Restore SQL Server from Transaction Log

Transaction Log (also known as Database Log or Binary Log) is an action history that is executed by the database system to ensure ACID properties when the system fails.

***Network administration* - Backup is a part of the work that needs to be done during the SQL server upgrade and operation. And the rest of the work is that the restore process is performed every time the SQL server has an error. One of these situations may be related to the restore from the Transaction Log file Backups.**

Transaction Log (also known as Database Log or Binary Log) is an action history executed by the database management system to ensure ACID (atomicity, consistency, isolation, durability) properties when the system is error.

There is a problem with Transaction Log Backups that require a restore from multiple files rather than just a full backup file. To successfully restore the database, you must use every file Transaction Log Backups that have been created and they must be restored in order when created. If a Transaction Log Backup file fails, you will not be able to restore any Transaction Log Backup files after that error file. They need to be restored in order and you cannot ignore any files.

The above are 5 basic steps to take when performing a restore to a SQL database from Transaction Log.

Step 1: Backup the Transaction Log operation

In case the SQL server fails and you need to restore to another server, you first need to backup the Transaction Log operation to save the existing transactions that have not been backed up to an existing Transaction Log Backup. . To create Transaction Log Backup, please use the command with the following syntax:

```
BACKUP LOG Northwind
TO DISK = 'C: SQLBackupNorth.bak'
WITH NO_TRUNCATE
```

Where C: SQLBackupNorth.bak is the address to save the Transaction Log Backup file.

This command will create another Transaction Log Backup that will be used during the restore process.

Step 2: Determine the data to restore

If you do not know any important information in the database that needs to be restored, then you should query the SQL server manifest in msdb, which will display all backup files on the server, including Backup files are created with Maintenance Plans, wizards in Enterprise Manager, T-SQL commands and other third-party tools using the integrated SQL Server feature to create backup files.

In msdb will contain the following table types:

1. **backupfile** - Contains a record for each data or Log file that has been backed up.
2. **backupmediafamily** - Contains a record for each vehicle group.
3. **backupmediaset** - Contains a record for each backup tool set.
4. **backupset** - Contains a record for each backup file group.

If you want to perform a complete restore process, you must first make a Full Backup including Differential (update file) and Backup Logs for Differential. The above tables will display the backup file done first, so you need to find the latest Full Backup file and other backup files created after performing a Full Backup.

Step 3: Check the contents of the Backup file

In addition to the **RESTORE** command, which restores backup files, some other **RESTORE** commands allow you to check the contents of backup files. These commands include **RESTORE HEADERONLY** and **RESTORE FILELISTONLY** .

RESTORE HEADERONLY

This command allows you to check the general information of all backup files on a specific backup tool. This command is useful when you need to restore from multiple backup files or if the backup file is from another server that you do not manage. To check what is stored in that backup file, run this command in Query Analyzer.

RESTORE FILELISTONLY

This command allows you to check the database list and log files in the backup file group, the size of the data and the Log files. LogicalName and PhysicalName are the main data components used during the restore process. The command syntax is in the form:

```
RESTORE FILELISTONLY FROM DISK = 'C: SQLBackupNorth.bak'
```

Page 2 : **Step 4: Select the restore option**

Step 4: Select the restore option

There are several options to apply when restoring the backup file, including the Transaction Log Backup files. Here are some restore options:

NORECOVERY

This option allows to restore additional backup files. You can use it when restoring Full, Differential or Transaction Log Backup. Use the following command:

```
RESTORE DATABASE NORTH
FROM DISK = 'C: SQLBackupNorth.bak'
WITH NORECOVERY
```

RECOVERY

This is the default option if no options are selected. This hconj option will be applied to the last restore process. When applied, you cannot restore additional backup files; if you want to restore additional backup files, you must perform a restore from scratch. This option can be used when restoring Full, Differential or Transaction Log Backup. To select this option, use the following command:

```
RESTORE LOG NORTH
DISK FROM = 'C: SQLBackupNorth_Log.bak'
WITH RECOVERY
```

STANDBY

This option allows you to switch the database to Read-Only mode, but it still allows to restore additional Transation Log files. This option can be used when restoring Full, Differential or Transaction Log Backup. The option to select this option takes the form:

```
RESTORE LOG NORTH
DISK FROM = 'C: SQLBackupNorth_Log.bak'
WITH STANDBY = 'c: undo.ldf'
```

MOVE

When restoring the database to another server, you may have to use the MOVE option if the servers are not installed by the same method. As mentioned above, when using LogicalName and PhysicalName from the RESTORE FILELISTONLY command. The MOVE option allows you to move physical files to another location on the server. This option should be used for all backup file types including Full, Differential and Transaction Log. The option to select this option takes the form:

```
RESTORE LOG NORTH
DISK FROM = 'C: SQLBackupNorth_Log.bak'
WITH RECOVERY,
MOVE 'Northwind_Data' TO 'c: dataNorthwind.mdf',
MOVE 'Northwind_Log' TO 'c: dataNorthwind _log.ldf'
```

Step 5: Select the Restore time

In addition to completely restoring Transaction Logs, SQL Server also has options to stop at a specific time or transaction mark. You can choose these options when you know when or where errors occur on this database, you can restore database transactions for a specific point to avoid errors. For example, if someone deletes every record in a table, you may want to restore the database to a point before restoring it to the table that was deleted.

STOPAT

This option restores every delivery performed up to a certain time. For example:

```
RESTORE LOG Northwind
DISK FROM = 'C: SQLBackupNorth_Log.bak'
WITH RECOVERY,
STOPAT = 'Sep 22, 2009 09:00 AM'
```

The next two commands help you recover transactions that use transaction marks that must be named transactions used in the application. If using unnamed transactions, this option will not work.

STOPATMARK

With this option you can restore all transactions that occurred until the Invoice1024 transaction milestone. Add commands with the following syntax to select this option:

```
RESTORE LOG Northwind
DISK FROM = 'C: SQLBackupNorth_Log.bak'
WITH RECOVERY,
STOPATMARK = 'Invoice1024'
STOPBEFOREMARK
```

This option restores all transactions performed before the Invoice1024 transaction milestone. To use this option, please add the command with the following syntax:

```
RESTORE LOG Northwind
DISK FROM = 'C: SQLBackupNorth_Log.bak'
WITH RECOVERY,
STOPBEFOREMARK = 'Invoice1024'
```

Some recovery options can be used for all backup files and certain options can only be used for Transaction Log Backup files.