



Why just backing up your database isn't enough

How to improve on your existing current backup strategy to ensure faster, easier, and more successful recoveries from a variety of common data loss scenarios.

Introduction

The problem – Although critical to a successful disaster recovery plan, database backups alone are inadequate for dealing efficiently with most common data loss recovery scenarios.

Dealing with the full cross section of recovery scenarios from small inadvertent changes up to full scale database disasters requires more than a database backup. Fully restoring a database from a backup and reconciling it to the current database can be a time consuming and error prone process. It will also risk the loss of any data entered subsequent to the latest backup. Backups won't help recover lost data entered since the last backup nor will it help with file based systems like Share Point.

“Simply backing up your database isn't enough to adequately protect against data loss.”

The Solution – A more comprehensive approach is required to ensure a faster, more precise recovery from data loss, that doesn't jeopardize newer data/changes. Such a solution should complement existing backup and recovery plans and work seamlessly with native SQL database backups. *

This article discusses how [ApexSQL Recovery Studio](#), a powerful bundle of recovery tools, can complement your existing backup plan to ensure quick and seamless recovery from many cases of inadvertent loss of data, no matter how large or small. The tools in [ApexSQL Recovery Studio](#) offer a level of precision that enables DBAs to isolate changes and do more granular restorations quickly and without the risk of losing newer production data. In many cases, these tools can be installed after the fact, when there is a disaster, and read database backups without even having to restore them.

The following mini-case studies illustrate common data loss scenarios and how to efficiently deal with them using the tools included in [ApexSQL Recovery Studio](#).

Row level UNDO from a backup

Problem: When you want to undo changes made to database records, and you have a database backup that contains a copy of the records before the changes were made, undoing changes may be as simple as restoring the backup. The problem with backup restoration, though, is that

you will lose any changes made since the database was backed up. This is the inherent “cost” of restoring from a backup.

Solution: What you can do instead is compare your current database with the database backup using [ApexSQL Data Diff](#) and **selectively** isolate changes that you want to undo from the comparison results. Changed records will be shown as *Different*, while deleted or inserted records will be shown as either *Missing* or *Additional*. Synchronize only the rows changes you wish to revert, without risking the loss of new data, all without even restoring your backup!

✖ **Different (6)**
 The data rows in both objects with differing values in one or more of their column pairs

HumanResources.Employee							
? Missing (0) ≡ Identical (285) ✖ Different (6) + Additional (2)							
<input checked="" type="checkbox"/>	EmployeeID	NationalIDNumber	NationalIDNumber	ContactID	ContactID	LoginID	LoginID
<input checked="" type="checkbox"/>	2	253022876	253022876	1030	1030	adventure-works\kevin0	adventure-w
<input checked="" type="checkbox"/>	3	509647174	509647173	1002	1002	adventure-works\roberto0	adventure-w
<input checked="" type="checkbox"/>	4	112457891	112457892	1290	1290	adventure-works\rob0	adventure-w
<input checked="" type="checkbox"/>	6	24756624	24756624	1028	1028	adventure-works\david0	adventure-w
<input checked="" type="checkbox"/>	8	690627818	690627818	1071	1071	adventure-works\ruth0	adventure-w
<input checked="" type="checkbox"/>	9	695256908	695256907	1005	1005	adventure-works\gail0	adventure-w

Record: 1 Of 6

What if you **don't** have a database backup containing the original changes? If your database is in full recovery mode, then you can find the changes you made on your database's transaction log instead. Using [ApexSQL Log](#), examine your transaction log and find the transaction that executed the changes and roll it back.

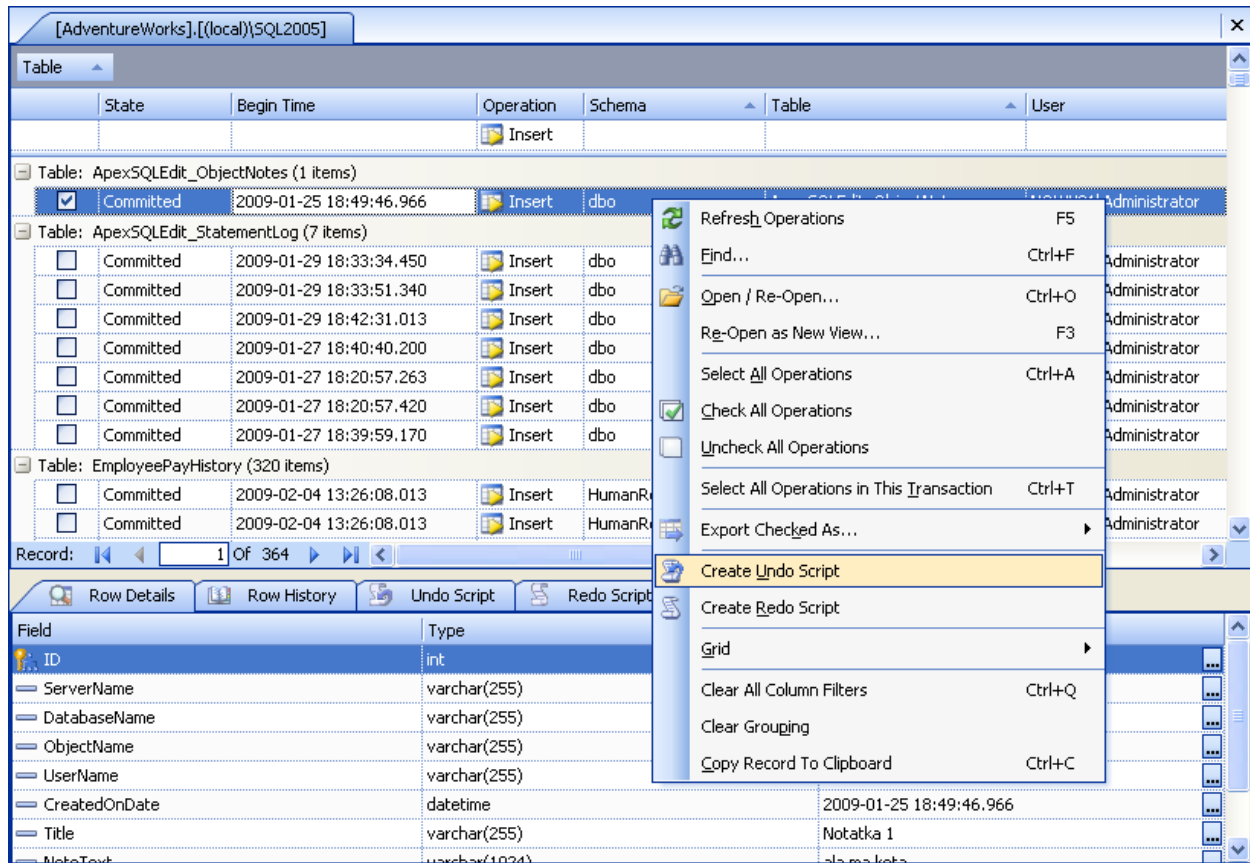
Rogue transaction rollback

Problem: Simple SQL coding mistakes like an UPDATE without a WHERE clause, end users actions and/or client application malfunction are the most common forms of inadvertent data loss. In these scenarios a means to quickly isolate and UNDO isolate transactions, without requiring a full restore of the last backup, would be ideal.

Solution: If your database is in full recovery mode, find the transaction you want to rollback from the transaction log using [ApexSQL Log](#) and roll it back. Transactions can be searched for



by time, user, application, and computer among others. All affected record changes will be rolled back.



Recovering truncated data

Problem: Truncate table operations will remove the entire contents of a table almost immediately. Although an efficient means to empty a table, an inadvertent truncate can be disastrous as well as provide unique challenges to data recovery.

Solution: Use ApexSQL Recover to connect to the database where the TRUNCATE was executed and use the **Recover Lost Data due to Truncate operation** recovery option. Do this immediately after the TRUNCATE is executed for a higher chance of successful recover.



Object level recovery - Recovering a dropped table

Problem: If you accidentally drop a table and you have a database backup, you can restore the backup and port the table and its contents to your current database but this can be a time consuming process. A means to recover just that object, without having to even restore a backup, would be faster.

Solution: You can save time by recreating just the dropped table using [ApexSQL Diff](#). Simply compare your current database with another database containing your table's schema and synchronize. To restore the data, use [ApexSQL Data Diff](#) to read the database backup and synchronize all records that you want to restore.

Another way to do it would be to use [ApexSQL Recover's Recover table data from a database backup](#). The approach is simple: select your backup file and pick the table(s) that you dropped. This option would generate a recovery script to restore only the data found on the tables that you selected.

If you don't have a database backup, use [ApexSQL Recover's Recover Lost Data due to a DROP table operation](#) instead. This option would generate a recovery script that would recover most of your data and restore your dropped table's structure to its bare minimum. Using [ApexSQL Diff](#), compare the table restored by [ApexSQL Recover](#) to a copy of the original table structure (use an older version of the database) and synchronize to apply the structure differences to the restored table.

Undoing schema changes

Problem: Developers/DBAs can sometimes accidentally change or drop objects such as stored procedures, functions, or views. Without object versioning though quickly reverting these changes can be difficult.

Solution: Objects can be saved in a version control system (as scripts) that will allow [ApexSQL Diff](#) to compare and synchronize directly against your repository, without having to rebuild your versioned database. [ApexSQL Diff](#) works natively with Team Foundation Server, Subversion, VSS another popular source control systems.

[ApexSQL Diff](#) provides for an even simpler means of versioning your database schema and objects though, via proprietary schema snapshots that can be scheduled for nightly creation via

the command line interface. This feature allows for a daily, versioned history of your database schema in a format that allows ApexSQL Diff to easily compare against and use to revert changes.

Source	Destination	Schema	Action	Status
HumanResources				
- Status: (4 items)				
dEmployee	dEmployee	HumanResources	<input type="checkbox"/> Ignore	= Equal
Department	Department	HumanResources	<input type="checkbox"/> Ignore	= Equal
vJobCandidateEducation	vJobCandidateEducation	HumanResources	<input type="checkbox"/> Ignore	= Equal
vJobCandidateEmployment	vJobCandidateEmployment	HumanResources	<input type="checkbox"/> Ignore	= Equal
+ Status: ? (22 items)				
- Status: X (12 items)				
<input checked="" type="checkbox"/> Employee	Employee	HumanResources	<input checked="" type="checkbox"/> Update	X Not Equal
<input checked="" type="checkbox"/> EmployeeDepartmentHistory	EmployeeDepartmentHistory	HumanResources	<input checked="" type="checkbox"/> Update	X Not Equal
<input checked="" type="checkbox"/> EmployeePayHistory	EmployeePayHistory	HumanResources	<input checked="" type="checkbox"/> Update	X Not Equal
<input checked="" type="checkbox"/> JobCandidate	JobCandidate	HumanResources	<input checked="" type="checkbox"/> Update	X Not Equal
<input checked="" type="checkbox"/> Shift	Shift	HumanResources	<input checked="" type="checkbox"/> Update	X Not Equal
<input checked="" type="checkbox"/> uspUpdateEmployeeHireInfo	uspUpdateEmployeeHireInfo	HumanResources	<input checked="" type="checkbox"/> Update	X Not Equal

Like snapshots, an older version of your database can also help you here. Use ApexSQL Diff to compare the older version of the database w/ the online database and synchronize the missing objects.

Soon ApexSQL Diff will be able to compare against Backups as well. Look for this feature to be released sometime in Q1 2008.

As a last resort, that is, if you don't have anything else but the database you're currently connected to, use ApexSQL Recover's **Recover lost objects due to a dropped operation** option. This will generate a script to restore your dropped objects.

Recovering a corrupted Database

Problem: The database files of SQL server are saved in .mdf file format. If the database file gets corrupt or damaged, users won't be able to access the data in them rendering your database unusable and unless you have the right tools, unrecoverable.



Solution: With a corrupted database, you can still restore some data on your tables as long as you still have the corrupted file. Your corrupted database's MDF file would have to be scanned for any recoverable data and restored to a new database.

Extract recoverable data using [ApexSQL Recover's Recover from corrupted database or detached mdf](#) option which can help you read the corrupted MDF. Recovered data will be in the form of INSERT scripts.

Run the recovery script on a fresh database that already has your original database's schema. To re-create your database schema on a new database, use an older version of your database that contains your schema (if a backup is available, restore it). Use either ApexSQL Script or [ApexSQL Diff](#) to deploy the schema to a new database.

If SharePoint documents are deleted

Problem: Anybody who has worked with Windows SharePoint Services is aware of the possibility of inadvertently deleting documents. Any file found the SharePoint [Recycle Bin](#) can still be restored. But, *what if they're not there anymore?*

Solution: SharePoint documents are stored in the DocVersions table of the SharePoint database. The column Content contains the actual document in binary format. To restore the documents, you would need to restore the deleted records.

If you have a database backup that contains the deleted records, you can use ApexSQL Data Diff to find which records were deleted by comparing the backup with your online SharePoint database. The deleted files will appear as Missing in the comparison result. Choose which records you want to recover from the comparison result and synchronize to restore the documents.

If you don't have a database backup, you can use [ApexSQL Recover's Recover BLOB data as files](#) option to recover the deleted documents.

Select Recovery Option

Select the option that best describes the recovery that you are trying to perform.



Recover from specific operations:

- Recover lost data due to Delete operation
- Recover lost data due to Truncate operation
- Recover lost data due to a Drop Table operation
- Recover lost objects due to a Drop <object> operation

Special Recovery options:

- Recover from corrupted database or detached mdf
- Recover table data from a database backup
- Recover BLOB data as files

Description

Allows recovery of BLOB data and saving it as a file from both deleted, and non-deleted records

The documents will be recovered as FILES. Re-upload the files that you want to restore to your Document Library.

Summary

The modern SQL Developer/DBA faces many risks from minor data loss to full scale database disasters. Backups, although critical, aren't precise enough to isolate object, transaction or row level changes. Full recoveries from backups can result in further loss of data, subsequent to the last backup, be labor intensive as well as error prone. ApexSQL Recovery Studio, combined with existing backup and recovery plans, provides the comprehensive toolset required to effectively and efficiently manage the full spectrum of data loss problems.


* ApexSQL Recovery Studio works directly with Hyperbac's backup tool for SQL Server. Other 3rd party backup tools can be converted to native backups, at which point they can be read by ApexSQL software




A powerful set of tools to complement your disaster recovery plans

 Includes 1 yr FREE Support and Upgrades

 **\$1,499** represents a discount of **50%** off the combined prices of the individual products (\$3,057)

 Only a **25%** Maintenance renewal fee per year (or **\$375**)

 A single download and installer for all tools, allows for quick and easy install of all tools at once.

ApexSQL Recovery Studio includes the following products:



ApexSQL Recover - Recover deleted, dropped, corrupted, or lost data

- Object-level recovery from a backup
- Recover lost data due to DELETE, TRUNCATE, DROP operations
- Recover from corrupted database or detached MDFs
- Recover BLOB data as files



ApexSQL Log - Audit data changes, rollback and reconstruct transactions

- Transaction and operation-level rollback
- Revert changes that happened even before the software was installed



ApexSQL Diff - Compare and synchronizes SQL Database Structures

- Structure auditing through daily periodic capturing of database snapshots
- Compare and synchronize databases, snapshots, and scripts to revert changes



ApexSQL Data Diff - Compare and synchronizes SQL Database Data

- Row-level recovery from a backup



ApexSQL Source Control - Expand the source control capability of ApexSQL products

- Schema change Version-control and labeling
- Allows direct comparison and deployment from Source Control



Case Studies

Visit the ApexSQL Recover Studio product page for comprehensive step by step articles on the following topics.

- Disaster recovery from a corrupted database
- Inadvertent operation (truncate, delete, drop, update) rollback without a backup
- Object level recovery from a backup
- Row level recovery from a backup
- Transaction level recovery without a backup

Contact us

For questions, please contact ApexSQL at sales@apexsql.com or 919.968.8444 to speak to a sales representative or technical support consultant.