Adabas for Linux, UNIX and Windows

Database Monitoring and Tuning

Version 6.7

October 2018

ADABAS & NATURAL

This document applies to Adabas for Linux, UNIX and Windows Version 6.7 and all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 1987-2018 Software AG, Darmstadt, Germany and/or Software AG USA, Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA, Inc. and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at http://softwareag.com/licenses.

Use of this software is subject to adherence to Software AG's licensing conditions and terms. These terms are part of the product documentation, located at http://softwareag.com/licenses/ and/or in the root installation directory of the licensed product(s).

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third-Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software AG Products / Copyright and Trademark Notices of Software AG Products". These documents are part of the product documentation, located at http://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software AG.

Document ID: ADAOS-DBMON-67-20211006

Table of Contents

Database Monitoring and Tuning

This document contains information about database performance, monitoring and tuning.

The following topic is covered:

1 About this Documentation

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
`Monospace font`	Identifies service names and locations in the format *`folder.subfolder.service`*, APIs, Java classes, methods, properties.
`Italic`	Identifies: Variables for which you must supply values specific to your own situation or environment. New terms the first time they occur in the text. References to other documentation sources.
`Monospace font`	Identifies: Text you must type in. Messages displayed by the system. Program code.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
|	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the | symbol.
[ ]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information and Support

Software AG Documentation Website

You can find documentation on the Software AG Documentation website at **https://documentation.softwareag.com**.

Software AG Empower Product Support Website

If you do not yet have an account for Empower, send an email to empower@softwareag.com with your name, company, and company email address and request an account.

Once you have an account, you can open Support Incidents online via the eService section of Empower at **https://empower.softwareag.com/**.

You can find product information on the Software AG Empower Product Support website at **https://empower.softwareag.com**.

To submit feature/enhancement requests, get information about product availability, and download products, go to **Products**.

To get information about fixes and to read early warnings, technical papers, and knowledge base articles, go to the **Knowledge Center**.

If you have any questions, you can find a local or toll-free number for your country in our Global Support Contact Directory at **https://empower.softwareag.com/public_directory.aspx** and give us a call.

Software AG Tech Community

You can find documentation and other technical information on the Software AG Tech Community website at **https://techcommunity.softwareag.com**. You can:

- Access product documentation, if you have Tech Community credentials. If you do not, you will need to register and specify "Documentation" as an area of interest.
- Access articles, code samples, demos, and tutorials.
- Use the online discussion forums, moderated by Software AG professionals, to ask questions, discuss best practices, and learn how other customers are using Software AG technology.
- Link to external websites that discuss open standards and web technology.

Data Protection

Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

2 Database Monitoring And Tuning

Buffer Pool Manager

Buffer Pool Overview

A buffer pool manger is an important feature in database systems, introduced to improve performance.

If a database block is required, it is not directly read from disk, but it is requested from the buffer pool manager. The buffer pool manager keeps database blocks in memory - the buffer pool. If a database block is requested, the buffer pool manager checks whether the requested block is already in the buffer pool. If yes, it is not necessary to read it again from disk, and it can directly be returned to the caller. Only if the block is not yet in the buffer pool, does it have to be read from disk. Because typically most requested database blocks are used relatively often, this way, the number of physical disk I/Os can be reduced dramatically. A buffer pool efficiency defined as (1 - physical I/Os / logical I/Os) >= 99% is often achieved.

Adabas also uses the buffer pool for storing temporary blocks (NUCTMP, NUCSRT). These blocks are only relevant for the current nucleus session and for allocating internal work areas.

Buffer Flushes

Not only the number of read I/Os is reduced with the buffer pool, also the number of write I/Os: if database blocks are logically written to the buffer pool, they are not written directly to disk, but only if a certain number of blocks is marked as modified - then a “buffer flush” is performed. Because the same database block is often updated several times in quick succession, this way the number of write I/Os is reduced. Nevertheless, it must be guaranteed that no transactions are lost in case of a nucleus crash, for example because of a power failure. This is achieved via a log, that is required anyway in order to be able to roll back transactions. In Adabas, the WORK container is used for this purpose.

Buffer flushes are normally performed in parallel to the normal processing of the Adabas nucleus by a separate buffer flush thread. Therefore, the impact of a buffer flush usually is not very big, but the following performance reductions are possible:

- Because of the high I/O load generated by a buffer flush, other I/Os performed by the nucleus may become slower. As a result, commands requiring I/Os may become slower.
- I/Os may also result in additional CPU load, for example for copying the data to I/O buffers or for encrypting the data, if data is written encrypted to disk.

The amount of performance degradation caused by buffer flushes is very much system dependent; in some environments it may be nearly zero, while in other environments it may be noticeable or even disturbing.

Usually you want to keep the amount of modified database blocks not yet written to disk relatively small in order to keep the time small that is required for the regeneration of these blocks from the log on the WORK. In contrast, temporary blocks are relevant only for the current Adabas nucleus session and therefore need not be regenerated during a nucleus crash and should only be written to disk in order to avoid a buffer overflow.

For this reason, we have defined two separate flush lists (lists of blocks to be written to disk): one for database blocks and one for temporary blocks. These flush lists also result in separate buffer flushes for database blocks and for temporary blocks. If you have enough memory available, you should use a buffer pool size that temporary blocks need not be written to disk at all.

There are the following reasons for performing buffer flushes:

- Explicit buffer flushes: in some situations, Adabas explicitly starts a buffer flush, for example the nucleus performs an explicit buffer flush after it has created the checkpoint entry during the nucleus start, or during shutdown processing. Some utilities initiate an explicit buffer flush because they read Adabas files directly from disk, and therefore all modified database blocks of these files in the buffer pool must be written to disk before the utility can start processing.
- The write limit is exceeded. There are different write limits for database blocks and for temporary blocks: The write limit for database blocks can be defined via the nucleus parameter WRITE_LIMIT; for temporary blocks, the write limit is 50%.
- WORK limit buffer flush: blocks not yet written to disk must be regenerated via the log records stored on the WORK. If too many WORK blocks are required for the regeneration of these blocks, a buffer flush is started to avoid a WORK overflow.
- Wait-for-space buffer flush: there is no more free space in the buffer pool to allocate another block. If this happens, the command processing waits until the buffer flush is finished. Because this can result in large command execution times, this should be avoided: reducing the write limit or increasing the buffer pool, if enough memory is available.
- Emergency buffer flush: if the WORK really overflows, the command processing stops until the buffer flush is finished. Because this can result in large command execution times, this should be avoided: reducing the write limit or increasing the WORK may help.
- Ignored blocks buffer flush: When a modified block is being updated again just when the buffer flush tries to write the block to disk, the buffer flush ignores this block and continues with the next block to be written. However, in order to avoid emergency buffer flushes, directly after the termination of the buffer flush, another buffer flush is initiated to get these ignored blocks written to disk as well.

Autorestart processing

An important issue related to the buffer pool management is the autorestart processing: An autorestart is performed during nucleus start after an abnormal termination of the nucleus, for example because of a power failure. During the autorestart, the blocks that were in the buffer pool at the abnormal nucleus termination, but not yet written to disk, are regenerated and all open transactions are rolled back. For production database usually the maximum autorestart times should be kept small, because the autorestart times mean production downtime. The following is relevant for the autorestart time:

- The write limit for database blocks (nucleus parameter WRITE_LIMIT). The higher the write limit, the more modified blocks may be in the buffer pool that must be regenerated during a buffer flush. This leads to longer autorestart times.
- For some applications, the size of the WORK container is important. The WORK is used to log the database modification commands for the autorestart processing. If the same database blocks are updated very frequently, it can happen that a large number of WORK blocks are written to log the update operations. The more WORK blocks are written before a buffer flush occurs, the more update operations must be regenerated. This leads to longer autorestart times. In such a case, you can reduce the autorestart times by using a smaller WORK, because then a Work autorestart is performed, before the write limit triggers a buffer flush.

Buffer Pool Configuration

You can imagine that the configuration of the buffer pool has a large impact on the overall database performance. There are the following Adabas nucleus parameters for this purpose:

- BFIO_PARALLEL_LIMIT - Buffer flush I/O parallel limit. This parameter limits the number of I/Os that are started in parallel. If this value is too high, a read I/O done in parallel to the buffer flush, may become very slow, because before the read operation, at first the write I/Os are performed. The optimal value depends very much from the I/O optimizations done by the storage system and the operation system, but in most cases the default value of 50 will result in a good performance. This parameter can be changed dynamically with the utility ADAOPR while the nucleus is running.
- LBP - Length of buffer pool: You should try to define the buffer pool large enough, that you achieve a buffer pool efficiency >= 99%. It is also desirable that the buffer pool has a size where no buffer flushes for temporary blocks are performed. On the other hand, the optimal buffer pool size is limited by the available memory; LBP must not be so large that the operating system starts paging.
- READ_PARALLEL_LIMITS allows you to read ahead database blocks for sequential processing. This parameter can be changed dynamically with the utility ADAOPR while the nucleus is running. However, this increases the performance only if most of the Adabas commands process the database sequentially, for example in batch programs. Otherwise, you read blocks that are never used. Therefore it is recommended to switch this parameter on only explicitly when you run programs that really take advantage of the reading ahead.

- WRITE_LIMIT. This parameter specifies how many database blocks are modified until a buffer flush is started:
  - A larger value will usually reduce the number of write I/Os: Assume a block is updated twice. With the smaller WRITE_LIMIT, it may happen that the block is already written after the first update and before the second update, while with the larger WRITE_LIMIT, it takes longer
  - On the other hand, a larger value of WRITE_LIMIT also means that in case of a nucleus crash, for example after a power failure, there may be more modified blocks in the buffer pool that have not yet been written to disk. These blocks must be recreated in the "autorestart" performed when the database is restarted again. This means tt takes longer until the database is up again after the failure.
  - A smaller value for WRITE_LIMIT may be useful to avoid wait-for-space and emergency buffer flushes. Then a buffer flush may be triggered earlier, and modified blocks can be written to disk before they could cause a wait-for-space or emergency buffer flush.

WORK Size Consideration

Because the WORK container is used to log the database modifications and is required for an autorestart, also the WORK size is important for the buffer pool management. Please consider the following when you define the WORK size of a database:

- The WORK must be large enough to log all open transactions. If it is too small, transactions are rolled back, and an Adabas response code 9, subcode 3, "LP" is returned. If this happens too often, and not only for transactions that are open for a long time, you should increase the WORK.
- A lot of log space on WORK is required in particular if you use the following features:
  - Large object (LOB) values. For the insert or delete of a LOB value, the additional WORK space required is the size of the LOB value plus some administration data; for an update of a LOB value you need the double amount of space.
  - Adabas-to-Adabas replication. All database modifications are logged in Adabas files, the replication system files, and the modifications in the replication system files are also logged on WORK. The WORK space requirements with replication may be up to a factor 4 compared to the WORK requirements without replication.
- If the same database blocks are updated very often, it may happen that a buffer flush must be triggered to avoid a WORK overflow, before the write limit is exceeded. If this happens with a large WORK container, the WORK size may be a significant factor for the autorestart time, and it may be useful to reduce the WORK size. However, you should not reduce the WORK size so much that transactions must be rolled back.
- If a WORK buffer flush is triggered, and the update load for the database is high it can happen that an emergency buffer flush becomes necessary, because the database blocks are not written to disk fast enough. This should be avoided, because this can significantly reduce the Adabas performance. Either a larger WORK or a lower write limit may help.

Buffer Pool Fragmentation

When you define a database container, the block size can be any multiple of 1 KB up to 32 KB; NUCTMP blocks generally have a block size of 4 KB.

However, it is recommended to use only block sizes that are multiples of each other, in particular it is recommended to use block sizes that are powers of 2. Otherwise the buffer pool may become fragmented. It may happen that you cannot use the complete space in the buffer pool.

Monitoring the Buffer Pool Behaviour

For monitoring the buffer pool behavior (and generally the database behavior), you can use:

- The ADAOPR DISPLAY command, in particular DISPLAY=BP_STATISTICS and DISPLAY=BF_STATISTICS.
- The ADAMON utility.

For more information see the Utilities documentation.

Monitoring Usage of Resources

It is the responsibility of the DBA to monitor the database environment on a continuing basis in order to ensure that an efficient level of service is provided while maintaining database integrity. This responsibility for monitoring takes the form of a variety of activities and procedures, some of which are covered in this section.

The responsibility of the DBA to maintain database performance at an acceptable level is critical and difficult. The sources of degradation of service are numerous and often difficult to trace, while the process of making adjustments can be complex.

The DBA must implement a set of procedures which are designed to foresee degradation before the event and to adjust the operation or design of the database in an orderly and controlled way. This set of procedures will include the following activities:

- Identifying potential sources of degradation;
- Establishing tools for monitoring database performance;
- Controlling the implementation of adjustments;
- Controlling the implementation or redesign of the database to meet the new requirements.

Reporting on Usage

The DBA should make regular reports on database usage and performance to both data processing and user management. These reports should be as factual as possible, but also include recommendations for the tuning of the database environment as he develops them. It should be remembered that tuning, while benefiting the organization as a whole, may adversely affect the service received by one or more users. Any decision on tuning should, therefore, be taken by all affected users.

Monitoring Database Control

The DBA should establish appropriate controls and monitor them to assist him in ensuring the integrity of the database.

Computer-generated control totals can be checked and cross-footed between computer processing runs or generated reports. Batch responses (or inquiries) may include such information as the exact run time, search parameters, time of last update of data, and the primary parameter controls. This increases the confidence level and helps to ensure the integrity of the database.

The problem of control totals is one which will take on many different guises at different installations. It is not possible to lay down hard and fast rules in this area. However, it is felt appropriate to give some general guidelines.

The DBA should ensure that proper consideration is given to the following areas in the design of each application system which will use the database:

- What controls can be checked on every batch update run? For example, record counts, additions, deletions, updates;
- What controls require a full file pass to check them? For example, value field hash totals;
- What input transactions, Adabas logs, etc. should be retained in order to be able to recover when control totals are found to be wrong at the end of a given period;
- Are 'localized' control totals (e.g., by branch, product group) of any use in identifying the areas affected by a file control total error?

Performance Management, Statistics, Tuning

The following table illustrates some of the monitoring statistics that may be used and what adjustments to (or tuning of) the database environment may result.

Changes in	May require tuning of ...				
	Database Structure	Access Method Used	Hardware or Software Configuration	Processing Priority	Disk Storage Allocation
Terminal and line traffic		X	X	X	X
Response times (application performance)	X	X	X	X	X
Access totals by user and descriptor	X	X			X
Database size	X	X	X		X
Database growth rate	X	X	X		X

When any alteration is made to a production database, great care must be taken to ensure a continued high level of reliability and integrity. Whatever the change, the DBA must make sure that the decision is the right one and that it is properly and accurately implemented. He should retain absolute control over the tuning process and ensure that it follows the formal acceptance procedures.

The DBA must be very careful not to over-react to changes in the items listed in the table. A sudden change in line traffic, response times, etc., may only be a temporary affair. It is far better to wait for a while to see whether this is a permanent trend or a temporary disturbance to the normal way of operating. Another way of viewing the table is that the tuning required may be necessary when a new project will cause a significant change in terminal and line traffic, response times, etc. The DBA can then act in advance to minimize these effects before the new application system is implemented.

Command Logging

The Adabas command log option of the Adabas nucleus may be used to generate information on all the commands issued by users to Adabas.

Some of the information provided is:

- User identification
- Time of day
- What command was used

- Which file was accessed
- Which record was accessed
- What Adabas response code was received
- How long the command took to perform

This information provides a further level of knowledge of the activity against the database. It can be displayed using the utility ADACLP.

showcls (UNIX only)

showcls is a tool that analyses the client queue semaphore of a database. It is called with the following syntax:

```
showcls <dbid>
```

showcls displays the status of each client queue element of the database:

Status	Meaning
Free	The client queue entry can be used for a new client connection.
Obsolete	The client queue entry can be reused after the next client queue cleanup.
In use	The client queue entry is currently in use.

showipc (UNIX only)

showipc is a tool that the DBA can use in order to look at the UNIX kernel ipc configuration, and to look at the Adabas/NET-WORK ipc structures and clean them up. For each database, showipc displays the shared memory, the message queues, and the semaphores associated with the database in question.

The following options are available:

```
showipc [-adfhiksv] [-e<string>] [-g<g_name>] [-r<sec>] [-u<u_name>] [<dbid>...]
```

One or more <dbid> arguments may be supplied. If <dbid> is omitted, showipc processes the ipc structures for all active databases. In addition to the numeric values that represent valid DBIDs, the strings CSCI, ACS and U can also be supplied. CSCI (client server communication interface) returns information for all ipc elements of CSCI users. ACS returns information about the ipc elements used by NETACS (NET-WORK access server). U indicates all elements that cannot be associated with a database (database unknown).

showipc also supports multiple logical NET-WORK nodes. The NET-WORK ipc structures are identified by the <dbid> zero and a NET-WORK ID. Unless the -a or -e options are used (see below), information is only displayed for the default NET-WORK ID. If the environment variable NET_WORK_ID is set, the corresponding NET-WORK ID will also be processed.

Note: The information provided above concerning NET-WORK is only relevant for NET-WORK Version 2, which is no longer supported with Adabas Version 6. However, showipc of Adabas Version 6 also supports Adabas Version 5 databases, which can still be used with NET-WORK Version 2.

The following table shows the meanings of the various options that are supported:

Option	Meaning
-a	displays the structures of all of the logical NET-WORK nodes that are running simultaneously.
-d	this option was used up to Adabas Version 3.1. For compatibility reasons, this option can still be specified with the current version of Adabas, but it has no effect.
-e<string>	specifies additional NET-WORK IDs for which information is to be displayed. Because the first character of a NET-WORK ID must be unique, each character of <string> uniquely identifies one NET-WORK ID.
-f	used with the -k option, this forces the removal of the NETACS and CSCI elements (which would not be deleted otherwise).
-g<g_name>	searches for the ipc elements that are owned by the group <g_name>, instead of the default users ('sag', 'adabas', 'natural', 'esq', $SIPGROUP).
-h	calls up a help screen that explains how to use showipc.
-i	used with the -k option, this requests confirmation before deleting any ipc element.
-k	removes ipc elements that are no longer in use. Before it removes anything, showipc checks to see whether the GCB shared memory of the database is still in use. If it is still in use, any removal requests are rejected. Software AG strongly recommends that the -i option is used with the -k option. showipc is run with the effective user ID of root, otherwise you would not be able to remove the IPC resources if they were created by an Adabas nucleus process that was started by another user.
-r<sec>	the specified command is repeated every <sec> seconds. The repetition can be stopped by entering CRTL-C.
-s	displays the ipc configuration values of the running UNIX kernel. This can be useful to check whether the UNIX kernel is configured correctly.
-u<u_name>	searches for the ipc elements that are owned by the user <u_name>, instead of the default users ('sag', 'adabas', 'natural', 'esq', $SIPUSER).
-v	displays the database information in verbose format.

Output of showipc

The first column of the default output format contains either “NET” for NET-WORK (followed by the NET-WORK ID if it is not the default NET-WORK ID), or “NACS” for NETACS, or “CSCI” for CSCI, or the numeric DBID of the database in question. This is followed by a list of the ipc elements for the database. The entry for each element consists of an abbreviation to indicate its type, and, separated by a colon, the ipc id of the element. This id can then be used for the “ipcrm” command ("showipc -k <dbid>" is, however, much more convenient). The following abbreviations are used:

shared memory

ATB	attached buffer shared memory
CSA	common shared area
CSM	CSCI shared memory
DRV	Adabas ipc driver shared memory
GCB	general control block
GDT	global database table (NET-WORK)
OPR	shared memory created with ADAOPR (called with CSA=)
PHx	protocol handler with ID x (NET-WORK)
ESi	Adabas SQL server shared memory ID i

message queues

CLM	communication client message queue
CSQ	CSCI message queue
RSQ	user response queue
SRV	communication server message queue
SVQ	nucleus thread queue
USQ	user request queue

semaphores

CLS	communication client semaphore
CSS	CSCI semaphore
PSE	private semaphore
SEM	other semaphore
ESQ	Adabas SQL server semaphore

The verbose output format (-v option) displays additional information for each element. Three tables are displayed for each database: one for shared memory, one for message queues, and one for semaphores. The columns in these tables have the following meanings:

shared memory

Column	Meaning
TYPE	type of ipc element (see abbreviations)
NI	NET-WORK ID (if not default); may be truncated to one character
ID	ipc ID
SIZE	size of the shared memory in bytes

Column	Meaning
LOPPID	process ID of the process that performed the last shmop() call
CRPID	process ID of the creator process
ATTPROC	number of attached processes (not always supported by the operating system)

message queues

Column	Meaning
TYPE	type of ipc element (see abbreviations)
NI	NET-WORK ID (if not default); may be truncated to one character
ID	ipc ID
NBYTES	number of bytes in the message queue
NMSGS	number of messages in the message queue
LSND	process ID of the last sending process
LRCV	process ID of the last receiving process

semaphores

Column	Meaning
TYPE	type of ipc element (see abbreviations)
NI	NET-WORK ID (if not default); may be truncated to one character
ID	ipc ID

For NETACS elements, the column TYPE is replaced by DOMN, which represents the NETACS domain.

Note: Because showipc reads the kernel memory structures, it must be run with the effective user id of root.
If the running kernel does not have the default name (for example “/hp-ux” for HP-UX,, ...), showipc will abort with an appropriate error message. This can be prevented by setting the environment variable SIP_KERNAM to the correct name, in csh for example “setenv SIP_KERNAM '/mykernel'”.

Environment Variables

The following environment variables are used together with showipc:

Environment Variable	Meaning
NET_WORK_ID	This specifies a NET-WORK ID to be processed in addition to the default NET-WORK ID (see the showipc options -e and -a for further information).
SIP_KERNAM	This sets the name of the kernel (see the note above for further information).
SIPGROUP	If the environment variable SIPGROUP is set, its contents will be appended to the list of default groups (see also option -g).
SIPUSER	If the environment variable SIPUSER is set, its contents will be appended to the list of default users (see also option -u). For example, if SIPUSER is set to “harry”, all of the ipc elements that belong either to the default users (currently “sag”, “adabas”, “esq” and “natural”) or belong to “harry” will be displayed.

Messages

The following messages may be received when using showipc:

Cleared up driver resources for DB xx

Explanation showipc not only removes ipc structures if the -k option is used, but for Adabas Version 2.1 and above also releases the ipc driver resources for the corresponding database.

Action None.

Could not attach to global data area of database xxx

Explanation The CSA of the database xxx is corrupted or not in the format expected by showipc.

Action If this problem occurs with an active database, contact your nearest Adabas support centre.

Database yy of Adabas Vz.z might be incorrectly displayed by showipc V x.x

Explanation The database in question is of a more recent version than the current version of showipc, which means that it cannot be guaranteed that the output of the ipc structures for this database is correct.

Action Use an appropriate, more recent version of showipc.

Db xx is of version ww - please use version ww of showipc to remove

Explanation The nucleus of database xx is of a higher version than the showipc called. showipc will not remove any ipc structure for this database.

Action Use an appropriate, more recent version of showipc.

Found inconsistent data structures for DB xxx

Explanation The internal data structures of database xxx are incompatible with showipc.

Action Check the version of the database. If it is more recent than the version of showipc that you are using, use an appropriate version of showipc. If the problem still occurs, contact your nearest Adabas support centre.

GCB of DB xx still in use, I do not remove anything

Explanation showipc -k was called for a database that is still active (running nucleus or being accessed by Adabas utilities). No ipc structures will be removed.

Action If you really want to remove the ipc structures for the database in question, terminate the process that is accessing the database (use showipc -v <xx> to determine the process) and try the command again.

GDT is still in use by DB xx; not removed

Explanation There is still at least one database using the GDT: it will not be removed by showipc.

Action None.

I do not remove private semaphores

Explanation showipc does not remove Adabas semaphores that are not associated with a specific database.

Action None. Removing the semaphores with ipcrm could cause Adabas utilities to fail/abort.

NET-WORK xx still running, I do not remove anything

Explanation The NET-WORK with the specified ID is still active. showipc does not remove any ipc structures that are associated with this NET-WORK ID.

Action None.

Sorry, but this program has to be run with supervisor privileges

Explanation The showipc executable does not have the correct permissions set. It must have a set user ID permission, and have the owner root.

Action Check and, if necessary, correct the permissions.

Sorry, I will remove CSCI elements only with -f option

Explanation CSCI elements can only be removed if the -f option is used.

Action Add the -f option to the -k option.

Sorry, I will remove CSCI elements only with -f option

Explanation NET-ACS elements can only be removed if the -f option is used.

Action Add the -f option to the -k option.

Structure mismatch for DB xx - Cannot identify version

Explanation The internal data structures of the database in question cannot be correctly interpreted.

Action This message may occur during the startup of a database: in this case it can be ignored.

Warning: could not find adanod for clean up of Adabas driver

Explanation In some cases, the utility adanod is required when cleaning up the Adabas ipc driver resources.

Action Set the environment variable ADANOD to the full path name of the adanod executable.

Warning: Found <structure> with invalid database ID xx contained

Explanation A structure that belongs to one of the relevant users has an invalid key. The structure can be either shared memory, a message queue, or a semaphore.

Action Check which process created the structure in question (showipc displays the creator's process ID). If it is not a SOFTWARE AG product, it should be modified so that it does not interfere with Adabas. If it is a SOFTWARE AG product, contact your nearest Adabas support centre.

Warning: Some NET-WORK IDs may be displayed truncated to one character

Explanation When the showipc options -e or -a are used, showipc might truncate some NET-WORK IDs to one character. NET-WORK IDs are, however, identified uniquely by their first character.

Action You can force one NET-WORK ID to be expanded by setting the environment variable NET_WORK_ID accordingly.

Examples

```
showipc
```

This displays the ipc structures of all Adabas databases and of the default NET-WORK ID.

```
showipc -av
```

This displays the ipc structures of all Adabas databases and of all logical NET-WORK IDs in the verbose format.

```
showipc -ki 127
```

This removes all of the ipc structures of database 127 if the database nucleus is not running. showipc will ask for confirmation before an ipc structure is removed.

Providing Diagnostic Information for Adabas Support

General Information

Diagnostic information should generally be submitted in the form of attachments to Support Incidents via eService or email. Attachments of 5 MB or larger should be submitted via FTP, SFTP or FTPES, according to the instructions below for your region. You should only send DVDs (or other approved physical media) if requested by Software AG.

Please use the following file naming convention when you upload diagnostic data of all kinds to Software AG using FTP, SFTP, or FTPES. The name of the file must begin with the two letters 'SR', followed by the 7-digit Support Incident number. Following this prefix, you can use any kind of meaningful and descriptive file name and file extension. Valid examples are:

- SR1234567Diagnostics1.zip
- SR1234567Diagnostics2.zip
- SR1234567_more.1000.Z
- SR1234567_configuration_information.txt
- SR1234567ACMETestCorp.acme
- SR1234567_no_file_extension

If you experience issues when uploading large files, you should make use of the `Split to Volumes` feature provided by file compression tools such as WinZip or 7-Zip.

In order to verify the integrity of the documents you send, please consider providing us with MD5 hash sums of the files.

Sending Files via FTP

To upload files via "plain" FTP

1 Select the server closest to you:

- ftp.softwareag.com (located at Software AG's headquarters in Germany)
- ftp.softwareagusa.com (located at Software AG's U.S. offices in Reston, VA.)

2 Log in with the following credentials:

- User ID: customer
- Password: customer

Note: Your files will *NOT* be associated with the Support Incident if you use "anonymous".

The directory will appear to be empty, and it does not allow the `DIR` or `LS` commands to be executed, so, after uploading, you will no longer be able to see the file(s). However, after allowing approx. 45 minutes of processing time, you can confirm receipt of the file(s) in the eService section of Empower – the Support Incident will show steps of type 'FTP – Inbound'.

Please use the 'BIN' option of your FTP client.

How to prepare Problem Information

If a problem should occur at a customer site, Software AG's support team wants to supply solutions as fast as possible. To minimize the time required to analyze a problem and deliver a solution, the support team requires detailed information about the problem.

The standard problem information should contain:

- Problem description:
  - Adabas component or function used
  - error messages (message ID)
  - short description of the problem
- Environment description:
  - Adabas version
  - corrections used
  - operating system version
  - hardware environment (CPU type, disk device types, etc.)
- Can the problem be reproduced ?

- What are the steps to reproduce it
- Which data is needed to reproduce it

If the problem is not reproducible, the problem information should contain additionally:

- Where does the problem occur ?
  - on all databases or on a single databases only
  - on databases within a specific environment only
  - environments where the problem does not occur
- When does the problem occur ?
  - always or only sporadically
  - in parallel with other events
  - periods of time when the problem does not occur
  - when did the problem first occur ?
  - changes of the environment around this date
  - last changes of the environment before this date
  - in the case of data corruption problems:
  - date of last successful database verification run
  - utilities used on the corrupted file (checkpoint list)
  - disk problems encountered