Depository Databases

Depository databases

A depository database (depository) is a long-term archive, that provides time-unlimited archiving values in SQL databases. All values of an object with the option Depository enabled at its configuration, are stored in the depository database. Request for the storing into the depository database is generated immediately after storing into the archive database. So this storing and the storing into the archive database are executed almost in parallel. Unlike the archive database, data from the depository database is not available automatically in on-line mode and is never deleted. On-line accessing the data from depository databases is allowed after their mounting into the system.

Depository databases are available on Sybase, Oracle, and PostgreSQL platforms and their features depend on the possibilities of respective platforms. For the Microsoft SQL Server platform, resp. Microsoft Desktop Engine (MSDE) platform and depository databases are not implemented.

Note: To find out about the configuration of the depository database, use a LIST_TREZOR TELL command.

Depository databases on the Sybase platform

Configuration of the D2000 Archiv process contains a configured time period TrezorPeriod, during which the process writes data to one depository database. When this period elapses, the database is released and the system starts storing data in a new database. The released database is available for storing delayed data for a defined time (configuration parameter TrezorCompressTime). After this time, the database is compressed by Sybase tools and moved to a directory specified by the configuration parameter TrezorCompressPath. The name of the compressed database contains the end time of the repository database (e.g. *Trezor_2008_03_20_16.cdb*).

Note: Compression of Sybase databases is implemented by supported versions of Sybase Adaptive Server Anywhere 6 to 9. Starting with the version Sybase SQL Anywhere 12, the utility for database compression does not exist anymore and compressed databases are considered to be obsolete. Therefore if you use Sybase SQL Anywhere 12, the depository databases are not being compressed but they remain in the directory specified by configuration parameter TrezorCompressPath, as files *.db* and *.log* (e.g. *Trezor_2010_02_04_07.db* and *Trezor_2010_02_04_07.log*).

Premature releasing of the depository database can be performed using the RELEASE_TREZOR TELL command.

Off-line analysis of the data from the depository database is possible without performing any decompression. The support for the data import from the depository database into MS-EXCEL was created for the analysis. As the Sybase Adaptive Server Anywhere is a standard SQL server, any tool suitable for working with SQL databases can be used for the analysis.

Configuration of the parameters defining the method of depository database creation is implemented in the D2000 Application Manager process. They may be defined for each process D2000 Archiv.

Configuration parameters for the Sybase platform:

- **AutoMountPath** - path to a directory from which all the depositories will be mounted automatically after the start of the D2000 Archiv process
- **TrezorPath** - path to the directory where the depository databases are created
- **TrezorCompressPath** - path to the directory for storing the compressed depository databases
- **TrezorPeriod** - the period of the depository creation. It is specified in format *NX*, where *N* is an integer number and *X* defines a time unit. Permitted time units are:
  - H - hour
  - D - day
  - W - week
  - M - month.

  Note: Setting the parameter either to an empty value or to *0H* disables the functionality of depository databases.
- **TrezorCompressTime** - time defining the compression of the released depository database. The database is available for writing delayed data during this time. The value of TrezorCompressTime must be less than the value of TrezorPeriod.
- **TrezorCompressOffline** - value 1 (default) means that the compressed depository database will remain offline. Value 0 means that the compressed depository database will be re-connected and available for reading.

The parameters *TrezorPeriod* and *TrezorCompressTime* can be given in hours, days, weeks, or months. The syntax is e.g.:

- 4H - four hours
- 2D - two days
- 1W - one week
- 1M - one month

Note: It is possible to query the configuration of depository databases and currently mounted depository databases by Tell command LIST_TREZOR.

Warning: When a depository database is created or if it does not exist, the process D2000 Archiv of database templates copies the template of the depository database and initializes its structure. Because of this, it is not allowed to copy the templates of the depository database manually since the **D2000 Archiv** process does not initialize the depository database when it finds it in the **Trezor** subdirectory of the application directory.

Depository databases on the Oracle platform

The depository database is implemented as an independent tablespace named *APPLICATION*_TS_TREZOR*ID*, where ID=1,2.. etc (e.g. TEST_TS_TREZOR5). The tablespace contains datafiles - their number is defined by the parameter **TrezorCountDatafiles** (1, by default). Data files are generated in the **TrezorPath** directory.
If **TrezorCountDatafiles**=1, the data file name is *APPLICATION*_TS_TREZOR*xxID_YYYYMMDD*.ORA.
If **TrezorCountDatafiles**>1, the data file names are *APPLICATION*_TS_TREZOR*xxID*_01_*YYYYMMDD*.ORA, *APPLICATION*_TS_TREZOR*xxID*_02_*YYYYMMDD*.ORA etc, where *xxID* is ID enhanced from the left with a zero to the length of 4 characters and *YYYYMMDD* is the depository database creation date, e.g. *TEST_TS_TREZOR0005_03_20060930.ORA*. Datafile size is specified by the parameter **TrezorStartDatafileSize**, the parameter **TrezorMaxDatafileSize** allows specifying the maximum data file size.

Similarly to Sybase, the configuration of the process D2000 Archiv contains a configured time period TrezorPeriod, during which the process writes data to one depository database. When this period elapses, the database is released and the process starts storing values into a new database. The released database is available for storing delayed data for a given time (configuration parameter TrezorCompressTime). After the expiration of this time, the tablespace is first switched into read-only mode (ALTER TABLESPACE *APPLICATION*_TS_TREZOR*ID* READ ONLY) and then if the value of parameter TrezorCompressOffline=1, switched to offline mode (ALTER TABLESPACE *APPLICATION*_TS_TREZOR*ID* OFFLINE). If the parameters **TrezorPath** and **TrezorCompressPath** are defined, data files are to be moved to the **TrezorCompressPath** directory.
After releasing a depository database (and possible moving), it is possible to run an external program - its name and path to it are defined by the parameter TrezorPostCompressCmd and its parameters are defined by the parameter TrezorPostCompressPar.

The TELL command RELEASE_TREZOR allows releasing a depository database before time.
The process D2000 Archiv creates its internal list of depository databases - the **Trezors** table.

Note 1: When switching tablespace to read-only mode, Oracle requires no active transactions in the whole database. The D2000 Archiv process, therefore, commits the writes to the archive as well as to an active depository database. Therefore it is important for the database containing the tablespace not to be used for other purposes which would require transactions that could take more time because the depository parts of the archive will be blocked until the transactions are finished (see the**TrezorReadOnlyTimeout** parameter).

Note 2: Maximum length of tablespace name in Oracle is 30 characters. That limitation must be taken into account when defining an application name - the name *APPLICATION*_TS_TREZOR*ID* (or *APPLICATION*_TS_TREZOR*ID_seg* in use of depository database segments) cannot be longer than 30 characters.

Note 3: Oracle database can contain as many datafiles as defined by the parameter *DB_FILES* in the initialization file of the database. Therefore, setting up the value of the parameter to sufficient value before creating depository databases is enabled.

Depository database segments

Since the 7.01.10 version, the process D2000 Archiv supports the concept of depository database segments on the Oracle platform. Depository database segments is a depository database tablespace, several of which can be created and filled in parallel. Each historical value is stored in one segment, the number of which is specified by the parameter Depository segment in the historical value configuration.

The number of segments created by the process D2000 Archiv is specified by the parameter TrezorCountSegments in the Windows registry. Historical values with the parameter depository segment higher than the value of the parameter TrezorCountSegments are to be stored in zero depository segment (it is the segment that is also created when depository database segments are disabled).

The main purpose of depository database segments is the creation of several smaller depository databases so that reading one historical value for a long time interval requires less disk space (because only specific segments containing that historical value need to be mounted) - compared to mounting all non-segmented depository databases for a required time interval.
Having depository database segments enabled, the name of zero segment tablespace is *APPLICATION*_TS_TREZOR*ID* (identical with the name of the depository database tablespace if segments are disabled) and the names of all other segments are *APPLICATION*_TS_TREZOR*ID_seg*, where *seg* is a double-digit number of segment (between 01 and TrezorCountSegments).

If **TrezorCountDatafiles**=1, the names of data files are *APPLICATION*_TS_TREZOR*xxID_Syy_YYYYMMDD*.ORA
If **TrezorCountDatafiles**>1, the names of data files are *APPLICATION*_TS_TREZOR*xxID_Syy_zz_YYYYMMDD*.ORA, where *xxID* is ID enhanced from the left with zeros to the length of 4 characters, *yy* is the segment number enhanced from the left with a zero to the length of 2 characters, *z* is the datafile number enhanced from the left with a zero to the length of 2 characters and *YYYYMMDD* is the depository database creation time. Example: TEST_TS_TREZOR0005_S02_01_20060930.ORA

Note: Starting with D2000 version 7.02.010, there is possible to specify a different suffix of datafile than *ORA* by using the parameter TrezorDatafileSuffix.

Note: Default prefix of depository tablespaces and datafiles *APPLICATION*_TS_TREZOR can be changed using the parameter TrezorPrefix.

Configuration parameters for the Oracle platform:

All the configuration parameters in the Windows Registry are of *String* type, besides the parameters *TrezorCountDatafiles*, *TrezorCountSegments*, *TrezorCountDatafiles,* and *TrezorNoLogging* of DWORD type.

- **TrezorPath** - path to the directory, the data files of tablespaces are created in. You must also enter the character "\" after the directory name. If the parameter is not given, data files are created in the database directory.
- **TrezorCompressPath** - path to the directory, the data files of tablespaces released are being moved to. You must also enter the character "\" after the directory name. Once the *TrezorCompressPath* parameter is defined, you must also define *TrezorPath*, otherwise, data files are not to be moved.
- **TrezorPeriod** - the period of the depository creation. It is specified in format *NX*, where *N* is an integer number and *X* defines a time unit. Permitted time units are:
    - H - hour

    - D - day
    - W - week
    - M - month.

    Note: Setting the parameter either to an empty value or to *0H* disables the functionality of depository databases.
- **TrezorCompressOffline** - value 1 (default) means that the compressed depository database will be put into offline mode. Value 0 means that the depository database remains in read-only mode and data will be available for reading. Value 2 means that D2000 Archiv will not access the depository database while TrezorPostCompressCmd is running, so this command may perform various maintenance tasks which could otherwise block the archive.
- **TrezorCompressTime** - time for releasing depository tablespace. The tablespace is available for writing delayed data during this period. *TrezorCompressTime* must be less than *TrezorPeriod*.
- **TrezorCountSegments** - number of depository database segments (depository database tablespaces), that are being created in parallel. The default value is 0 (only one segment 0 is being created), the maximum value is 99 (segments 0 to 99 are being created). The parameter is of DWORD type !!!.
- **TrezorCountDatafiles** - the number of data files in the tablespace, by default, is 1 (the parameter is of DWORD type !!!).
- **TrezorStartDatafileSize** - the size of data file that is to be created (in Oracle syntax, e.g. 500K, 300M). The parameter must be specified.
  Note: For OpenVMS+Oracle 9.2.0.x platform, there is a problem with data files damaged at the size of 4GB - must be tested prior to using!
- **TrezorMaxDatafileSize** - maximum size of data file (in Oracle syntax, e.g. 500K, 300M or UNLIMITED). If it is not defined, the size of data file is specified at its creation (see the note for the previous parameter).
- **TrezorDatafileSizeStep** - the size of data file increment (in Oracle syntax, e.g. 200K, 1M) during the growth of the datafile (ON NEXT clause of the CREATE TABLESPACE command). If the parameter is not defined, the ON NEXT clause is not used and the increment is the database default value (8K, by default).
- **TrezorDatafileSuffix** - suffix of data file. The default value of this parameter is *ORA*.
- **TrezorPrefix** - if this parameter is not set, the prefix of depository tablespaces and datafiles is *APPLICATION*_TS_TREZOR.
  Setting the value of TrezorPrefix changes this prefix. The parameter can be used together with the DbUsername parameter when doing a migration of the archive database from one application to another (or when renaming an application) and keeping the archive tablespace and existing depositories (i.e. setting the value of TrezorPrefix to *OldApp_TS_TREZOR*).
- **TrezorReadOnlyTimeout** - specifies the timeout (in seconds) for switching the depository database into READ ONLY mode. After expiration, the archive generates the system alarm *"Changing trezor tablespace read-only takes more than 120 sec, possible Oracle lock!"*. The default value is 120 seconds.
- **TrezorReadSegment0** - Parameter has a meaning only if depository segments are configured (non-zero value of TrezorCountSegments parameter). Activation of a parameter (value 1) instructs the archive to read also from depository segment 0 when reading from any other depository segment. Reading from depository databases will be slower, but it enables reconfiguring historical values from default depository segment 0 to a different segment while keeping access to data previously stored in depository segment 0.
  The TrezorReadSegment0 parameter can be changed by the **SET_OPTION TELL** command.
- **TrezorReadSinceCreate** - value 1 means that depositories that are older than Create Time of historical value will not be read. The TrezorReadSinceCreate parameter can be changed by the **SET_OPTION TELL** command.
- **TrezorPostCompressCmd** - the name of a program to be run after releasing the depository database and possible moving of data files of the depository database released. The program may be used e.g. to pack data files and copy them within the network.
- **TrezorPostCompressPar** - optional parameter, used for the program specified by the parameter *TrezorPostCompressCmd*. It can contain optional text and predefined macros to be replaced before running the program:
    - **#ID#** - is replaced by depository database Id (e.g. 15).
    - **#TREZOR#** - is replaced by the depository tablespace name (e.g. TEST_TS_TREZOR5).
    - **#SEGMENTS#** - is replaced by the number of depository database segments (i.e. the value of the parameter *TrezorCountSegments*).
    - **#FILES#** - is replaced by the number of data files of the depository database (i.e. the value of the parameter *TrezorCountDatafiles*).
    - **#FILE1#**, **#FILE2#,** etc. - is replaced by the name of the corresponding data file along with the path, if the path is defined by the parameter *TrezorPath* (e.g. *C:\ora920\oradata\D2000\TEST_TS_TREZOR5_1.ORA*).

  Example 1: There is enabled the creating of depository databases with one data file to be packed into the file *TrezorID.zip* (ID=1,2 etc.) and moved to the directory *D:\backup*
  TrezorCountDatafiles 1
  TrezorPostCompressCmd C:\utils\zip.exe
  TrezorPostCompressPar -m D:\backup\Trezor#ID#.zip #FILE1#

  After releasing for example the depository database nr. 5, a program can be run:
  C:\utils\zip.exe -m D:\backup\Trezor5.zip C:\ora920\oradata\D2000\TEST_TS_TREZOR5_1.ORA

  Example 2: Configuration, which will remount (using TELL command) the depository tablespace immediately after it is released by:
  TrezorPostCompressCmd c:\D2000\D2000.E70\bin\tell.exe
  TrezorPostCompressPar dst=SELF.ARC cmd="MOUNT_TREZOR #ID#" uid=myuser pwd=mypassword
- **TrezorNoLogging** - the parameter of *DWORD* type, its non-zero value causes the depository tablespace to be created in the NOLOGGING mode. If its value is zero or doesn't exist, the depository tablespace is to be created in the LOGGING mode (it generates REDO logs and is restorable).

Depository databases outside the archive database on the Oracle platform

Oracle-based archives support creating the depository databases in a different database than the archive is in. The TNS of the depository database is defined by the configuration parameter **TNS_Service_Name_Trezor**. The user name and password are the same as in the archive database.
Parameter **TNS_Service_Name_Trezor** is located in the registry in a branch belonging to the application and archive, e.g.
HKEY_LOCAL_MACHINE\Software\Ipesoft\D2000V70\cfg_test\SELF.ARC
The database, in which the depository will be created, must contain the archive tablespace (created in the same way as if the archive was supposed to be located in this database) and appropriate archive user *application_name*_archiv, e.g. test_archiv.

Common depository database for more archives on the Oracle platform

Oracle-based archives support the configuration where two or more shadow archives work with a common depository database. Only an active instance of the archive uses the depository database. The common depository database is defined by the setting of the configuration parameter **Trezor_Active_Only** (of REG_DWORD type) to a non-zero value. This parameter is located in the registry in a branch belonging to the application and archive. As the archives are shadow archives, the branch of the archive contains also instance numbers, generally HKEY_LOCAL_MACHINE\Software\Ipesoft\D2000V70\cfg_**APPLICATION_NAME\ARCHIVE_NAME**.ARC_*INSTANCE_NUMBER,* e.g. HKEY_LOCAL_MACHINE\Software\Ipesoft\D2000V70\cfg_test\SELF.ARC_2\)

Note 1: Parameter **Trezor_Active_Only** must be configured on all instances of the shadow archive, otherwise the archives will block each other from trying to write the same value to the depository database.

Depository databases on the PostgreSQL platform

A depository database is implemented as an independent database. Names of depository databases are defined by configuration parameter PG_TrezorName0.
Similarly to Sybase, the configuration of the process D2000 Archiv contains a configured time period TrezorPeriod, during which the process writes data to one depository database. When this period elapses, the database is released and the process starts storing values into a new database. The released database is available for storing delayed data for a given time (configuration parameter TrezorCompressTime). After expiration of this time, a default read-only access is configured for the database (ALTER DATABASE *APLIKACIA*_TREZOR_*#ID#* SET default_transaction_read_only = true) and then if value of parameter TrezorCompressOffline=1, access to depository database is forbidden (update pg_database set datallowconn = false where datname = '*APLIKACIA*_TREZOR_*#ID#*'). After disconnecting a depository database, it is possible to run an external program - its name and path to it are defined by the parameter TrezorPostCompressCmd and its parameters are defined by the parameter TrezorPostCompressPar.
This program can for example move and compress a disconnected depository database or create a dump of read-only depository database via PostgreSQL utility *pg_dump*.
Note 1: In order for a database user *dba* (used by D2000 Archiv to connect to PostgreSQL server), to create and disconnect the databases, it is recommended to grant the *dba* user a superuser privilege (ALTER ROLE dba WITH SUPERUSER;).
Note 2: As every depository database (resp. every depository database segment) is a separate database, which is accessed by the D2000 Archiv via a separate database connection, it is necessary to configure the PostgreSQL server to permit a sufficient number of connections (parameter *max_connections*), for D2000 Archiv to open a connection to all mounted depository databases.

Depository database segments

Depository database on the PostgreSQL platform supports also depository database segments (similarly to the Oracle platform). The depository database segment is a separate depository database, several of which can be created and filled in parallel. Each historical value is stored in one segment, the number of which is specified by the parameter Depository segment in the historical value configuration.

The number of segments created by the D2000 Archiv process is specified by the parameter TrezorCountSegments in the Windows registry. Historical values with the parameter depository segment higher than the value of the parameter TrezorCountSegments are to be stored in zero depository segment (it is the segment that is also created when depository database segments are disabled).

The name of the database representing depository segment 0 is specified by configuration parameter PG_TrezorName0, and names of databases representing depository segments 1..N are specified by configuration parameter PG_TrezorName.

The main purpose of depository database segments is the creation of several smaller depository databases so that reading one historical value for a long time interval requires less disk space (because only specific segments containing that historical value need to be mounted) - compared to mounting all non-segmented depository databases for a required time interval.

Configuration parameters for PostgreSQL platform:

- **PG_CreateTrezor** - SQL command for creating a depository database. Default value is
  CREATE DATABASE "*#TREZOR#*" WITH ENCODING='UTF8' OWNER=dba TABLESPACE="D2000" TEMPLATE=template0
  where *#TREZOR#* is the name of the depository database (defined by configuration parameter PG_TrezorName0 resp. for depository database segments 1..N by parameter PG_TrezorName). By default existence of tablespace named *D2000* and template database *template0* is required in the PostgreSQL database server.
- **PG_TrezorFileMulti** - parameter is active if also the PG_TrezorFilePath parameter is specified. The parameter defines a multiplicator for the CommitCount parameter. The default value of this parameter is 10.
- **PG_TrezorFilePath** - parameter can be used for PostgreSQL 9.5 and above. It activates writing to depository databases via files, which is 2 to 3-times faster than the standard batch insert via the ODBC interface. After PG_TrezorFileMulti * CommitCount values are accumulated, they are stored into a file named archiv_<*TrezorId*>_<*SegmentId*>.txt (e.g. archiv_4_1.txt) in directory PG_TrezorFilePath and an UPSERT into a depository database is performed, using a foreign table mapped onto this file (using PostgreSQL extension file_fdw) as a source of data.
  The directory *PG_TrezorFilePath* must be available both for reading and writing to the archive as well as to the PostgreSQL database, therefore this parameter can be used if both archive and database are on the same computer.
- **PG_TrezorName0** - mask for the name of created depository databases. If depository database segments are enabled, this mask will be used for segment 0. The default value is *appname*_TREZOR_*#ID#*, where *appname* is the application's name and *#ID#* is replaced by the depository database number.
  E.g. for the application named *Test,* depository databases Test_TREZOR_1, Test_TREZOR_2, Test_TREZOR_3, etc will be created.
  The specified mask must contain text *#ID#*.
  Note: If the default value is not acceptable, the desired value must be configured before enabling depository databases. Should the value of this parameter be changed after some depository databases have been created, it is necessary to rename all existing depository databases according to the newly specified mask.
- **PG_TrezorName** - mask for the name of created databases - depository database segments 1..N.
  Parameter is used only when TrezorCountSegments > 0. The default value is je *appname*_TREZOR_*#ID#*_*#SEG#*, where *appname* is the application's name, *#ID#* is replaced by the depository database number and *#SEG#* is replaced by the segment number.
  E.g. for an application named *Test* with a number of segments equal to 1, databases Test_TREZOR_1, Test_TREZOR_1_1, Test_TREZOR_2, Test_TREZOR_2_2, Test_TREZOR_3, Test_TREZOR_3_1, etc will be created.

The specified mask must contain text *#ID#* and *#SEG#*.
Note: If the default value is not acceptable, the desired value must be configured before enabling depository databases. Should the value of this parameter be changed after some depository databases have been created, it is necessary to rename all existing depository databases according to the newly specified mask.

- **TrezorPeriod** - the period of the depository creation. It is specified in format *NX*, where *N* is an integer number and *X* defines a time unit. Permitted time units are:
  - H - hour
  - D - day
  - W - week
  - M - month.
    Note: Setting the parameter either to an empty value or to *0H* disables the functionality of depository databases.

- **TrezorReadSegment0** - Parameter has a meaning only if depository segments are configured (non-zero value of TrezorCountSegments parameter). Activation of a parameter (value 1) instructs the archive to read also from depository segment 0 when reading from any other depository segment. Reading from depository databases will be slower, but it enables reconfiguring historical values from default depository segment 0 to a different segment while keeping access to data previously stored in depository segment 0.
  The TrezorReadSegment0 parameter can be changed by the **SET_OPTION TELL** command.
- **TrezorReadSinceCreate** - value 1 means that depositories that are older than Create Time of historical value will not be read. The TrezorReadSinceCreate parameter can be changed by the **SET_OPTION TELL** command.
- **TrezorCompressOffline** - value 1 (default) means that the compressed depository database will be put into offline mode. Value 0 means that the depository database remains in read-only mode and data will be available for reading. Value 2 means that D2000 Archiv will not access the depository database while TrezorPostCompressCmd is running, so this command may perform various maintenance tasks which could otherwise block the archive.
- **TrezorCompressTime** - time for releasing the depository database. The database is available for writing delayed data during this period. *TrezorCompressTime* must be less than *TrezorPeriod*.
- **TrezorCountSegments** - number of depository database segments (depository databases), that are being created in parallel. The default value is 0 (only segment 0 is being created), and the maximum value is 99 (segments 0 to 99 are being created). The parameter is of DWORD type !!!.
- **TrezorPostCompressCmd** - the name of a program to be run after releasing the depository database. This program can for example move and compress a disconnected depository database or create a dump of read-only depository database via PostgreSQL utility *pg_dump*.
- **TrezorPostCompressPar** - optional parameter, used for the program specified by the parameter *TrezorPostCompressCmd*. It can contain optional text and predefined macros to be replaced before running the program:
  - **#ID#** - is replaced by depository database Id (e.g. 5).
  - **#OID#** - is replaced by OID (object identifier) of the depository database (OID is related to the name of the directory containing the respective database inside PostgreSQL tablespace).
  - **#OID0#**, **#OID1#**, **#OID2#** etc - is replaced by OIDs (object identifiers) of databases representing respective depository database segment 0, 1, 2 etc.
  - **#TREZOR#** - is replaced by the depository database name (e.g. Test_TREZOR_5).
  - **#TREZOR0#**, **#TREZOR1#**, **#TREZOR2#,** etc - is replaced by the database name representing the respective depository database segments.

An example of a batch file used to perform a depository database maintenance and export as well as the export of *trezors* table located in archive database *MyApp.Archiv*. The batch file requires as a parameter the name of the depository database, which can be provided by setting the **Trezor PostCompressPar** parameter to value **#TREZOR#**.

```
rem Target directory for exports
set MyDir=D:\Trezors_export\
rem Set PGPASSWORD to password assigned to postgres user during installation
set PGPASSWORD=MyPostgresPassword
set PATH=%PATH%;c:\Program Files\PostgreSQL\9.5\bin
rem export of table trezors from the archive database
pg_dump -Fc -U postgres -f "%MyDir%\MyApp_arc_trezors.dmp" --table \"trezors\" MyApp.Archiv >> %MyDir%\%1.
log
rem permit write access to depository database and cluster the data table
echo alter database "%1" set default_transaction_read_only=false | psql -S -U postgres MyApp.Archiv >> %
MyDir%\%1.log
echo alter table data cluster on ix_data_rc | psql -S -U postgres %1 >> %MyDir%\%1.log
echo cluster data | psql -S -U postgres %1 >> %MyDir%\%1.log
rem set access to depository database back to read only
echo alter database "%1" set default_transaction_read_only=true | psql -S -U postgres MyApp.Archiv >> %
MyDir%\%1.log
pg_dump -Fc -U postgres -f "%MyDir%\%1.dmp" %1 >> %MyDir%\%1.log
```

Example of an equivalent batch file for Linux:

```
#!/usr/bin/env bash
MyDir=/trezorbackup
MyArc=mes_tpd_archive_self
MyLog=$MyDir/$1.log

#path to pg_dump, psql etc
PATH=/usr/pgsql-11/bin:$PATH

#password for dba user
export PGPASSWORD=***

#export of table trezors from the archive database
pg_dump -Fc -U dba -f "$MyDir/arc_trezors.dmp" --table \"trezors\" $MyArc >> $MyLog

#permit write access to depository database and cluster the data table
echo alter database \"$1\" set default_transaction_read_only=false | psql -S -U dba $MyArc >> $MyLog
echo alter table data cluster on ix_data_rc | psql -S -U dba "$1" >> $MyLog
echo cluster data | psql -S -U dba "$1" >> $MyLog

#set access to depository database back to read only
echo alter database \"$1\" set default_transaction_read_only=true | psql -S -U dba $MyArc >> $MyLog
#execute dump of depository database
pg_dump -Fc -U dba -f "$MyDir/$1.dmp" "$1" >> $MyLog
```