By Gregory Larsen January 3, 2013

As a DBA one of the most important tasks you might be asked to do is to restore an application database. When a database becomes corrupted for some reason you need to be able to restore the data to the last know point in time when the database was in a consistent state. In order to do this you need to make sure you have backup strategy that supports a variety of different restore situations. Therefore developing a backup procedure that makes sure you can do restores should be one of your top priorities as a DBA. If you have not planned accordingly and your restore is not successful, then you might need to polish off your resume and start looking for a new job. In this article I will be discussion some of the concepts you should consider when developing, building and testing a backup strategy.

Understanding Your Backup Requirements

One never knows what might happen to a database. It might get corrupted due to some bad code, or bad hardware. It might even get lost due to a data center fire, flooding, or some other catastrophic event. In order to recover from data corruption or total loss of a database you need to back up your databases based on some requirements.

Before you can build a backup strategy you need to determine your backup requirements. By backup requirements I mean you need to identify how much data you can afford to lose, and how long you can take to recover lost or corrupted data. How much data can you lose is commonly called recovery point objective (RPO), whereas how long can you take to recovery your lost data is commonly called recovery time objective (RTO).

In order to identify the RPO and RTO for your environment you need to meet with your data owners/customers. Your customers will identify how critical their data is and whether or not they are able to re-create, or re-enter data should their database become corrupted, or get lost due to a disaster. Here is a partial set of questions you should ask your customers to help define backup requirements:

- How often does their data change?
- Can they re-enter, or re-create data if the data gets corrupted or lost?
- If they can re-enter, or re-create data, how far back in time would be reasonable for them to re-enter, or re-create their data?
- How long can they live with their system being down while a recovery operation is being performed?
- How much data can they lose?

Your customers' requirements will guide you in determining what kind of backup and restore strategy you need to design, build and test.

Offsite

Backup requirements are important, but you also need to think about where you place your database backups. You need to have your backup files readily available if should you need them. But you need to make sure they are not vulnerable to a server melt down, or a total data center disaster. Therefore you need to consider where to store your backup files onsite, as well as where you might store your backup files offsite.

For onsite backups you need to write or store your backup files so they are available in the event your database server should crash, or a database gets corrupted. Placing your backups directly on your database server allows you to write your backups fast because there is no network I/O that has to occur. But if your backups are on your database server, and the server should melt down then you have lost your backups. An alternative to this is to write your backups to a network drive. This keeps your backups off your database server, but does not provide for a secondary location should the machine where your network backups are stored have a problem. It is always best to make sure your backups are copied to a secondary backup location, like tape, or a network backup storage solution. This way should the primary backup device not be available, than you can always get your backups from the secondary location. You should make sure your backups are copied to the secondary location soon after they are created. This minimizes the time that you only have a single backup copy.

Having your backup's onsite allows you to quickly obtain a backup file should you need to do a restore. But what happens when your onsite facility has a problem like a major fire, flood, or is damage by some other natural disaster? If something happens to your onsite backups, then you will not be able to recover unless you have taken the precautions of copying your backups to an offsite location. As you are developing your backup solution you need to develop a plan that copies your backup files to a secure offsite storage location.

How Long Should You Keep Those Backup Copies?

When you work with the data owners to identify the backup requirements for a given database you need to also work out how long you need to keep those backups. The more backups you try to keep the more disk space you will need for those backups. There is no magic set of questions, or timeframe you should consider when determining how long to keep backups. Like most SQL Server questions it really depends on your situations as to how long you should keep those backups. You will need to assess your environment and determine how many copies and how long you want to keep those copies for your situation. You should consider the following when determining how many backup copies you should keep:

- How often your database is updated?
- How long it might take for a data issue to be discovered?
- How far backup in time is realistic for you to restore your database?

When you determine how many backups to keep you need to balance how far back you will recover verses the amount of disk space you have. Once you decide how long to keep backups and how many copies you will you need, make sure you clearly communicate to the data owners how many copies you plan to keep and how far they can go back in time to recover their databases. It is never good to have a data owner ask you to restore their database back to 30 days ago, when you only have backups for the last 14 days.

Types

There are a number of different database backup types. The table below lists the different backup types and a short description of how they are used:

Backup Type	Description
COPY_ONLY	Is either a FULL or TRANSACTION LOG backup that doesn't affect the normal sequence of backups
FULL	Backs up all the data extents in a database
DIFFERENTIAL	Backs up all the data extents that have been changed since the last FULL, PARTIAL, or FILE backup
TRANSACTION LOG	Backs up the transaction log information (only relevant for databases in FULL recovery mode)
FILE	Backs up one or more database files or file groups
PARTIAL	Backs up all read-write file groups and optionally one or more read only files

Each one of these is a little different and is used for different purposes. The three most common backup types used are FULL, DIFFERENTIAL and TRANSACTION LOG.

You need to determine which combination of these backup types work best for your environment. Here are some guidelines to use for determining when to use these different database backup types:

- If you are not concerned about recovering data that might be entered between database backups then you might be able to just perform FULL database backups. If this meets your needs then you should also put your databases in SIMPLE recovery mode to keep your transaction log from growing uncontrollably.
- If your database is large, takes a long time to backup, your backup files are large, and/or not very much of the data in a database changes over time, then it might be appropriate to take a combination of FULL and series of DIFFERENTIAL backups.
- If your data changes quite often, and you want to do point in time recoveries then you might want to consider deploying a combination of FULL and TRANSACTION LOG backups , and possibly DIFFERENTIAL backups depending on the size of your database.
- FILE and PARTIAL backups are useful if you know that only a particular file or file group has been updated.
- If you want to take a special backup say to move a database to your test server for testing, but don't want to mess up the chain of production backups, then you might consider using the COPY_ONLY backup option.

Before you implement a particular set of database backups to back up your database environment, make sure you completely understand your RTO and RPO requirements, and how each of the different backup types and storage locations will help you meet your RTO and RPO timeframes.

Building a Backup Solution

Once you have develop your backup requirements, and where you plan to place your onsite and offsite backups you need to consider how are you going to take your backups. There are three common options most people use to create their database backups: Buying a custom off the self (COTS) solution, using the built-in maintenance plan method that comes with SQL Server, or building their own homegrown backup solution. Each of the options has their own merits.

The COTS features th

easy to implement and has enough functionality to provide most shops with a reliable backup solution. If you have unique backup requirements you might find the only way to meet your requirements is by building a homegrown backup solution. One thing to consider when building a home-grown backup solution is you will need to maintain it overtime, whereas the COTS and the maintenance plan solutions are maintained by the vendors.

Verifying You Can Restore

Just because you have backed up your databases doesn't mean you can restore your databases from a backup. After you have created your backups you should probably verify that the backups are good. There are a couple of options to accomplish this.

One of those options is to occasionally perform an actual restore of your databases from those backups you have taken. This might not be practical in your production environment, but you should at least periodically test to make sure you can restore your backups in some non-production environment.

Additionally you can verify the backups are written correctly. This can be accomplished with the "BACKUP VERIFYONLY" command. This command actually reads the backup file and verifies that the data in the backup can actually be restored. By using the verify option SQL Server will be able to detect if the backups were writing correctly.

Restoring a single database backup only means you can restore that single backup, but not a complete set of backups. You should consider periodically doing a complete bare metal restore. This is where you would find an unused machine and try to restore the OS, SQL Server software, and all the databases including the system databases. Performing a complete server restore will give you the confidence in your backup strategy and will reduce your anxiety when you have a real disaster where you have to do a complete restore.

Security of Your Backup Files

When you build your backup plan you need to consider where you place those backups and how secure that location might be. Since backups contain your corporate data, you need to make sure the backup location is adequately secured. By securing your backup location you ensure your backups don't mistakenly get in the hands of someone that could misuse the backups.

If you have sensitive data in your databases, like credit card numbers or personal identifiable data you need to be concerned who has access to your database backups. If your database backups are not encrypted then it would be easy for someone to get a hold of your data by just obtaining a database backup and then restoring it onto a server where they have SQL Server SysAdmin permissions.

If you do have confidential and/or sensitive data stored in your databases you should consider making sure your backup location is only available to those that are authorized to see, manage and use your database backups. Alternatively you could require your applications to encrypt the confidential data within the database, or encrypt the database backups in some way. One way to encrypt your backups is to turn on Transparent Database Encryption (TDE) on those databases that contain non-encrypted confidential data. If you use TDE to encrypt your backups, then you will need to develop a method to manage and restore the certificates that were used encrypt the database. Without restoring the certificates you cannot restore your TDE e × how to encrypt database backups is outside the s

Safety

When you are developing a backup solution you need to first identify the RPO and RTO for your backups. Once you have those timeframes it will help guide you in determining what types of backups you need, as well as where you will store your backups.

Safety in numbers is something you need to consider when developing your backup strategy. Just like in the animal world where certain species feel more comfortable when they have large numbers of their own kind around, I also feel better when I have more backups then I need. It is better to have too many backups than not enough.

Lastly a backup solution is only effect if you can restore the database backups you create. You need to verify your backups periodically and should consider at least once a year or so to perform a full blown server recovery test.

See all articles by Greg Larsen