SQLskills Immersion Event
IE0: Accidental/Junior DBA

Module 4: Backup and Restore

Erin Stellato
Erin@SQLskills.com

Introduction

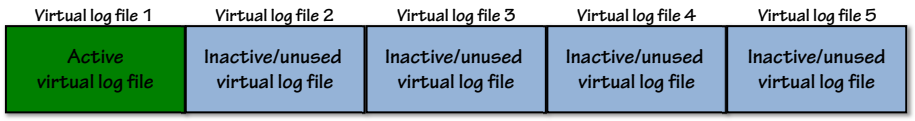
- One of the most important tasks a DBA has to perform is a database backup
- This is a very easy task to automate
- You would be surprised at how often we hear about systems where backups are old, or simply *don't exist*
- Secondary to backups, but no less important, are restores
- How do you know if you have a successful backup if you've never restored it?
- How do you know how long it will take to restore a database if you've never tried it?
- The worst time to test your restore methodology is when there's a problem and your manager is standing right next to you
- This is why we say, "Plan a recovery strategy, not a backup strategy"

Overview

- The transaction log and database recovery models
- Backup fundamentals
- Restore fundamentals

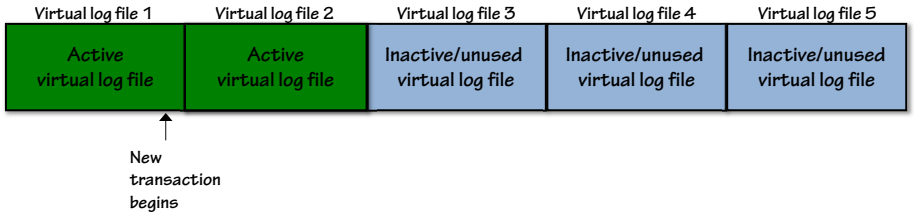
Transaction Log

- The transaction log is a physical file, divided up into chunks called virtual log files, or VLFs for short
  - VLFs are logically defined within the log file
- A VLF is either wholly active or inactive
  - There is always at least one active VLF
- Newly created VLFs are inactive and unused



Transaction Log

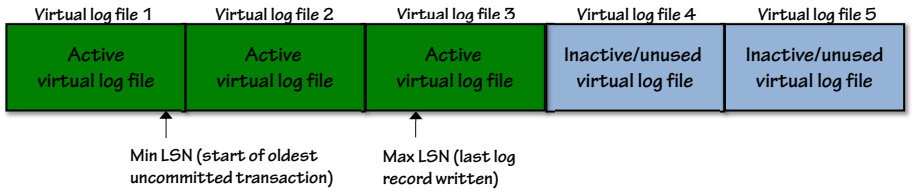
- When a transaction modifies the database, a log record is written to the transaction log
- The first log record written to a VLF makes the VLF active



SQLskills immerse yourself in sql server

Moving Through the Transaction Log

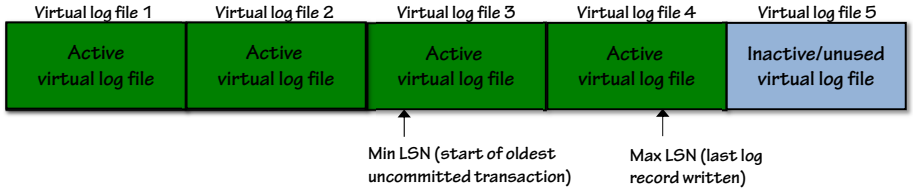
- As more log records are written, more VLFs become active
- SQL Server tracks:
  - Which portion of the transaction log is still required
  - The start of the oldest uncommitted transaction
  - The most recent log record written
- The oldest uncommitted transaction defines the oldest active LSN and therefore the oldest active VLF



SQLskills immerse yourself in sql server

Transaction Log Clearing

- **A VLF can be made inactive once all log records are not required**
  - This is called clearing (or truncating) the transaction log
- **Log clearing is done by a log backup in the FULL or BULK_LOGGED recovery models, or by a checkpoint in SIMPLE recovery model**
  - There are no exceptions to this
- **Transaction log backup or checkpoint occurs…**

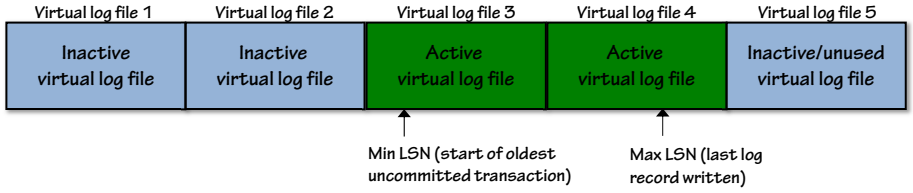


SQLskills immerse yourself in sql server

7

Transaction Log Clearing

- **All that happens is that zero or more VLFs are marked inactive**
  - Nothing is zeroed, removed, overwritten ("clearing" is a misnomer)
  - The transaction log file size does not change ("truncating" is a misnomer)

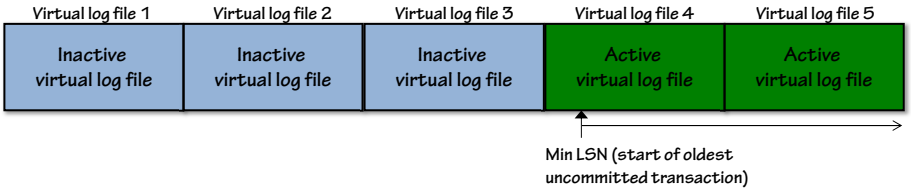


SQLskills immerse yourself in sql server

8

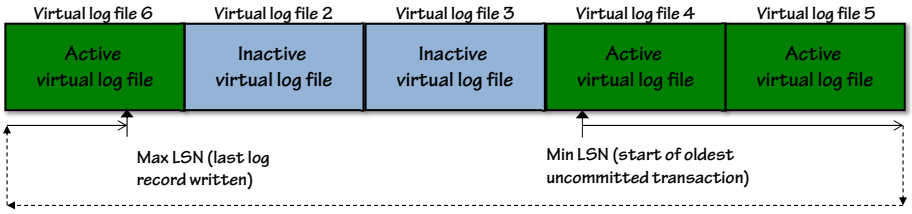
Circular Nature of the Transaction Log

- Once the end of the transaction log is reached, it wants to wrap around and start using the first VLF again, as long as log clearing has occurred, without having to grow the transaction log file
- The log manager checks the active status of the first VLF in the transaction log



Circular Nature of the Transaction Log (2)

- If the first VLF is inactive, the transaction log can wrap
  - Note that VLF 1 is reactivated and becomes VLF 6
- Log records continue to be written
- See example in Paul's post *Inside the Storage Engine: More on the circular nature of the log*
  - https://www.sqlskills.com/ie0/tlogcircular
- If log clearing does not occur, the transaction log will grow forever



Demo

Examining VLF usage

What is Recovery Model?

- **Database property that has 3 options:**
  - FULL
  - BULK_LOGGED
  - SIMPLE
- **Determines:**
  - How some transactions are logged
  - If backups of the transaction log are required (and even allowed)
  - The type of restore operations available
- **To determine the best recovery model for a database, you need to know the SLA for the corresponding application**
  - Service Level Agreements (SLAs) are determined by the business
- **Each recovery model has pros and cons that you need to understand, not just to help you decide the best model, but also because it affects what *you* have to do as the DBA**

Downtime SLA

- **Maximum allowable downtime or RTO**
  - Recovery Time Objective
- **Commonly discussed in terms of 'number of nines'**
  - 5-nines = 99.999% uptime
    - Slightly over 5 minutes downtime per year
  - 4-nines = 99.99% uptime
    - Almost 52.5 minutes downtime per year
  - 3-nines = 99.9% uptime
    - Almost 8.75 hours downtime per year
- **Must consider how downtime is defined for you**
  - Is it 24x7 or, maybe just 9am-5pm weekdays
- **Achieving 5-nines of 24x7 operation is very hard**
- **Management may ask for zero downtime!**
  - Not technically possible
- **You have to know how technologies introduce downtime**
  - e.g. failover time with an Availability Group

Data Loss SLA

- **Maximum allowable data loss or RPO**
  - Recovery Point Objective
- **Must consider how data loss is defined for you**
  - Number of transactions
  - Work done in a period of time
- **May be different for different tables or databases**
- **Zero data loss is easier to achieve compared to 5-nines uptime**
- **Press the business owners for a proper answer**
- **Management may ask for zero data loss too!**
  - This IS achievable, but is it absolutely necessary?

FULL Recovery Model

- **Everything is fully logged**
  - Log truncation usually not possible until log backup
  - Operations like creating or rebuilding an index create as much log as the size of the index being created/rebuilt
- **Requires transaction log backups to prevent the log file from continually growing**
- **Allows for point-in-time recovery**
- **Potentially zero data loss in the event of a failure**

BULK_LOGGED Recovery Model

- **Minimal logging for a few things, full logging otherwise**
  - Log truncation usually not possible until log backup
  - The complete list of minimally-logged operations is in the Books Online topic *The Transaction Log (SQL Server)*
    - https://bit.ly/35iaUKP
- **Requires transaction log backups to prevent the log file from continually growing**
  - Log backups will NOT be smaller than when in FULL!
- **Data loss is possible**
- **Used for more advanced scenarios**
- **For ease of recovery, do not use unless you're very familiar with its implications**

SIMPLE Recovery Model

- **Same logging as for BULK_LOGGED**
  - Log is cleared/truncated on checkpoint
- **Transaction log backups are not possible**
- **If you have a problem and need to restore a backup, you can only restore to the last full backup**
- **Potential for significant data loss, depending on full backup frequency**

Recovery Models and Log Backups

- **FULL recovery model**
  - All changes are fully logged therefore log backups will support:
    - Up-to-the-minute recovery by backing up the tail-of-the-log
    - Point-in-time recovery (requires STOPAT option and either a specific time or marked transaction)
- **BULK_LOGGED recovery model**
  - Bulk operations are minimally logged
    - If the tail-of-the-log contains a minimally-logged operation and the data files are unavailable, the tail-of-the-log backup will succeed but will corrupt the database when restored
    - NO point-in-time recovery possible for the time covered by a log backup containing a minimally-logged operation
- **SIMPLE recovery model does not support log backups**

Switching Recovery Models

- When switching to FULL for the first time, the database behaves as if it in the SIMPLE recovery model until the first full backup is performed
  - This is called being in 'pseudo-SIMPLE'
  - The transaction log will clear whenever a checkpoint occurs
- The number one cause of transaction log files growing out of control is that a database has been running in pseudo-SIMPLE and then someone takes a backup, really switching the database into FULL
  - The log will grow forever unless log backups are also being taken, or until it runs out of disk space
- Switching between FULL and BULK_LOGGED does NOT break the log backup chain, and does not affect log shipping
  - This is NOT possible with Database Mirroring or Availability Groups
- Switching to SIMPLE breaks the log backup chain

How Does Recovery Model Affect DBA Tasks?

- Each recovery model has pros and cons that you need to understand, not just to help you decide the best model, but also because it affects what *you* have to do as the DBA
- Regardless of the recovery model, you need to schedule full backups
- For FULL and BULK_LOGGED, transaction log backups are required
- Only FULL allows point-in-time recovery

	SIMPLE	BULK_LOGGED	FULL
Full backups needed?	YES	YES	YES
Transaction log backups needed?	NO	YES	YES
Point-in-time recovery?	NO	NO*	YES

- *Can be possible if no minimally-logged operations have occurred

Demo

Changing recovery model

Overview

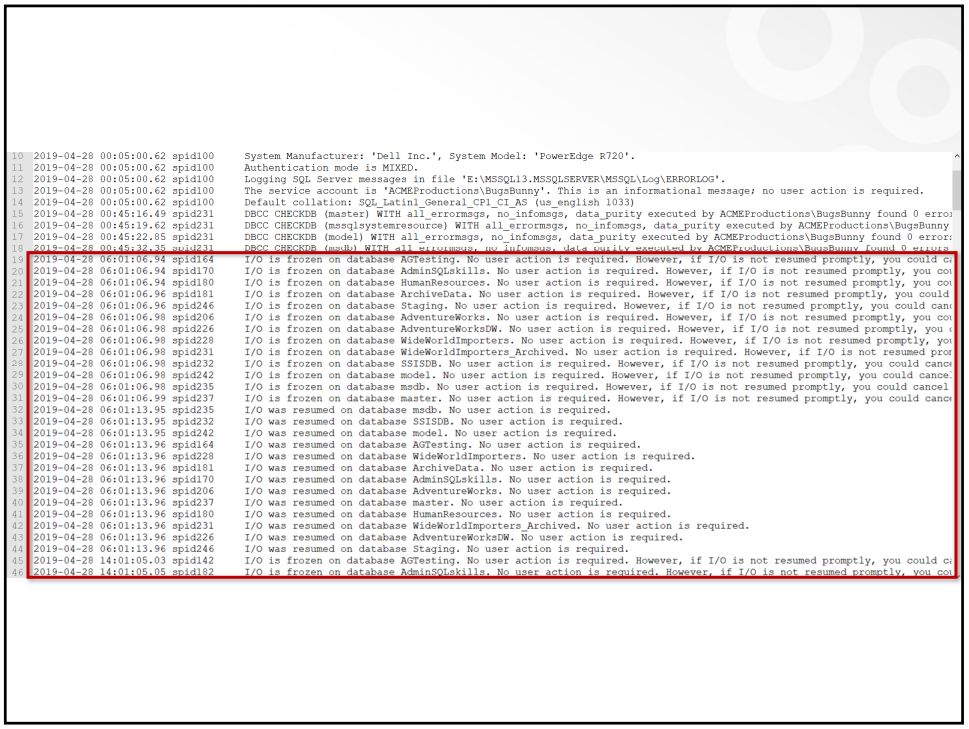
- The transaction log and database recovery models
- Backup fundamentals
- Restore fundamentals

Why Take Backups?

- **To allow a restore to take place**
  - Disaster recovery
    - DR is not just for hardware failure, it can also be for recovering data that was accidentally changed or deleted
  - Log shipping
  - Initializing Database Mirroring or Availability Groups
  - Initializing Replication
  - Transferring a database to another environment (e.g. development, test)
  - Moving/upgrading a database
- **To manage the size of the transaction log**
- **Just in case…**
  - Before performing a database repair
  - Before performing an upgrade of any kind
    - Hardware, OS, SQL Server

How to Backup

- **Native backups to disk**
- **Native backups to cloud**
  - Azure
  - Amazon
  - Other
- **VSS**
  - Volume Shadow Copy Service
  - Framework that contains a set of functions used by third-party applications to perform backups
- **VDI**
  - APIs to hook into SQL Server backups
  - e.g. LiteSpeed, Redgate SQL Backup



```
2019-04-28 00:05:00.62 spid100      System Manufacturer: 'Dell Inc.', System Model: 'PowerEdge R720'.
2019-04-28 00:05:00.62 spid100      Authentication mode is MIXED.
2019-04-28 00:05:00.62 spid100      Logging SQL Server messages in file 'E:\MSSQL13.MSSQLSERVER\MSSQL\Log\ERRORLOG'.
2019-04-28 00:05:00.62 spid100      The service account is 'ACMEProductions\BugsBunny'. This is an informational message; no user action is required.
2019-04-28 00:05:00.62 spid100      Default collation: SQL_Latin1_General_CP1_CI_AS (us_english 1033)
2019-04-28 00:45:16.49 spid231      DBCC CHECKDB (master) WITH all_errormsgs, no_infomsgs, data_purity executed by ACMEProductions\BugsBunny found 0 erro
2019-04-28 00:45:19.62 spid231      DBCC CHECKDB (mssqlsystemresource) WITH all_errormsgs, no_infomsgs, data_purity executed by ACMEProductions\BugsBunny
2019-04-28 00:45:22.85 spid231      DBCC CHECKDB (model) WITH all_errormsgs, no_infomsgs, data_purity executed by ACMEProductions\BugsBunny found 0 error
2019-04-28 00:45:32.35 spid231      DBCC CHECKDB (msdb) WITH all_errormsgs, no_infomsgs, data_purity executed by ACMEProductions\BugsBunny found 0 errors
2019-04-28 06:01:06.94 spid164      I/O is frozen on database AGTesting. No user action is required. However, if I/O is not resumed promptly, you could ca
2019-04-28 06:01:06.94 spid170      I/O is frozen on database AdminSQLskills. No user action is required. However, if I/O is not resumed promptly, you cou
2019-04-28 06:01:06.94 spid180      I/O is frozen on database HumanResources. No user action is required. However, if I/O is not resumed promptly, you cou
2019-04-28 06:01:06.96 spid181      I/O is frozen on database ArchiveData. No user action is required. However, if I/O is not resumed promptly, you could
2019-04-28 06:01:06.96 spid246      I/O is frozen on database Staging. No user action is required. However, if I/O is not resumed promptly, you could can
2019-04-28 06:01:06.98 spid206      I/O is frozen on database AdventureWorks. No user action is required. However, if I/O is not resumed promptly, you cou
2019-04-28 06:01:06.98 spid226      I/O is frozen on database AdventureWorksDW. No user action is required. However, if I/O is not resumed promptly, you c
2019-04-28 06:01:06.98 spid228      I/O is frozen on database WideWorldImporters. No user action is required. However, if I/O is not resumed promptly, you
2019-04-28 06:01:06.98 spid231      I/O is frozen on database WideWorldImporters_Archived. No user action is required. However, if I/O is not resumed pro
2019-04-28 06:01:06.98 spid232      I/O is frozen on database SSISDB. No user action is required. However, if I/O is not resumed promptly, you could canc
2019-04-28 06:01:06.98 spid242      I/O is frozen on database model. No user action is required. However, if I/O is not resumed promptly, you could cance
2019-04-28 06:01:06.98 spid235      I/O is frozen on database msdb. No user action is required. However, if I/O is not resumed promptly, you could cancel
2019-04-28 06:01:06.99 spid237      I/O is frozen on database master. No user action is required. However, if I/O is not resumed promptly, you could canc
2019-04-28 06:01:13.95 spid235      I/O was resumed on database msdb. No user action is required.
2019-04-28 06:01:13.95 spid232      I/O was resumed on database SSISDB. No user action is required.
2019-04-28 06:01:13.95 spid242      I/O was resumed on database model. No user action is required.
2019-04-28 06:01:13.96 spid164      I/O was resumed on database AGTesting. No user action is required.
2019-04-28 06:01:13.96 spid228      I/O was resumed on database WideWorldImporters. No user action is required.
2019-04-28 06:01:13.96 spid181      I/O was resumed on database ArchiveData. No user action is required.
2019-04-28 06:01:13.96 spid170      I/O was resumed on database AdminSQLskills. No user action is required.
2019-04-28 06:01:13.96 spid206      I/O was resumed on database AdventureWorks. No user action is required.
2019-04-28 06:01:13.96 spid237      I/O was resumed on database master. No user action is required.
2019-04-28 06:01:13.96 spid180      I/O was resumed on database HumanResources. No user action is required.
2019-04-28 06:01:13.96 spid231      I/O was resumed on database WideWorldImporters_Archived. No user action is required.
2019-04-28 06:01:13.96 spid226      I/O was resumed on database AdventureWorksDW. No user action is required.
2019-04-28 06:01:13.96 spid246      I/O was resumed on database Staging. No user action is required.
2019-04-28 14:01:05.03 spid142      I/O is frozen on database AGTesting. No user action is required. However, if I/O is not resumed promptly, you could ca
2019-04-28 14:01:05.05 spid182      I/O is frozen on database AdminSQLskills. No user action is required. However, if I/O is not resumed promptly, you cou
```

Recommended Backup Options

- **WITH CHECKSUM**
  - Tests all page checksums in the database
  - Calculates checksum over entire backup and puts it into the backup header
  - With this option, you can use RESTORE … WITH VERIFYONLY, CHECKSUM
    - Does everything except actually restoring the database
    - Read all the pages in the backup and checks any page checksums
    - Re-calculates the backup stream checksum and checks against the one in the header
- **WITH COMPRESSION**
  - Backup compression sometimes pre-allocates space for the backup for performance, which can lead to backup failures when the destination drive does not have enough space
  - Trace flag 3042 prevents this behavior (2008+, see KB 2001026)
- **Either setting can be specified for a backup, or enabled for all backups by default (instance settings covered in Module 2)**

Native Backups

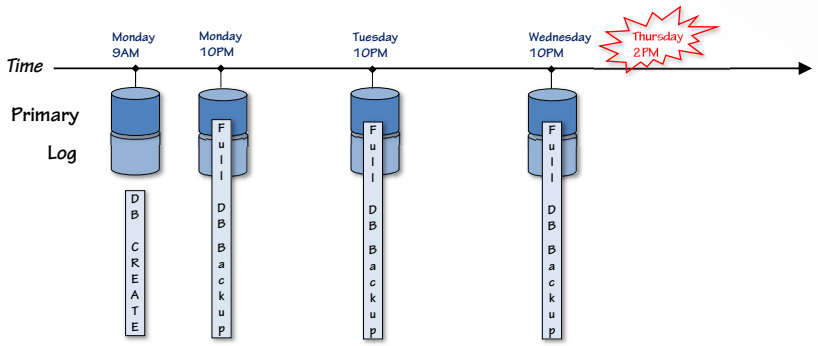
- **Manually**
  - Use appropriate BACKUP statement for type of backup performing
- **SQL Server Management Studio (SSMS)**
  - Generate a one-time backup
  - Create scheduled backups
  - Create a script from which to build a backup
- **Database Maintenance Plan Wizard**
- **SQL Server Management Objects (SMO)**
- **Ola Hallengren's scripts**
- **Create backup devices to simplify naming**
  - Local: do not use UNC names!
  - Network: use FULLY QUALIFIED UNC names

Full Database Backups

- **Creates image for a possible recovery starting point**
- **Complete image of all data, etc. (less free space)**
- **Data is not changed, it's a straight copy**
- **Data is read sequentially**
- **Does not require a backup 'window'**
- **Usually NOT used alone as other backups must be used to roll-forward to a later point in time**
  - Exception: SIMPLE recovery model
- **Backup image is backup completion, meaning that if restored, the image will be the committed changes up to the time that the backup finished**

Full Database Backup Only Strategy

- Full backups only provide a single point in time to which the database can be restored
- If there is a failure, all work since the last full backup is lost



Transaction Log Backups

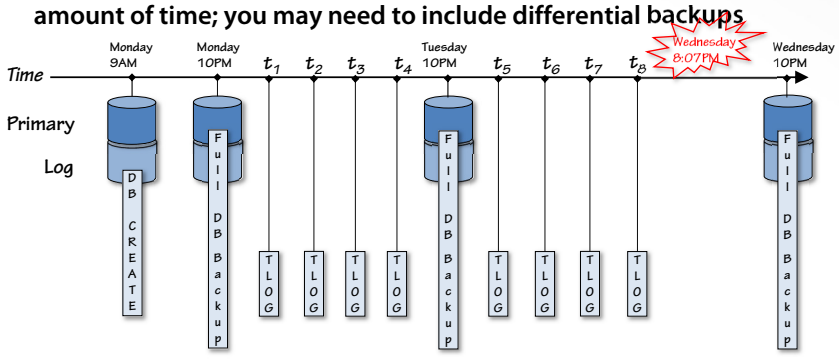
- Information about the changes since the last transaction log backup (can be described as incremental)
- Taken at frequent intervals therefore usually fairly small
- Once backed up, SQL Server clears the inactive portion (the inactive VLFs) of the log, if possible
  - This can help to keep the transaction log file small and manageable because the file wraps around
  - Remember, SQL Server does NOT release the file's free space
- Key purposes
  - Allow roll-forward to point of failure
  - Allow recovery to an arbitrary point in time

Full Database Plus Log Backup Strategy

- **Relatively common configuration**
  - Easy to setup
  - Easy to control
  - Easy to recover
- **Costly to manage**
  - Takes more time
  - Requires more space than just full backups

Full Database Backup Plus Log Backups Strategy

- **Up-to-point-of-crash recovery is possible with no data loss**
- **Possible to work around a damaged full backup if prior log backups are still available**
- **Restoring a large number of log backups can take a significant amount of time; you may need to include differential backups**

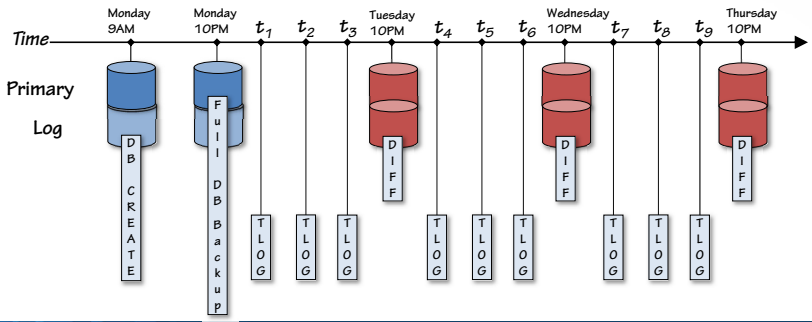


Differential Database Backups

- Similar to full backup except only includes the extents modified since the last full backup
  - i.e. differential backups are NOT incremental, they are cumulative
- Time to create the backup is proportional to the number of extents modified since last full backup.
- Excellent to reduce roll-forward time on restore
- Faster than restoring the comparable transaction log backups for same timeframe
- How big will the next differential backup be?
  - https://www.sqlskills.com/ie0/dbchangesincefull

Full, Diff Database, and Log Backup Strategy

- Differential database backups are only the extents changed since the last full database backup
- Simplifies recovery time by allowing you to recover the database, the last differential, and only the logs after that, until the time of the disaster



Full Database Backup Concerns

- **Large database**
  - Time: reading a lot of data takes a lot of time
  - Space: wasting space on backup media with data that has not changed
    - Solution: partitioning or backup compression
  - Cost: in terms of time, space, and media storage
- **Log backups do not clear the log during full (or diff) backups**
  - Log can fill and/or cause significant auto-growth
  - Log needs to be as large as space required to hold active transactions during full backup
    - Longer the full, larger the log
  - Log clearing is deferred until the data backup completes
  - **Note:** You can still take transaction log backups during a backup, but the log will not clear until the backup finishes

The Log Backup Chain

- **The log backup chain is a *continuous* sequence of log backups**
- **If the log backup chain is not complete (meaning one or more log backups are damaged or missing) then complete recovery is not possible**
  - Must have complete chain of log backups from which to restore
- **Consider using mirrored backups of the transaction log and/or storing them on redundant disks**
- **What breaks the log chain?**
  - Clearing the transaction log manually (using WITH TRUNCATE_ONLY or WITH NO_LOG in SQL Server 2005 or earlier)
  - Changing the database to SIMPLE recovery model (and then changing back)
  - Deleting, overwriting, or throwing away a log backup that was not made as COPY_ONLY
  - Reverting from a database snapshot

Demo

Taking full and transaction log backups

Tail-Of-The-Log Backups
Up-To-The-Point-Of-Crash Recovery

- This is the best case!
- If the transaction log is accessible and the data portion of the database is damaged (in part or entirely) then you might be able to back up the transaction log
- If you can backup the transaction log *and* you have all of the transaction log backups since the last full *and* you're in FULL recovery model (or BULK_LOGGED but no minimally logged operations have occurred) *then* up-to-the-minute recovery is possible and with NO DATA LOSS!

How to Backup the Tail of the Log?

- **Use the NO_TRUNCATE clause with the BACKUP LOG command**
  - BACKUP LOG dbname TO device1 WITH NO_TRUNCATE
- **Through SSMS (not recommended)**
  - Right-click on a working database, choose Tasks, Backup Database… , select the damaged database from drop-down
  - Choose the Options 'page' on backup dialog
  - Under the Transaction Log section
    - Truncate the transaction log by removing inactive entries
    - SELECT THIS ONE >>> Backup the tail of the log, and leave the database in a restoring state
      - i.e. BACKUP WITH NO_TRUNCATE, NORECOVERY
- **Important: If you want to keep your database 'partially available' you should use the Transact-SQL syntax instead**
  - You do NOT want your database put into the restoring state

Demo

Tail-of-the-log backups

Overview

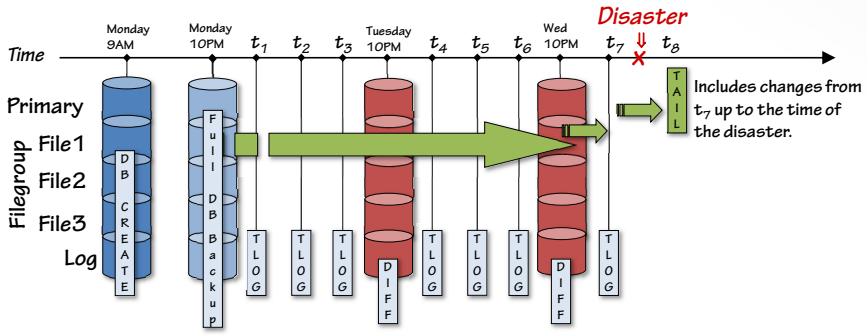
- The transaction log and database recovery models
- Backup fundamentals
- Restore fundamentals

How to Restore

- In an emergency, always go to the operations guide/DR handbook/run book
  - If you don't have one, it is recommended to create one
- Operations guide should define:
  - Location of backups
  - How to put together the restore sequence, depending on what is damaged
  - Commands or scripts to use
  - Amount of time expected per step during restore process
- Recommended to NOT use SSMS when recovering from disaster: you want to see what's happening
- Make sure to use the correct RESTORE options
- Always want to restore the fewest backups possible to reduce downtime
- This is one of the most important things to practice as a DBA

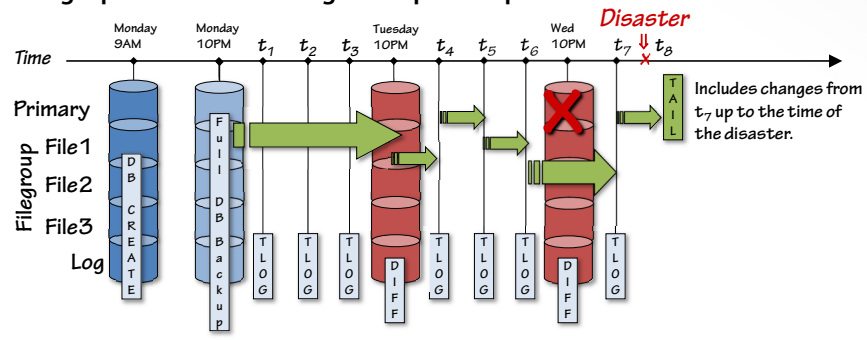
Recovery with Full, Diff, Log

- Full backup on Monday, differential backups on Tuesday and Wednesday
- In the event of failure at time x on Thursday…
  - Backup "tail" of log
  - Restore Monday full, Wednesday differential, and all log backups up to the point of the failure



Falling Back on Log Backups

- What if the differential backup from Wednesday was bad?
- Restore Monday's full, latest working differential (Tuesday) and all log backups up to the point of the failure (t4 through t7) and finally the tail of the log (t8)
- Begs question of how long to keep backups?



Recovery Completion State: RECOVERY

- `RESTORE … WITH RECOVERY`
  - This is the default – catches a lot of people out
- **Using WITH RECOVERY leaves the database operational and *no* additional backups (differential or log) can be restored**
- **Active transactions at the time the backup completed are rolled back**
  - i.e. the UNDO portion of recovery is run, which generates log records to affect the rolling back of active transactions
  - As the database has generated new transaction log records, no additional backups can be restored
- **The restored database is left in a transactionally consistent state and the restore sequence is completed**

Recovery Completion State: NORECOVERY

- `RESTORE … WITH NORECOVERY`
- **Using WITH NORECOVERY does *not* leave the database operational and additional backups (differential or log) *can* be restored**
- **Active transactions at the time the backup completed are not rolled back**
- **The database remains inaccessible and in the RESTORING state until recovery is manually completed or another backup is restored**
- **The REDO part of recovery will have been performed**
  - Slightly confusing, but REDO is done on each backup that is restored as it does not affect whether more backups can be restored

Recovery Completion State
Important Points

- RECOVERY is the default
- Use NORECOVERY for safety and make it *your default* for all restore operations in the restore sequence
  - If you make a mistake, you're back to square one
  - Required when setting up database mirroring or log shipping on the initial restore of the full database backup
- Use RESTORE DATABASE … WITH RECOVERY to complete the restore sequence and bring the database online

Restoring an Entire Database to a Point in Time

- Common example: restore a database to the state it was in before a user or application error occurred (e.g. dropping a table)
- Start with most recent full backup and proceed with smallest set of backups to the desired point
- Best practice is to use WITH STOPAT on all operations in the restore sequence
  - Date and time specific
  - Transaction name (i.e. marked transaction)
  - LSN
- Note: All work after the stop point is lost

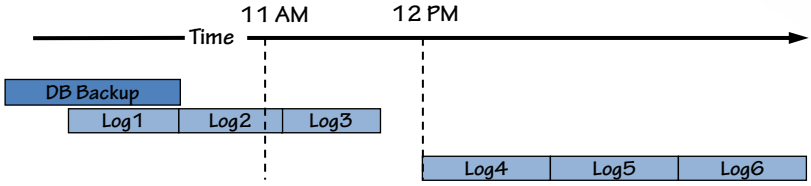
Demo

Restore from backup, recovery options, and point-in-time restore

Continuity of the Log Chain in P.I.T.

- Updating a database after point-in-time restore creates a new recovery path
- Follow point-in-time restore with a new full backup to avoid having to point-in-time restore again if another disaster occurs

At 12 noon, the database is restored and recovered to 11:00 AM



- The log backups taken after the database was recovered are only useful if the exact same recovery path is followed
  - Log backup #3 ends the first recovery path

Restore Sequence: Media Failure

- **Backup the tail of the log**
- **Identify damaged devices (i.e. what needs to be restored?)**
- **Verify backups (location and existence)**
- **Provision undamaged location to restore to**
- **Decide the SMALLEST restore possible: page, file, filegroup, multiple filegroups, database…**
- **Restore (all WITH NORECOVERY):**
  - Full database backup
  - Optionally, restore differentials
  - All required log backups
  - Tail-of-the-log backup
- **Finish recovering the database**
  - RESTORE DATABASE … WITH RECOVERY

Restore Sequence: Human Error

- **Investigate nature of problem: do you know what the error was, and when it occurred?**
  - Excellent if you do know, not so good if you don't
  - Are you auditing? Can you look in audit logs? Default trace?
  - Can you restore to other location and attempt to find when data changed?
    - PAINFULLY time consuming
    - RESTORE … WITH STANDBY and slowly query the database
  - Third-party products can help (but beware, not infallible!)
- **Possibly take damaged component offline**
- **Choose a restore location**
  - In place or alternate location
- **Begin restore sequence as for media failure**
- **Possibly merge in damaged data to existing database**

Restore in Place

- All application dependent information is already on the server (i.e. the application ecosystem)
- Replaces current database
- Must replace damaged devices first or use WITH MOVE to restore on stable media
- Restore based on backup strategy
- No further steps necessary as everything required is already there

- Easiest restore location by far!

Restoring to Alternate Location

- Same instance, different name – relatively easy!
  - Consider name changes to jobs and SQLIS packages, and anything that connects directly
  - Client connectivity issues
- Different instance on same server (includes all of the above)
  - Different logins
  - Different msdb (Agent Jobs/SSIS packages)
  - Different service accounts (mail profiles)
  - Different master (user-defined SPs do not exist)
  - Different servername (client connectivity)
- Physically different server (includes all of the above)
  - Any external dependencies? (batch files, email…)
  - Local Windows authentication accounts may not exist
- KB 246133 on migrating logins between servers

Restoring to Non-Enterprise

- SQL Server 2005 will not allow a database that contains table/index partitioning to restore on any Edition except Enterprise and Developer
- SQL Server 2008+ expands this list to also include:
  - Change data capture
  - Data compression
  - Transparent data encryption
- All of these require elevated privileges except data compression, which only requires ALTER TABLE
- This can be disastrous in a disaster recovery situation as the database must be restored BEFORE the server can tell whether any of these are present
- Use sys.dm_db_persisted_sku_features to check

Restoring to Non-Enterprise in SQL Server 2016 SP1

- Select features that were previously Enterprise-only are available in Standard Edition of SQL Server 2016 SP1 and higher, for example:
  - Columnstore
  - Partitioning
  - Compression
  - In-Memory OLTP
- There are limitations with these features in Standard Edition
- It *may* be possible to restore a database previously running on Enterprise to Standard Edition
- Best practice: stay with the original edition

Data Encryption

- **When restoring a database that contains encrypted data there are multiple possibilities:**
  - Same server/instance – no troubles
  - Different instance or different server – different service master key
    - Must open up the database master key and RE-encrypt the data using the new instance's service master key – EASY if you know to do it and you have the password with which the original database master key was created
- **Especially important in SQL Server 2008+ with Transparent Data Encryption**
  - Encrypted database cannot be restored without the encryption key

VLDB Concerns when Restoring

- **Restoring whole database can take a lot of time**
- **Media failure**
  - If isolated corruption can restore the file or filegroup from a full backup
  - Must go up-to-the-point-of-the-crash and only if piecemeal restore is an option
- **User error**
  - Best choice is a secondary server that's not quite up to date, e.g. log shipping with a load delay!
- **VLDB backup and restore is very challenging**
  - Do as much as possible to speed things up
  - Compression, striping, partitioning, granular backups

Restoring System Databases

- **System databases can only restore from backups of the same SP level as the server (for example, an RTM backup of msdb will not restore on SP1)**
- **master**
  - Requires the instance to be in single-user mode, restore using WITH REPLACE
- **model**
  - Restore in same way as for user databases
- **msdb**
  - Restore in same way as for user databases
  - Stop SQL Server Agent before restoring
  - Make sure to set the recovery model to FULL if msdb was rebuilt
- **resourcedb**
  - Cannot be backed up or restored
  - Use file-system copy operations but be careful not to overwrite resourcedb with an older version
- **tempdb**
  - Cannot be backed up or restored

Key Takeaways

- **Backups are one of the most essential tasks you must perform as a DBA**
- **Recovery model affects whether you need to also take transaction log backups**
  - Understand the difference between FULL and SIMPLE and how that affects the transaction log and recovery
- **Develop a restore strategy (not a backup strategy) based on RTO and RPO for the database(s)**
- **Restore your backups on a regular basis, and practice disaster recovery scenarios**

Resources (1)

- **Posts:**
  - SQLskills Accidental DBA Series
    - https://www.sqlskills.com/help/accidental-dba/
  - Kimberly's Backup and Restore Category
    - https://www.sqlskills.com/ie0/backuprestorekt
  - Paul's Backup and Restore Category
    - https://www.sqlskills.com/ie0/backuprestorepr
  - A SQL Server DBA myth a day: (30/30) backup myths
    - https://www.sqlskills.com/ie0/backupmyths
  - Disaster recovery 101: backing up the tail of the log
    - https://www.sqlskills.com/ie0/backuptaillog

SQLskills immerse yourself in sql server

Resources (2)

- **Articles:**
  - TechNet Magazine : Understanding SQL Server Backups
    - http://technet.microsoft.com/en-us/magazine/dd822915.aspx
  - TechNet Magazine: Recovering from Disasters using Backups
    - http://technet.microsoft.com/en-us/magazine/ee677581.aspx
  - SQL Server Magazine: Advanced Backup/Restore Options
    - https://www.itprotoday.com/sql-server/advanced-backup-and-restore-options
  - INF: Space requirements for backup devices in SQL Server
    - http://support.microsoft.com/kb/2001026
  - SQL Server Central: Backup Monitoring and Reporting
    - http://www.sqlservercentral.com/articles/Backup+%2f+Restore/66564/

SQLskills immerse yourself in sql server

Resources (3)

- **Articles:**
  - How to transfer logins and passwords between instance of SQL Server
    - http://support.microsoft.com/kb/246133
- **Pluralsight:**
  - SQL Server: Logging, Recovery, and the Transaction Log
    - https://www.pluralsight.com/courses/sqlserver-logging

SQLskills immerse yourself in sql server