**What is a database file?**

When you create a database, two primary files are created by default: the data file and the transaction log file. The primary purpose of the data file is to hold all the data, such tables, indexes, store procedures and other relevant data. While the data file is simple to understand and requires some management, the transaction log file requires a greater attention and understanding.

**What are a transaction log file and its purpose?**

The primary function of the transaction log file is to:

Record all changes to the database

Record changes sequentially

All data is written to the transaction log file first before committing the changes to the data file

A triggering mechanism, called a checkpoint, is triggered each few minutes to the transaction log to indicate that a particular transaction has been completely written from the buffer to the data file; this process keeps flushes out the committed transaction, and maintains the size of the physical transaction log file (only in simple recovery mode)

Key object needed to restore databases

Controls the size of the transaction log file and prevents the log consuming the disk space

Used in log shipping, database mirroring, and replication

Allows recover to a point in time

**Reason the transaction log file is out of control in size**

Transaction log backups are not occurring while in Simple recovery mode

Very long transactions are occurring, like indexes of table or many updates

Demo to control the size of transaction log by doing log backups

**Stop**

**What are the recovery models and their roles?**

The recovery models in SQL Server, Simple, Full, Bulk-Logged, determine whether a transaction log backup is available or not

**With Simple recovery model**

Transaction log backups are not supported

Truncation of the transaction log is done automatically, thereby releasing space to the system

You can lose data, as there are no transaction log backups

When in recovery mode, data in the T-Log will not grow

**With Bulk-logged recovery model**

Supports transaction log backups

As in Full mode, there is no automated process of transaction log truncation

Used primarily for bulk operations, such as bulk insert, thereby minimal

**Full recovery model**

Supports transaction log backups

Little chance of data loss under the normal circumstances

No automated process to truncate the log thus must have T-log backups

The transaction log backups must be made regularly to mark unused space available for overwriting

When using Full recovery mode, the T-log can get large in size

During most backup processes, the changes contained in the logfile are sent to the backup file

Scripts for Recovery Models:

select [name], DATABASEPROPERTYEX([name],'recovery')  
from sysdatabases  
where name not in ('master','model','tempdb','msdb')

--Change the recovery mode:

USE master;

GO

-- Set recovery model to SIMPLE

ALTER DATABASE Admin SET RECOVERY SIMPLE;

GO

 -- Set recovery model to FULL

ALTER DATABASE Admin SET RECOVERY FULL;

GO

**SCRIPTS TO CHECK THE TRANSACTION LOG**

--Demo that taking a full backup does not truncate the log file, but taking a transaction log file does truncate the log file

--drop the current database

EXEC msdb.dbo.sp\_delete\_database\_backuphistory @database\_name = N'ADMIN'

GO

USE [master]

GO

ALTER DATABASE [ADMIN] SET SINGLE\_USER WITH ROLLBACK IMMEDIATE

GO

USE [master]

DROP DATABASE [ADMIN]

GO

--Create a database for testing

USE MASTER

GO

CREATE DATABASE ADMIN

GO

-- Create a table and insert data from AdventureWorks2012.HumanResources.Employee

USE ADMIN

GO

SELECT \*

INTO dbo.AAA

FROM AdventureWorks2012.HumanResources.Employee

SELECT \* FROM AAA

--Change the recovery mode to full so as to take transactional log backups

USE MASTER;

ALTER DATABASE ADMIN

SET RECOVERY FULL;

USE ADMIN

GO

-- View the space used and allocated to transaction log

DBCC SQLPERF (LOGSPACE);

--Database Name Log Size (MB) Log Space Used (%) Status

--ADMIN 0.5078125 74.71154 0

--Take a full backup of database

BACKUP DATABASE admin

TO DISK = 'C:\FullBackups\admin.bak';

-- Modify the database by updates, and deletes

USE Admin

GO

UPDATE AAA

SET MaritalStatus = 'S'

WHERE JobTitle = 'Design Engineer';

DELETE AAA

WHERE BusinessEntityID > 5;

--take a full database backup to set it in full mode

BACKUP DATABASE admin

TO DISK = 'C:\FullBackups\admin.bak';

--check the space used by log file after full database backup, notice the log space used had not reduced in size!

DBCC SQLPERF (LOGSPACE);

--Database Name Log Size (MB) Log Space Used (%) Status

--ADMIN 0.8046875 74.45388 0

--take a transaction log backup. Note that the size of the log file space used is reduced, but not the actual size of the file

--this is because, when you take a log backup, the inactive transactions are removed from the log file to the backup file!

BACKUP log admin

TO DISK = 'C:\FullBackups\admin.bak'

DBCC SQLPERF (LOGSPACE);

--Database Name Log Size (MB) Log Space Used (%) Status

--ADMIN 0.8046875 48.36165 0

**Viewing inside the SQL Server Database Transaction Log to prevent internal fragmentation**

**(Auto growth and sizing of auto growth log)**

There is a function called (fn\_dblog) which requires a beginning LSN and an ending LSN (Log Sequence Number) for a transaction, but for our purpose we will use NULL as starting and ending values. The following example will show how the transaction log records multiple rows of data for any activity against the database and how a transaction log backup truncates the transaction file, once it’s written to the backup log. (fn\_dblog) A ways to view the contents of the log. DONOT RUN THIS COMMAND IN PRODUCTION AS IT IS UNDOCUMENTED.

USE [master];

GO

CREATE DATABASE VIEWLOG;

GO

-- Create tables.

USE VIEWLOG;

GO

CREATE TABLE Cities

(City varchar (20));

USE VIEWLOG;

GO

Backup database fnlog to disk = 'C:\FullBackups\VIEWLOG.back'

Insert into Cities values ('NewYork')

Go 1000

Select COUNT (\*) from fn\_dblog (null, null)

Backup database fnlog to disk = 'C:\FullBackups\VIEWLOG.back'

Select COUNT (\*) from fn\_dblog (null, null)

Backup log fnlog to disk = 'C:\FullBackups\VIEWLOG.back'

Select COUNT (\*) from fn\_dblog (null, null)

Select \* from fn\_dblog (null, null)

Options to truncate the log file:

1. Backup the database.
2. Detach the database,
3. Delete the transaction log file. (or rename the file, just in case)
4. Re-attach the database
5. Shrink the database
6. None of the above