Sybase Plugin

Bacula Systems Documentation

Contents

Contents

1 Overview

This user guide presents various techniques and strategies to backup Sybase Adaptive Server Enterprise with Bacula Enterprise.

1.1 Scope

This paper will present solutions for Bacula Enterprise version 10 and later, which are not applicable to prior versions.

2 Executive Summary

IT organizations are constantly being challenged to deliver high quality solutions with reduced total cost of ownership. This white paper presents various strategies to backup Sybase Adaptive Server Enterprise using the Sybase Plugin with Bacula Enterprise version 10.

The Bacula Enterprise Sybase Plugin is designed to simplify the backup and restore operations of your Sybase Adaptive Server Enterprise. The backup administrator does not need to learn about internals of Sybase backup techniques or write complex scripts. The Bacula Enterprise Sybase Plugin supports Point In Time Recovery (PITR) with Sybase Backup Server Archive API backup and restore techniques.

The Bacula Enterprise Plugin is able to do incremental and differential backup of the database at a block level. This plugin is available on 32 and 64-bit Linux platforms supported by Sybase, and supports Sybase ASE 12.5, 15.5, 15.7 and 16.0.

3 Features Summary

- Sybase database online backup and restore using *Sybase Backup Server Archive API*.
- Support for **Full Database** backup and restore, **Cumulative Database** (incremental) backup and restore, **Database Transactions** backup and restore.
- Database backup levels are smoothly mapped to Bacula Enterprise backups levels.
- All backup or restore operations are managed by the database administrator.
- Ability to restore any database backup to an alternate location.
- Direct support of Point In Time Recovery (PITR) with database transaction restoration.

4 Conventions Used In This Guide

- <SYBSERVER> Anything between < and > should be adapted by the user, for example, <SYBSERVER> should be replaced by your current Sybase service name. If your Sybase service name is SYBASE16 a variable SYS=<SYBSERVER> will become SYS=SYBASE16.
- The % prompt means that the command should be run with a normal user such as sybase.
- A # prompt implies that the command should be run with the root account.
- N> means that the command should be run inside an isql utility session, where 'N' is an integer sequential number, i. e. 1, 2, 3, etc.
- * indicates that the command should be run as a bconsole command.

5 Using the Sybase Plugin

5.1 Installation

The Sybase plugin is available as a Bacula Enterprise package for all supported Sybase platforms.

```
bacula-enterprise-sybase-plugin.<version>.[rpm,deb]
```

This plugin has to be installed on the Client where the Sybase database resides. The package will install the following files:

```
/opt/bacula/scripts/bacula-sybase.sh
/opt/bacula/scripts/install-sybase.sh
/opt/bacula/scripts/sybase-backup.sh
/opt/bacula/lib/libsybacula.so
```

Note: On a Debian system it is necessary to install the bacula-enterprise-console package.

5.2 Plugin Configuration

As with all Bacula plugins, the **Plugin Directory** directive in the **FileDaemon** (or) resource of the `bacula-fd.conf` file has to be set:

```
FileDaemon {
   Name = sybase-fd
   (...)
   Plugin Directory = /opt/bacula/plugins
}
```

In order to send commands to the Sybase database, the Bacula Enterprise Sybase Plugin must share files on disk with Sybase. When using packages provided by Bacula Systems, these files are located in `/opt/bacula/sybase` and the directory permissions should allow to read and write files for the Sybase user, i.e.:

```
# ls -ld /opt/bacula/sybase/
drwxr-xr-x. 2 sybase sybase 85 11-22 09:40 /opt/bacula/sybase/
```

where `sybase` is the main user and group of the Sybase Database Unix user. These permissions are automatically set when installing packages, but, if the actual Sybase Unix user is not "sybase", it will be necessary to manually change permissions on this directory and make sure that the changes are still in effect after an upgrade of the Bacula Enterprise Sybase Plugin package.

6 Sybase SBT Configuration

This section describes how to properly install and configure the Bacula Enterprise SBT interface with the Sybase Backup Server.

When running a backup or a restore from Sybase, Sybase Backup Server will need to contact the Bacula Enterprise Director to get information about files and volumes, or to run backup and restore jobs. This communication involves shared FIFO command files, and the `bconsole` program.

When using the `sybase-sbt-fd` plugin, the Director will not be able to start a backup through `bconsole` or directly with a scheduled job. Only the Sybase Backup Server will be able to initiate the session and start a backup. Note that it is still possible run a regular system backup of your Sybase server, and, along with it, use a **RunScript** to call Sybase Backup Server automatically.

6.1 Bacula Configuration

When using the SBT interface, the Bacula console `bconsole` needs to be installed, and the console should be able to connect to your Director and have access to the local **Client**, the backup **Job**s used for Sybase database backups, and any required **Pool**s. To access the Director via `bconsole`, a restricted console will need to be confiured on the Director, and a `bconsole` configuration file needs to be configured on the client:

```
# cat /opt/bacula/etc/bconsole-sybase.conf
# Bacula User Agent (or Console) Configuration File
Director {
   Name = "sybase-dir"
   DirPort = 9101
   Address = 192.168.0.46
   Password = "NotUsed"
}
```

(continues on next page)

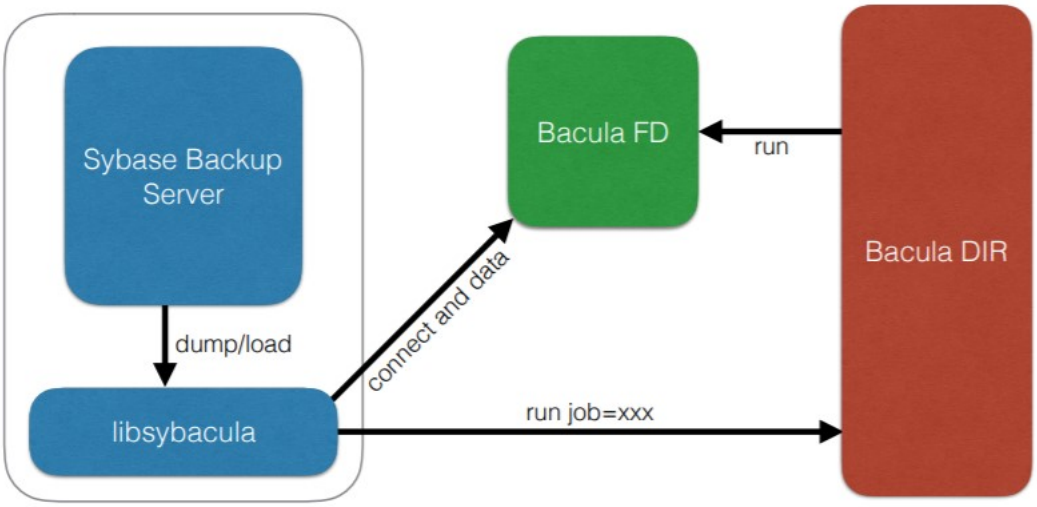

Fig. 1: Interaction Between Sybase Backup Server and Bacula

(continued from previous page)

```
Console {
   Name = "sybase-fd"
   Password = "pass"
}
```

To use a restricted console, the following or a similar **Console** definition and other required resources – **Client**, **Job**, **Pool** and **FileSet** must be explicitly allowed per ACL in the Director's configuration:

```
Client {
   Name = sybase-fd
   Maximum Concurrent Jobs = 10
}

Console {
   Name = sybase-fd
   Password = "pass"
   CommandACL = .bvfs_lsfiles, .bvfs_get_volumes, use, .bvfs_get_jobids, wait
   CommandACL = .bvfs_restore, .bvfs_cleanup, restore, run, gui, .jobs, quit
   # These commands are used only by the install-sybase.sh test
   # procedure and can be commented out after the installation
   CommandACL = show, status

   ClientACL = sybase-fd
   JobACL = SBT-Backup, RestoreJob
   CatalogACL = *all*
```

(continues on next page)

(continued from previous page)

```
   StorageACL = *all*
   FileSetACL = *all*
   PoolACL = *all*
   WhereACL =/
   DirectoryAcl = *all*
   UserIdAcl = *all*
}

Job {
   Name = SBT-Backup
   FileSet = SBT-FileSet
   Client = sybase-fd
   Maximum Concurrent Jobs = 10
   # Adapt the following resources
   # to your settings
   Messages = Standard
   Pool = Default
   Storage = File
}

FileSet {
   Name = SBT-FileSet
   Include {
      Options {
         Signature = MD5
      }
      Plugin = "sybase-sbt:"
   }
}
```

The unix "sybase" user or, more precisely, the user account used by the Sybase installation should be able to execute `bconsole` and read the associated configuration file `bconsole-sybase.conf`, which is **not the default configuration**. It is possible to copy the configuration file to `/opt/bacula/etc/bconsole-sybase.conf` with the following unix commands:

```
# cp /opt/bacula/etc/bconsole.conf /opt/bacula/etc/bconsole-sybase.conf
# chown sybase:sybase /opt/bacula/etc/bconsole-sybase.conf
# chmod go-rxw /opt/bacula/etc/bconsole-sybase.conf
```

6.2 Running Parallel Jobs

In order to run backups or restores using multiple stripes (check *Stripes*), the required resources in Bacula need to be properly configured using the `Maximum Concurrent Jobs` directive to allow concurrent jobs:

- Director: **Director** (ex: 100)
- Director: **Client** (ex: 10)
- Director: **Job** (ex: 10)
- Director: **Storage** (ex: 10)
- Storage: **Storage** (ex: 100)
- Storage: **Device** (ex: 10, or 10 devices grouped in a Virtual Changer)

- Client: **FileDaemon** (ex: 10)

To allow concurrent backup and restore jobs using the same Director Storage resource, the configuration should use a Virtual Changer disk device. See the *Disk Backup* whitepaper about this specific configuration.

6.3 Backup Server Module Installation – `libsybacula`

Once packages are installed, the Sybase Backup Server module `libsybacula.so` file will be present in the `/opt/bacula/lib` directory. A symbolic link to it, in `$SYBASE/$SYBASE_ASE/lib/` is the most convenient way to ensure it is available to the Sybase `backupserver` program. This can be done using the `install-sybase.sh` script available under `/opt/bacula/scripts`:

```
# /opt/bacula/scripts/install-sybase.sh install
Enter the SYBASE base install [/opt/sybase]:
Enter the SYBASE_ASE location at <SYBASE>/[ASE-16_0]:
Enter the unix Sybase account name [sybase]:
Enter the SYBASE service name [SYBASE16]:
INFO: write sybase.env file
INFO: linking sybacula module
INFO: Installation SBT interface OK.
```

or this may be executed manually:

```
# ln -s /opt/bacula/lib/libsybacula.so $SYBASE/$SYBASE_ASE/lib/
# ls -l $SYBASE/$SYBASE_ASE/lib/libsybacula.so
lrwxrwxrwx. 1 root root 30 11-21 09:05 libsybacula.so -> /opt/bacula/lib/libsybacula.so
```

As the library is loaded by the Sybase Backup Server on demand, there is no need to restart any Sybase component after successful installation.

During a manual installation, it is helpfu to insert the proper installation values into the `/opt/bacula/sybase/sybase.env` file which is used by other plugin tools, i. e. `sybase-backup.sh`. This step is optional. The file is automatically generated by the `install-sybase.sh` script.

```
# cat /opt/bacula/sybase/sybase.env
SYBASE=/opt/sybase
SYBASE_ASE=ASE-16_0
SYBASE_OCS=OCS-16_0
SYBASE_USER=sybase
SYBSERVER=SYBASE16
SAPASSWORD=secret
```

Here, `SYBSERVER` is a Sybase server name and `SAPASSWORD` a password for the `sa` user, the default database administrator user name.

6.4 Storage Consideration

Sybase imposes to the Media Manager, Bacula Enterprise, to not multiplex data streams from two concurrent stripes of the same backup job onto the same sequential device. This means that, when tape-based storage is used for Sybase backups, different tape devices have to be used for each concurrent backup job. This restriction does not apply to disk based storage. This limitation implies longer restore times.

6.5 Bacula SBT Configuration

The `libsybacula.so` can be configured in the `/opt/bacula/sybase/sbt.conf` or files, or using the proper arguments to Sybase `dump` or `load` commands. Supported parameters are:

client is a Bacula Client name, i.e. `client=sybase-fd`.

restoreclient is a Bacula Client name used for restore, such as `restoreclient=sybase-fd`.

job is a Bacula Backup Job name used for backup, i. e. `job=SBT-Backup`.

bconsole is a `bconsole` command with arguments used to communicate with Bacula Director, i.e. `bconsole="/tmp/bconsole -n"`. The default is `bconsole -u 60 -n -c /opt/bacula/etc/bconsole-sybase.conf`.

restorejob is a Bacula Restore Job name. If multiple restore jobs are defined in Bacula's configuration and this option is not used, the Sybase SBT Plugin will automatically choose the first restore Job defined. For example: `restorejob=RestoreFiles`

waitjobcompletion sets the Sybase plugin to wait for Job completion at the end of the backup or restore session. The default is to finish the session as soon as possible. Example: `waitjobcompletion`.

jobopt sets additional Job options the same way they can be passed in a `bconsole run` command, for example `jobopt="spooldata=no"`.

ctrlfile is the base path to the control files used to communicate between the Sybase Backup Server and the Plugin, i.e. `ctrlfile=/tmp/sybase`. The default is `/opt/bacula/sybase/sbt`.

ctrltimeout is the timeout to apply when connecting with the Bacula Director, i.e. `ctrltimeout=300`.

retry determines the number of retries when connecting to Bacula, for example `retry=30`. Note that, in a reasonably configured environment, this option should not be necessary or may be set to a relatively low value.

catalog is a Bacula Catalog name, such as `catalog="MyCatalog 2"`. With a Bacula instance which uses one catalog only – which should be the case for most production instances – this option will not be needed.

trace indicates the path to the trace file. Check **tracing** for more information. Example: `trace=/tmp/log.txt`.

debug is the debug level to apply when tracing. Again, see **tracing** for more information. Example: `debug=50`.

The minimal configuration file will require the **client** and **job** options to be set. Note that if a configuration item contains spaces (such as the **bconsole** example above), it needs to be enclosed in double quotes.

Note: The Sybase Backup Server Archive API limits the total size of archive device options (`libsybacula` parameters above) to 127 characters. If the needed parameter string exceeds this limit, a configuration file `sbt.conf` must be used.

```
# cat /opt/bacula/sybase/sbt.conf
client=sybase-fd
job=SBT-Backup
```


6.6 FileSet Configuration

The Sybase SBT plugin (`sybase-sbt`) accepts the following parameters in the Jobs **FileSet** configuration:

ctrlfile is the path to the control files used to communicate between Sybase Backup Server and the Plugin, i.e. `ctrlfile=/tmp/sybase`. The default is `/opt/bacula/sybase/sbt`.

6.7 Testing `sbt.conf` Configuration

To test the Bacula Enterprise Sybase SBT Plugin configuration, the following command can be used as the `root` user:

```
# /opt/bacula/scripts/install-sybase.sh test
INFO: Testing ...
INFO: Connection to the Director OK
INFO: Connection from the Director to the Client OK
INFO: Plugin installed correctly
INFO: Job found on the Director
INFO: FileSet configured on the Director
INFO: RestoreJob found on the Director
INFO: FileSet configured for the Restore job on the Director
INFO: Testing OK.
```

7 Backup

The **Bacula Enterprise** Sybase SBT Plugin supports different backup levels to backup Sybase database or transaction dumps. These levels are mapped as follows:

- Sybase Full Database dump – Bacula Full level
- Sybase Incremental (Cumulative) Database dump – Bacula Differential level
- Sybase Transaction dump – Bacula Incremental level

7.1 Full Database Backup

This is the most basic and simple database backup. It will generate a backup copy of the entire database, including the transaction logs.

To perform a Full Database backup execute a command like the following:

```
1> dump database pubs2 to 'sybacula::'
2> go
```

For more information about the `dump` command and additional parameters check the Sybase documentation: *Reference Manual: Commands 16.0*, *Commands*.

It is advisable to use a `sybase-backup.sh` script to execute a full database backup manually or through a Bacula backup job using the **Run Script** Job sub-resource:

```
# /opt/bacula/scripts/sybase-backup.sh pubs2 Full
```

This would save a single dump file like: `D.pubs2.2018-11-21.14:22:46.000` where: '`D`' indicates a full dump, '`pubs2` is the database name, '`2018-11-21.14:22:46`' is the backup time stamp and '`000`' is a stripe number.

7.2 Incremental Database Backup

The standard full database backup copies entire databases to the backup system (MMS in Sybase terminology) which for large databases can take time and consumes a lot of other resources (archive storage space, cpu, network, etc.). For this reason, Full Database Backups are executed less often, but transaction backups more frequently. In this case, a full recovery to the most current state requires a number of transaction dump loads which consumes resources as well and increase the potentially critical restore time.

To optimize both backup and restore operations it is possible to use Database (Cumulative) Incremental backups which create a copy of all the pages in the database that have been modified since the last full database dump. This backup allows for:

- Reduced recovery time – Recovery time is minimized when compared to a strategy that uses transaction log dumps.
- Reduced backup size, especially for databases that contain big read-only tables.
- Improved backup performance, especially on database loads that have substantial updates on only a subset of the database pages.
- Incremental backups of low-durability databases.

Incremental database backups are a comparatively new feature, available as of Sybase ASE 15.7 SP100. For more information, please see the Sybase documentation: *New Features Guide Adaptive Server Enterprise 15.7 SP100*, section *Enhancements to Backup and Restore*.

To perform a (cumulative) Incremental backup, execute a command such as:

```
1> dump database pubs2 cumulative to 'sybacula::'
2> go
```

More information about the `dump` command and its parameters can be found in the Sybase documentation: *Reference Manual: Commands 16.0*, *Commands*.

Of course, a `sybase-backup.sh` script to trigger a cumulative database backup manually or from a Bacula backup job using the **RunScript** directive is possible:

```
# /opt/bacula/scripts/sybase-backup.sh pubs2 Differential
```

This would save a single dump file like `C.pubs2.2018-11-22.18:04:42.000` where: '`C`' indicates a cumulative dump, '`pubs2` is a database name, '`2018-11-22.18:04:42`' the backup time stamp and '`000`' is a stripe number.

7.3 Database Transaction Backup

This type of database backup will make a copy of a transaction log, and removes the inactive portion of the log, provided the dump transaction is not running concurrently with another database dump.

To perform a Database Transaction backup execute a command such as

```
1> dump transaction pubs2 to 'sybacula::'
2> go
```

More information about the `dump` command and its additional parameters are available in the Sybase documentation *Reference Manual: Commands 16.0* in section *Commands*.

Again, a `sybase-backup.sh` script to execute a transaction database backup manually or by a Bacula backup job could be:


```
# /opt/bacula/scripts/sybase-backup.sh pubs2 Incremental
```

This would store a single dump file like: `T.pubs2.2018-11-20.18:11:00.000` where: 'T' indicates a transaction dump, 'pubs2 is a database name, '2018-11-20.18:11:00' is the database backup time stamp and '000' a stripe number.

7.4 Stripes

A standard `dump database` or command described above runs a single backup stream which corresponds to the execution of a single Bacula job. This solution is convenient for small to moderately sized databases. When backing up large database sets, multiple streams can be used for database and transaction dump or load operations.

To use multiple streams, a suitable `dump` or command which includes `stripe on 'sybacula::'` options is needed. It is possible to use as many `stripe on 'sybacula::'` options as required. Each stripe option added to the command will execute an additional concurrent job with Bacula, which implies that concurrency limits will need to be verified or adjusted. Please see *Running Parallel Jobs* for more detailed information.

The following example executes a full database backup job with three concurrent backup streams, and accordingly three Bacula backup jobs.

```
1> dump database pubs2 to 'sybacula::'
2>  stripe on 'sybacula::'
3>  stripe on 'sybacula::'
4> go
```

Multiple stream functionality is available for any backup or restore job type. The multiple stripes functionality can be used for particular backup levels, for example with Full Database dumps, leaving other backup levels with a single stream only.

Up to 32 streams, including the main stream in the `to 'sybacula::'` clause, can be used. This is a Sybase ASE limit.

If a backup has been performed with multiple stripes, a restore needs to be run with an identical number of stripes.

7.5 Database System Objects

System Database Backups

The Bacula Enterprise Sybase SBT Plugin can be used to backup and restore the system databases `master`, `tempdb`, `model`, and `sybsystemdb` like any other user databases. These backups will be available in the Bacula catalog like any other database.

```
# /opt/bacula/scripts/bacula-sybase.sh
Updating cache ...
Searching catalog ... Jobs found.
Full dump:      D.master.2018-11-22.17_15_01
Full dump:      D.model.2018-11-22.17_21_21
Full dump:      D.sybsystemdb.2018-11-22.17_21_29
```


7.6 Scheduling Backups

The following considerations should be taken into account when scheduling Sybase Database backups.

- Any type of dump can be run while a Sybase ASE database is running. However, any database backup could slow down the system by consuming resources like disk I/O, network throughput, CPU cycles, etc. It is recommended to execute large dumps (especially full database dumps) during lower database utilization periods.
- Implementing cumulative database backups will reduce backup size, especially with databases that hold big read-only tables.
- The `master` database should be backed up regularly and frequently. In addition to regular backups, `master` should be backed up after each `create database`, `alter database`, and `disk init` command issued.
- The `model` database should be backed up whenever it is modified.
- The `dump database` backup command should be run immediately after creating a database, to make a copy of the entire database. `dump transaction` can not be run on a new database until a full database dump has been performed.
- Each time a cross-database constraint is added, or a table containing a cross-database constraint is dropped, both of the affected databases should be dumped.

Warning: Loading earlier dumps of these databases can cause database corruption. This is a Sybase ASE limitation.

- Use the *SAP Adaptive Server Enterprise*, section *System Administration Guide* documentation to plan for backup and recovery.

8 Restore

8.1 Listing Database Dumps

Before any database restore or recovery is started, a list of available database or transaction dumps is needed to prepare for recovery. To list all available database dumps, the `/opt/bacula/scripts/bacula-sybase.sh` script can be used. It simply queries a Bacula catalog and displays information in human readable format.

```
# /opt/bacula/scripts/bacula-sybase.sh
Updating cache ...
Searching catalog ... Jobs found.

Full dump:       D.pubs2.2018-11-19.14_11_07
  Transaction dump: T.pubs2.2018-11-19.14_11_12
  Transaction dump: T.pubs2.2018-11-19.14_11_15
  Transaction dump: T.pubs2.2018-11-19.14_11_18
 Incremental dump:   C.pubs2.2018-11-19.14_11_21
  Transaction dump: T.pubs2.2018-11-19.14_11_25
  Transaction dump: T.pubs2.2018-11-19.14_11_28
  Transaction dump: T.pubs2.2018-11-19.14_11_30
```

In the above example, the values like `D.pubs2.2018-11-19.14:11:07` need to be used for the `dump=...` option of a Sybase restore session. More details are available in the following sections.

The above information can be obtained manually by browsing the Bacula catalog with the `bconsole` or `BWeb` applications. In this case, the saved filenames like `D.pubs2.2018-11-19.14:11:07.000` are of interest, where the suffix `.000` shows the stripe number of a particular dump file saved.

When database dumps were performed using the `striped on 'sybacula::'` parameter of the `dump` command, then `bacula-sybase.sh` will show how many stripes were used for any particular dump, i.e.:

```
# /opt/bacula/scripts/bacula-sybase.sh
Updating cache ...
Searching catalog ... Jobs found.

Full dump:       D.pubs2.2018-11-19.14_46_47 - stripped on 2
  Transaction dump: T.pubs2.2018-11-19.14_46_55 - stripped on 2
  Transaction dump: T.pubs2.2018-11-19.14_46_58 - stripped on 2
  Transaction dump: T.pubs2.2018-11-19.14_47_01 - stripped on 2
 Incremental dump:   C.pubs2.2018-11-19.14_47_05 - stripped on 2
  Transaction dump: T.pubs2.2018-11-19.14_47_09 - stripped on 2
  Transaction dump: T.pubs2.2018-11-19.14_47_12 - stripped on 2
  Transaction dump: T.pubs2.2018-11-19.14_47_15 - stripped on 2
```

The string "`stripped on 2`"[1] above indicates that two "`stripe on 'sybacula::'`" parameters will be needed to restore. More information on stripes can be found at *Stripes*.

8.2 Restoring a Database from a Full Backup

To restore a database it is necessary to always start from a Full Database backup. A command along with the selection of the required database dumpsusing the `dump=...` parameter is the starting point. The list of dumps to load can be obtained with the `bacula-sybase.sh` script as described in section *Listing Database Dumps*. The command might be executed as follows:

```
1> load database pubs2 from 'sybacula::dump=D.pubs2.2018-11-20.18_09_50'
2> go
```

If the Full Database backup was performed with multiple streams, the correct `load database` command needs to be prepared as described in section *Stripes*. The additional `stripe on` parameters of this command should not use any `dump=...` options, as the correct dump filenames will be selected automatically. The same procedure applies for any other backup type.

After a load operation has finished, the Sybase ASE does not bring the database online automatically, allowing the administrator to load subsequent cumulative or transaction log backups. Thus, after a restore process has been finalized, it is necessary to bring the restored database online manually, which is done with the command:

```
1> online database pubs2
2> go
```

More information about the `load` and commands and their additional parameters can be found in the Sybase documentation in the *Reference Manual: Commands 16.0*, section *Commands*.

[1] The typo "stripped" instead of "striped" will be fixed in a later edition of this manual, and in the software itself.

8.3 Restoring a Database from a Differential Backup

Database restores of full backups can be completed by loading a subsequent differential backups. In most cases, the latest differential backup will be needed, as that consists of all changes recorded since the previous Full database dump. It is not required to restore any previous cumulative dumps – differential Backups in Baculas terminology – for successful recovery.

To execute a differential restore, the command to use is `load database cumulative` with the proper dump file in the `dump=...` parameter. Which file to use may be determined by using the `bacula-sybase.sh` script as described in section *Listing Database Dumps*. Thus, a command might be such as

```
1> load database pubs2 cumulative from 'sybacula::dump=C.pubs2.2018-11-20.18_15_38'
2> go
```

Again, if the differential – "cumulative incremental" in Sybase terminology – backup was performed with multiple streams, the correct `load database` command has to be prepared as described in *Stripes*. The additional `stripe on` parameters of this command should not use any `dump=...` options, as in this case the correct dump filenames will be determined automatically by the plugin.

To finalize the restore now, if no subsequent transaction log backups will be needed, the `online database` command is used to bring the restored database online.

```
1> online database pubs2
2> go
```

More information about the `load` and commands and their parameters is available in the Sybase documentation, in *Reference Manual: Commands 16.0*, section *Commands*.

8.4 Restoring Database Transactions and PITR

Database recovery can be continued after loading a full dump and a subsequent cumulative incremental backup (described above) by loading and applying required transaction logs. All available and required database transactions should be selected and loaded. Restoration to a distinct point in time (Point in Time Recovery or PITR) before the latest transaction logged is possible by following the procedure in section *Loading Transaction Logs to a Point In Time (PITR)*.

For each transaction log backup – incremental backup in Bacula's terminology – a separate `load transaction` command is required. This command needs to be executed with the required database dump in the `dump=...` parameter. Which transaction log backup to restore may be determined using the `bacula-sybase.sh` script as described in section *Listing Database Dumps*.

An example command would be:

```
1> load transaction pubs2 from 'sybacula::dump=T.pubs2.2018-11-21.10_18_10'
2> go
```

Note that transaction logs need to be applied in their correct order, namely, oldest first, and no logs can be skipped.

Also note that this step needs to be repeated until all required or available transaction dumps are loaded to the restored database.

If transaction backups were performed with multiple streams, the correct `load transaction` command needs to be prepared as described in section *Stripes*. The additional `stripe on` parameters of this `load` command should not use any `dump=...` options. In this case also, the correct dump filenames will be chosen automatically. This is the same procedure for any backup type.

After all available or required transaction dumps were loaded to the restored database, the `online database` command is used to bring this database online:

```
1> online database pubs2
2> go
```

For more information about the `load` and commands and their parameters, the Sybase documentation should be consulted: *Reference Manual: Commands 16.0*, section *Commands*.

Loading Transaction Logs to a Point In Time (PITR)

It is possible to recover a database up to a specified point in time (Point In Time Recovery) in its transaction logs. To do so, the `until_time` option of the `load transaction` command is used. This is useful if, for example, a user inadvertently drops an important table. In this case, the `until_time` option would be set to a time just before the table was dropped.

To use the `until_time` option effectively after data has been destroyed, it is necessary to know the exact time the error occurred. In `isql`, the current timestamp can be determined using the `select getdate` command immediately after the time of the error:

```
1> select convert(char(26), getdate(), 109)
2> go
 --------------------------
 Nov 22 2018  2:06:30:610PM
```

After backing up the transaction log containing the error and loading the most recent database dump, plus possible cumulative (or differential) backups, all transaction logs that were created after the database was last dumped will be restored. The final one, containing the error, will then be loaded with the `until_time` option:

```
1> load transaction pubs2 from "sybacula::dump=T.pubs2.2018-11-22.12_07_30"
2> with until_time = "Nov 22 2018  2:06:00:0PM"
3> go
```

Note that in this example, we assume the time stamp was taken very quickly after the error was caused.

After a transaction log is applied with the `until_time` option, Adaptive Server restarts the database's log sequence. This implies that, until the database is dumped again, subsequent transaction logs after the load transaction using `until_time` can not be applied. You **must** dump the database before another transaction log backup can be done.

8.5 Restoring System Databases

In order to `load` the master database, the Sybase ASE engine must run in **single-user** mode. Please see the Sybase documentation, *SAP Adaptive Server Enterprise 16.0*, part *System Administration Guide: Volume 2* for more information about restoring system databases and required procedures.

8.6 Restoring Dumps to a Local Directory

With the `sybase-sbt` plugin it is possible to restore any database backups to a directory for manual database restore and recovery procedures. This feature is invoked by setting a proper `where=...` restore parameter pointing to a local directory of your choice.

Note that these restores need to be started through Bacula's command interface, for example with `bconsole`:

```
* restore client=sybase-fd fileset=SBT before="2018-11-21 10:00:00" \
    where=/tmp/restores select all done
Using Catalog "MyCatalog"
Run Restore job
JobName:         RestoreFiles
Bootstrap:       /opt/bacula/working/sybase-dir.restore.5.bsr
Where:           /tmp/restores
Replace:         Always
FileSet:         Full Set
Backup Client:   sybase-fd
Restore Client:  sybase-fd
Storage:         File1
When:            2018-11-22 15:32:51
Catalog:         MyCatalog
Priority:        10
Plugin Options:  *None*
OK to run? (yes/mod/no):
```

Note that, if stripes were used for such backups, some of the striped dump files will not be restored using the above method. It is necessary to restore missing striped dump files using the `JobId=...` restore parameter instead of the `before=...` option.

9 More Information

9.1 Troubleshooting

Connection Errors

In case an error message like

```
Msg 4002, Level 14, State 1:
Server 'SYBASECENTOS':
Login failed.
CT-LIBRARY error:
    ct_connect(): protocol specific layer: external error: \
        The attempt to connect to the server failed.
```

is encountered using a script like `sybase-backup.sh`, the contents of the file `/opt/bacula/sybase/sybase.env`, which contains parameters used to connect to the Sybase dataserver, should be checked.

Of particular importance are the `SYBSERVER` and `SAPASSWORD` settings.

Tracing

Job tracing information can be generated using the following parameters to the `dump` or `load` commands:

- `trace=<file>` provides the name of a file to store tracing information from the `libsybacula.so` module in.
- `debug=<level>` to set up a specific debug level for tracing. The higher the level, the more detailed the tracing information will be.

If the a `trace=...` parameter is omitted, but the `debug` one is provided, the default trace file will be `sybacula.<PID>.trace` in the or `C:/Program Files/Bacula/sybase/` directory.

An example is

```
1> dump database pubs2 to 'sybacula::debug=100'
2> go
```

9.2 Limitations

- Database backup and restore of compressed data must occur on the same platform. This is a Sybase Backup Server limitation.
- Automatic restore of striped backups to a local directory could skip some striped dump files. Manual restore using the JobId parameter is required. This limitation does not apply when restoring to a Sybase ASE dataserver.
- No more than 32 streams, including the main one in the `to 'sybacula::'` clause, can be used. This is a Sybase ASE limitation.
- Job estimation is not supported. This limitation might be removed in the future.
- Plugin listing mode is not supported. This limitation might be removed in the future.

Index