзакций не зафиксированных в журнале транзакций.

№3. Резервная копия файла или файловой группы.

Используется для снятия резервных копий определенных файлов или файловых групп.

По сути являются именованными коллекциями файлов и используются для упрощения размещения данных и выполнения задач администрирования.

Файлы журналов не входят в состав файловых групп.

Управление пространством журнала отделено от управления пространством данных, возможно только полное и разностное резервное копирование файлов и файловых групп.

№4. Резервная копия файла журнала транзакций.

Содержат все изменения базы данных при первичном резервном копировании лога транзакций, или изменения с последней успешной резервной копии журнала транзакций.

Не имеет смысла, если хоть раз не выполнялось полное резервное копирование, т.к. резервную копию лога невозможно будет восстановить при отсутствии полной резервной копии.

В процессе выполняются следующие действия:

Создается копия журнала транзакций от последнего резервного копирования лога до конца текущего.

Очищаются части журнала транзакций до начала активной части и отбрасываются сведения в неактивной части.

№5. Модель восстановления БД: полная.

Все изменения в базе полностью журналируются. Записи остаются в журнале транзакций до тех пор, пока не будет сделана резервная копия журнала. При выполнении резервирования журнала транзакций для базы, которая находится в режиме полного восстановления, записи журнала записываются в бэкап журнала транзакций, и записи завершенных транзакций удаляются из журнала транзакций.

Это не означает, что каждое изменение имеет отдельную запись в журнале, поскольку некоторые операции пишутся с меньшим количеством записей в журнале, но тем не менее журналируется полный эффект от операции.

Журнал транзакций не будет очищаться (т.е. его части не станут доступными для переиспользования) до тех пор, как не будет сделана резервная копия журнала транзакций.

Все опции восстановления доступны, когда база данных в полной модели восстановления (и была в ней с момента создания последней резервной копии).

№6. Модель восстановления БД: с неполным протоколированием.

Некоторые изменения (такие как перестроение индекса или пакетная загрузка, но НЕ стандартные INSERT/UPDATE/DELETE) могут минимально журналироваться, что снижает количество записей в журнале и журнал транзакций не становится слишком большим за время выполнения этих операций. Обратите внимание, что это не изменяет размер последующих резервных копий журнала транзакций.

Журнал транзакций не будет очищаться (т.е. его части не станут доступными для переиспользования) до тех пор, как не будет сделана резервная копия журнала транзакций (абсолютно так же, как и при полной модели восстановления).

Используя модель восстановления с неполным протоколированием, вы теряете часть опций по восстановлению (восстановление на момент времени и резервная копия заключительного фрагмента) в обмен на повышение производительности, связанное с минимально журналируемыми операциями.

№7. Модели восстановления БД: простая

Некоторые изменения могут минимально журналироваться (абсолютно так же, как и при модели восстановления с неполным протоколированием).

Журнал транзакций не будет очищаться до выполнения операции создания контрольной точки (CHECKPOINT) — обычно она выполняется автоматически.

Создание резервных копий журнала транзакций невозможно, поэтому у вас остается самое ограниченное число опций по восстановлению.

№8. Общие сведения об операциях, допускающих неполное протоколирование.

Модель восстановления с неполным протоколированием минимизирует использование пространства журнала транзакций при операциях с неполным протоколированием, подобных BULK INSERT, SELECT INTO или CREATE INDEX.

Модель восстановления с неполным протоколированием улучшает производительность операций с загрузкой больших объемов данных посредством сокращения количества журнализируемой информации.

Если база данных повреждается во время выполнения операции с минимальным протоколированием, базу данных можно восстановить только к последнему бэкапу журнала транзакций, созданному до первой операции с минимальным протоколированием.

№9. Восстановление БД из снимка.

Возврат не предназначен для восстановления носителей. Моментальный снимок базы данных — это неполная копия файлов базы данных, поэтому в случае, если база данных или ее снимок будут повреждены, вероятно, возврат к снимку станет невозможным.

Исходная база данных-источник перезаписывается базой данных, к которой выполняется возврат, при этом любые изменения, внесенные в базу данных после создания моментального снимка, будут утеряны.

Операция возврата также перезаписывает старый файл журнала и перестраивает журнал.

При возврате разрывается цепочка резервных копий журналов. Следовательно, прежде чем можно будет выполнять резервное копирование журналов возвращенной базы данных, необходимо создать полную резервную копию базы данных или файлов.

Во время операции возврата моментальный снимок и база данных-источник будут недоступны.

Метаданные возвращенной базы данных совпадают с метаданными на момент снимка.

Операция восстановления удаляет все полнотекстовые каталоги.

№10. Восстановление БД из полной копии.

Полное резервное копирование делает копию всей базы данных, включая все объекты и данные системных таблиц. Полная резервная копия не будет усекать (truncate) журнал транзакций. Это основной тип резервных копий, который требуется выполнять перед другими типами резервных копий.

Полную резервную копию вы можете восстановить за 1 шаг, так как она не требует других дифференциальных/инкрементальных копий.

Если модель восстановления базы SQL данных установлена как “Полная”, то при восстановлении бекапа вы можете указать параметр “STOPAT”, где указывается время (до секунды) на котором нужно остановить восстановление данных.

№11. Восстановление БД из полной копии и разностных копий.

Дифференциальное или разностное резервное копирование — это копирование только тех данных, которые появились с момента последней полной резервной копии.

Данный тип резервного копирования используют совместно с полной резервной копией, так как для восстановления дифференциальной копии необходима полная резервная копия.

Обычно при использовании разностного резервного копирования используют план по типу “полное раз в N дней, дифференциальное каждые N часов”. Если ежедневный оборот данных достаточно высокий, то данный тип резервных копий может быть неудобен в применении, так как копии будут весить довольно много.

Например, если полная резервная копия весит 300 GB, а дифференциальная спустя час работы 5 GB, то спустя сутки это будет 120 GB, что делает использование данного типа копий нерациональным.

№12. Восстановление БД на определенный момент времени.

Если в модели восстановления с неполным протоколированием резервная копия журнала содержит изменения с неполным протоколированием, то в пределах этой резервной копии восстановление до момента времени невозможно. База данных должна быть восстановлена к концу резервной копии журнала транзакций.

Восстановление на момент времени всегда производится из резервной копии журналов. В каждой инструкции RESTORE LOG последовательности восстановления необходимо указать целевое время или транзакцию в идентичном предложении STOPAT. В качестве предварительного условия для восстановления на момент времени необходимо сначала восстановить полную резервную копию базы данных, чья конечная точка предшествует моменту времени восстановления. Эта полная резервная копия базы данных может быть старше самой последней полной резервной копии базы данных, поскольку затем восстанавливается каждая последующая резервная копия журналов, вплоть до резервной копии журналов, содержащей целевой момент времени.

№13. Политика.

Компонент резервного копирования и восстановления SQL Server обеспечивает необходимую защиту важных данных, которые хранятся в базах данных SQL Server .

Создание резервных копий баз данных SQL Server , выполнение проверочных процедур восстановления резервных копий и хранение резервных копий в безопасном месте вне рабочей площадки помогают предотвратить возможную необратимую потерю данных. Резервное копирование — единственный способ защитить данные.

При правильном создании резервных копий баз данных можно будет восстановить данные после многих видов сбоев, включая следующие:

⦁ сбой носителя;

⦁ ошибки пользователей (например, удаление таблицы по ошибке);

⦁ сбои оборудования (например, поврежденный дисковый накопитель или безвозвратная потеря данных на сервере);

⦁ стихийные бедствия. При выполнении резервного копирования SQL Server в службу хранилища BLOB-объектов Azure можно создать резервную копию в регионе, отличном от региона своего фактического локального расположения, на случай возникновения природных катастроф в своем регионе.

Кроме того, резервные копии баз данных полезны и при выполнении повседневных административных задач, например для копирования баз данных с одного сервера на другой, настройки Группы доступности AlwaysOn или зеркального отображения баз данных и архивирования.