

How to Perform an Emergency SQL Server User Database Restore

By Brad M. McGehee

Microsoft SQL Server MVP

Director of DBA Education—Red Gate Software



Brad M. McGehee

It's inevitable. Whether it's tomorrow, or next year, you will have to perform an emergency user database restore. Here are some tips to get you started.

First Steps

- Be sure your production user databases are set to the Full Recovery mode, assuming you want to be able to recover as much data as possible during an emergency restore.
- Plan and practice. Maintain a document that details how you can perform an emergency database restore based on your organization's particular needs. This document should outline every step and command in detail. In addition, practice performing restores periodically, so you don't forget how.
- When you discover you need to perform an emergency restore, pause and take a deep breath to think carefully about what has happened and what you need to do next. Rash actions end up with poor results.

Restoring a Single User Database Running in Full Recovery Mode

- Assuming it is possible, back up the transaction log of the damaged database before you begin the emergency restore. This allows you to recover as much data as possible. If you cannot backup the transaction log, then all of the data since the last backup will be lost. Backing up a transaction log before a restore is called a tail-log backup.

When you perform a tail-log backup, the database may be either online or offline (depending on whether the database you are performing the tail-log backup on is corrupt or not). If the database you are restoring is online, back up the tail of the log using the WITH NORECOVERY option. If the database is offline and corrupt, perform a tail-log backup using either the WITH NO_TRUNCATE or the WITH CONTINUE_AFTER_ERROR option.

If you can't access the disk to perform a transaction log backup, skip this step. In this case, any data that has not been backed up will be lost.

```
--Backup transaction log if database is accessible
BACKUP LOG database_name
TO DISK = 'File_Path\transaction_log_name'
WITH NORECOVERY;
GO
```

```
--Backup transaction log if database is not
accessible
BACKUP LOG database_name
TO DISK = 'File_Path\transaction_log_name'
WITH NO_TRUNCATE;
GO
```

- You will always begin an emergency user database restore with the most current full database backup using the WITH NORECOVERY option.

```
--Restore the most current full backup
RESTORE DATABASE database_name
FROM DISK = 'File_Path\full_backup_name'
WITH FILE=1,
NORECOVERY;
GO
```

- If you have a differential backup available, restore it using the WITH NORECOVERY option. If you don't use differential backups, skip this step.

```
--Restore the most current differential backup
RESTORE DATABASE database_name
FROM DISK = 'File_Path\differential_backup_name'
WITH FILE=1,
NORECOVERY;
GO
```

- If you have one or more transaction log backups, start restoring them with the first log backup taken after the backup you just restored. For all transaction logs, restore them using the WITH NORECOVERY option. When restoring the last transaction log backup, you restore it to a specific point in time. The point in time can be the most recently available backup, a specific date and time, or a marked transaction. You will need to repeat this step for each transaction log you need to restore. The last transaction log restored will be the tail-log backup you performed earlier (assuming this was possible).

```
-- Restore each log backup, in order, one at a time
RESTORE LOG database_name
FROM DISK = 'File_Path\transaction_log_name'
WITH FILE=1,
NORECOVERY;
GO
```

- Once all the transaction logs are restored, recover the database using RESTORE DATABASE database_name WITH RECOVERY.

```
--Recover the database so that it is ready for use
RESTORE DATABASE database_name
WITH RECOVERY;
GO
```

Your user database has now been restored. If the database is restored on the same physical server where it was originally located, then security will be properly mapped. If you restore this database on a different server, you will have to manually resolve any security issues between the restored database and the master database of the new server. See SQL Server Books Online for more specific information on the commands described above.

Brad M. McGehee is a Microsoft SQL Server MVP with over 10 years' SQL Server experience. He founded the popular community site SQL-Server-Performance.Com where he now acts as the technical editor and forum moderator. Brad is also a frequent speaker at SQL PASS, SQL Connections, SQL Server user groups, and other industry seminars and he is the author or co-author of more than 12 technical books and over 100 published articles. He spends what time he has left with his family in Hawaii.

Sponsored by



www.red-gate.com

redgate®

ingeniously simple tools