SYBASE®
An SAP Company

Reference Manual: Tables

# Adaptive Server® Enterprise

15.7 ESD #2

DOCUMENT ID: DC36274-01-1572-01

LAST REVISED: July 2012



Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

# Contents

CHAPTER 1

# System Tables

System tables are tables supplied by Sybase®. Most system tables in Adaptive Server® are row-locked tables. Those that are not, are noted in the individual system table descriptions.



## Locations of system tables

System tables may be located in:

- The master database,
- The sybsecurity database,
- The sybsystemdb database, or
- All databases.

Most tables in the master database are system tables. Some of these tables also occur in user databases. They are automatically created when the create database command is issued.

### System tables in *master*

The following system tables occur *only* in the master database:

| System table | Contents |
|---|---|
| syscharsets | One row for each character set or sort order. |
| sysconfigures | One row for each configuration parameter that can be set by users. |
| syscurconfigs | Information about configuration parameters currently being used by Adaptive Server. |
| sysdatabases | One row for each database on Adaptive Server. |

| System table | Contents |
|---|---|
| sysdevices | One row for each tape dump device, disk dump device, disk for databases, and disk partition for databases. |
| sysengines | One row for each Adaptive Server engine currently online. |
| syslanguages | One row for each language (except U.S. English) known to the server. |
| syslisteners | One row for each type of network connection used by the current Adaptive Server. |
| syslocks | Information about active locks. |
| sysloginroles | One row for each server login that possesses a system role. |
| syslogins | One row for each valid Adaptive Server user account. |
| syslogshold | Information about the oldest active transaction and the Replication Server® truncation point for each database. |
| sysmessages | One row for each system error or warning. |
| sysmonitors | One row for each monitor counter. |
| sysprocesses | Information about server processes . |
| sysremotelogins | One row for each remote user . |
| sysresourcelimits | One row for each resource limit. |
| syssecmechs | Information about the security services available for each security mechanism that is available to Adaptive Server. |
| sysservers | One row for each remote Adaptive Server. |
| syssessions | Used only when Adaptive Server is configured for Sybase Failover in a high availability system. syssessions contains one row for each client that connects to Adaptive Server with the failover property. |
| syssrvroles | One row for each server-wide role. |
| systimeranges | One row for each named time range. |
| systransactions | One row for each transaction. |
| sysusages | One row for each disk piece allocated to a database. |

## System tables in *sybsecurity*

The following system tables occur *only* in the sybsecurity database:

| System table | Contents |
|---|---|
| sysauditoptions | One row for each global audit option. |
| sysaudits_01 – sysaudits_08 | The audit trail. Each audit table contains one row for each audit record. |

All auditing-related system tables are allpages locked.

## System table in *sybsystemdb*

The syscoordinations system table, which consists of one row for each remote participant of a distributed transaction, occurs only in sybsystemdb.

## System tables in all databases

The following system tables occur in all databases:

| System table | Contents |
|---|---|
| sysalternates | One row for each Adaptive Server user mapped to a database user. |
| sysattributes | One row for each object attribute definition. |
| syscolumns | One row for each column in a table or view, and for each parameter in a procedure. |
| syscomments | One or more rows for each view, rule, default, trigger, and procedure, giving SQL definition statement. |
| sysconstraints | One row for each referential and check constraint associated with a table or column. |
| sysdepends | One row for each procedure, view, or table that is referenced by a procedure, view, or trigger. |
| sysgams | Allocation bitmaps for an entire database. |
| sysindexes | One row for each clustered or nonclustered index, one row for each table with no indexes, and an additional row for each table containing text or image data. |
| sysjars | One row for each Java archive (JAR) file that is retained in the database. |
| syskeys | One row for each primary, foreign, or common key; set by user (not maintained by Adaptive Server). |
| syslogs | Transaction log. |
| sysobjects | One row for each table, view, procedure, rule, trigger default, log, and (in tempdb only) temporary object. |
| syspartitionkeys | One row for each partition key. |
| syspartitions | One row for each partition of a partitioned table or index. |
| sysprocedures | One row for each view, rule, default, trigger, and procedure, giving internal definition. |
| sysprotects | User permissions information. |
| sysquerymetrics | Gathers aggregated historical query information in a persistent catalog. sysquerymetrics is a view, not a table. |
| sysqueryplans | Abstract query plans and SQL text. |
| sysreferences | One row for each referential integrity constraint declared on a table or column. |
| sysroles | Maps server-wide roles to local database groups. |
| syssegments | One row for each segment (named collection of disk pieces). |
| sysslices | Obsolete, used only during upgrade. Formerly called syspartitions before Adaptive Server version 15.0. |
| sysstatistics | One or more rows for each indexed column on a user table. May also contain rows for unindexed column. |

| System table | Contents |
|---|---|
| systabstats | One row for each table, plus one row for each nonclustered index. |
| systhresholds | One row for each threshold defined for the database. |
| systypes | One row for each system-supplied and user-defined datatype. |
| sysusermessages | One row for each user-defined message. |
| sysusers | One row for each user allowed in the database. |
| sysxtypes | One row for each extended, Java-SQL datatype. Uses row-level locking. |

## About the *sybdiagdb* database

Sybase Technical Support may create the sybdiagdb database on your system for debugging purposes. This database holds diagnostic configuration data for use by Technical Support representatives.

## About the *syblicenseslog* table

The syblicenseslog table is described in syblicenseslog on page 9. It is not technically a system table, but you may need to consult it for license information related to shutting down Adaptive Server.

# Using system tables in the Cluster Edition

This section describes general changes to the system tables for the Cluster Edition. Changes to specific tables are listed under the table heading.

## *timestamp* columns

In Adaptive Server, if a table includes a timestamp column, its value is updated when a row is changed. Client applications can use this functionality to detect changes to rows using an access method called “optimistic locking.” The values in the timestamp column are unique in a database. However, in the Cluster Edition, timestamp column values are not guaranteed to be in increasing order in a database across tables, but they are guaranteed to be in increasing order for a particular table.

## Changed identity values

Identity columns in the Cluster Edition behave differently from those in non-clustered editions of Adaptive Server. Although the Cluster Edition guarantees that identity values are unique, for performance reasons the values may not monotonically increase.

In a non-clustered Adaptive Server, a set of identity values are burned into memory to reduce disk I/Os as inserts access the next value from memory. In the Cluster Edition, the same size set is burned into memory, but the set is split among the cluster instances. In a two-instance cluster with an identity set size of 250000, the first instance inserts values {1,2,3, and so on}, and the second instance inserts values {125000,125001,125002, and so on}.

The next-identity function reports the next identity value for a table from the instance in which next-identity is executed. For example, next-identity returns 4 for instance 1 and 125003 for instance 2.

The behavior of the identity-burn-max remains the same as for a non-clustered Adaptive Server because the burn size and burn behavior is unchanged in the Cluster Edition.

## Controlling fake-table materialization

Certain stored procedures, such as sp_who and sp_lock, read from fake tables such as sysprocesses and syslocks. Because their rows are not stored on disk, fake tables present an exception to the shared-data nature of a shared-disk cluster, and special features apply.

You can control whether a fake-table query returns rows from the local instance or all instances in the cluster by using the set system_view command. set system_view is a session-level command. In addition, set system_view also controls monitoring table materialization.

For information about setting the default system view at the logical-cluster level see the *Users Guide to Clusters*.

By default, Adaptive Server retrieves rows only from the local instance.

- To specify that fake-table queries materialize rows for all instances, use the cluster option. For example:

  ```
  set system_view cluster
  ```

- To specify that fake-table queries materialize rows for the local instance, use the instance option. For example:

```
set system_view instance
```

To retrieve the current system_view setting, select the *@@system_view* global variable.

Adaptive Server supports cluster-wide materialization for these fake tables:

- sysprocesses
- syslocks
- sysengines
- syslisteners
- sysmonitors
- syssechmechs
- syscurconfigs

**Note** sysinstances is always set for cluster-wide materialization, regardless of the system_view setting.

# Rules for using system tables

This section describes rules, restrictions, and usage information for system tables.

**Note** By default, a column is defined as NOT NULL. Nullable columns are described using the “null” keyword, and are listed in the column descriptions for the tables in this book.

## Permissions on system tables

Permissions for use of the system tables can be controlled by the Database Owner, just like permissions on any other tables. By default, when Adaptive Server is installed, the installmodel script grants select access to "public" (all users) for most system tables and for most fields in the tables. Instead, the default permissions on the system tables are assigned when Adaptive Server builds a new database. However, no access is granted to some system tables, such as syssrvroles, and no access is granted to certain fields in other system tables. For example, all users, by default, can select all columns of sysobjects except audflags. See the *Security Administration Guide* for more information.

sp_helprotect *system_table_name*

For example, to check the permissions of syssrvroles in master, execute:

```
use master
go
sp_helprotect syssrvroles
go
```

## Locking schemes used for system tables

In the allpages locking scheme in Adaptive Server, locks are acquired on data and index pages. See the *Performance and Tuning Guide: Locking* for more information on locking schemes.

All system tables use datarow locking except for the following, which use allpages locking:

- sysusermessages
- sysslices
- sysmessages

In addition, the following system tables are "fake"—or non-row-oriented—catalogs that give the appearance of using allpages locking:

- syslogs
- sysgams
- sysprocesses
- syslocks
- syscurconfigs

- syssecmechs
- sysmonitors
- sysengines
- systestlog
- syslisteners
- syslogshold

## Reserved columns

The word “reserved” in the column description means that the column is not currently used by Adaptive Server.

## Updating system tables

Direct updates on system tables are not allowed—even for the Database Owner. Instead, Adaptive Server includes system procedures that you should use to make any normally needed updates and additions to system tables.

You can allow direct updates to the system tables if it you must modify them in a way that cannot be accomplished with a system procedure. To allow direct udpates, a system security officer must use sp_configure to reset the configuration parameter called allow updates to system tables. For more information, see the *Security Administration Guide*.

## Triggers on system tables

You cannot create triggers on system tables. If you try to create a trigger on a system table, Adaptive Server returns an error message and cancels the trigger.

# syblicenseslog

master database only

Description

syblicenseslog contains one row for each update of the maximum number of licenses used in Adaptive Server per 24-hour period. syblicenseslog is updated every 24 hours. If Adaptive Server is shut down at any time, License Use Manager logs the number of licenses currently being used in syblicenseslog before the shutdown is complete. The 24-hour period restarts when you start Adaptive Server.

**Note** syblicenseslog is not a system table. Its type is "U" and its object ID is greater than 255.

Columns

The columns for syblicenseslogs are:

| Name | Datatype | Description |
|---|---|---|
| status | smallint | Status of the maximum number of licenses used; one of the following:<br>• 0 = number of licenses not exceeded<br>• 1 = number of licenses is exceeded<br>• -1 = housekeeper is unable to monitor number of licenses |
| logtime | datetime | Date and time the log was written |
| maxlicenses | int | Maximum number of licenses used during the 24-hour period |

# sysalternates

## All databases

Description

sysalternates contains one row for each Adaptive Server user that is mapped or aliased to a user of the current database. When a user tries to access a database, Adaptive Server looks for a valid uid entry in sysusers. If none is found, it looks in sysalternates.suid. If the user's suid is found there, he or she is treated as the database user whose suid is listed in sysalternates.altsuid.

Columns

The columns for sysalternates are:

| Name | Datatype | Description |
|---|---|---|
| suid | int | Server user ID of user being mapped |
| altsuid | int | Server user ID of user to whom another user is mapped |

Indexes

- Unique clustered index on suid.

# sysaltusages

## Scratch database

Description

The sysaltusages system table maps page numbers in an archive database to the actual page within either the database dump and its stripes, or the modified pages section. However, unlike the sysusages table in a traditional database, the sysaltusages table does not map every logical page in the database. sysaltusages maps pages that have been:

- Stored in a database dump
- Modified, and therefore, relocated to the modified pages section

See Chapter 14, "Archive Database Access," in the *System Administration Guide, Volume 2*.

Columns

The columns for sysaltusages are:

| Name | Datatype | Description |
|---|---|---|
| dbid | smallint | The database ID of the archive database |
| location | int | The location of the archive database segment where the physically contiguous block of pages resides.<br>In the location column, a value of 5 and 6 means the location is in the database dump, transaction log dump, or their stripes, and a value of 7 or 8 means that the location is in the modified pages section. A value of 4 is used to fill the gaps for pages that are not physically available. |
| lstart | unsigned int | The logical page number of the start of the block of physically contiguous pages. |
| size | unsigned int | The number of logical pages in the block of physically contiguous pages. |
| vstart | int | The offset of the start of the contiguous block of pages on the device given by vdevno. |
| vdevno | int | The device number on which the contiguous block of pages resides. |
| segmap | int | A map of the segments to which this block of pages is allocated. |

**Note** Because sysaltusages is a row-locked catalog, you may need to periodically use reorg to reclaim logically deleted space.

The scratch database stores the new sysaltusages table. The scratch database is used to provide flexibility as to where the sysaltusages table is located.

The scratch database can be any database (with some exceptions like master and temporary databases.) Sybase recommends that you dedicate a database that is used only as a scratch database, because:

- The size of sysaltusages may vary depending on the number of archive databases it supports. You cannot decrease the size of a database, but if it is too large, you can drop it and re-create a smaller database when required.
- It allows you to turn on the "trunc log on checkpoint" option so that the database log be automatically truncated.

Apart from hosting the sysaltusages table, this database is like any other. You can use threshold procedures and other space management mechanisms to manage space within the database.

**The scratch database** You must specify a database that is to be used as a scratch database, by entering:

```
sp_dboption <db name>, "scratch database", "true"
```

Each archive database can be assigned to only one scratch database at a time, but multiple archive databases can use the same scratch database. If you have a large number of archive databases, you may want to define multiple scratch databases.

sysaltusages includes a unique clustered index named csysaltusages on dbid, location, and lstart.

# sysattributes

## All databases

Description

System attributes define properties of objects such as databases, tables, indexes, users, logins, and procedures. sysattributes contains one row for each of an object's attribute definitions (configured by various system procedures). master..sysattributes defines the complete set of valid attribute values and classes for Adaptive Server as a whole. It also stores attribute definitions for server-wide objects, such as databases and logins.

Use only system procedures to access sysattributes. The permissions required for modifying sysattributes depend on the system procedure you use.

Columns

The columns for sysattributes are:

| Name | Datatype | Description |
|---|---|---|
| class | smallint | The attribute class ID. This describes the category of the attribute. In master..sysattributes, the special class 0 identifies all valid *classes* of attributes for Adaptive Server. |
| attribute | smallint | The attribute ID, attribute specifies a default decrypt on an encrypted column with a value of 1 (DECRYPT-DEFAULT_ID) for objects with a type of EC and a class of 25. |
| object_type | char(2) | A one- or two-letter character ID that defines the type of object to associate with the attribute. |
| object_cinfo | varchar(255) null | A string identifier for the object (for example, the name of an application). This field is not used by all attributes. |
| object_cinfo2 | varchar(255) null | A string identifier for the object (for example, the name of an application) in a SDC environment. This field is not used by all attributes. |
| object | int null | The object identifier. This may be an object ID, user ID, decrypt default ID, or database ID, depending on the type of object. If the object is a part of a table (for example, an index), this column contains the object ID of the associated table. |
| object_info1, object_info2, object_info3 | int null | Defines additional information required to identify the object. This field is not used by all attributes. The contents of this field depend on the attribute that is defined.<br>• object_info_1 – includes the table ID for a table whose encrypted column defines the decrypt default.<br>• object_info2 – specifies the *colid* of the encrypted column that includes the decrypt default. |
| int_value | int null | An integer value for the attribute (for example, the display level of a user). |
| char_value | varchar(768) null | A character value for the attribute (for example, a cache name). |
| text_value | text null | A text value for the attribute. |
| image_value | image null | An image value for the attribute. |

| Name | Datatype | Description |
|---|---|---|
| comments | varchar(255) null | Comments or additional information about the attribute definition. |
| object_datetime | null | datetime value for the attribute. Its use depends on the module using the attribute, but it typically refers to the date and time the attribute was created. |

Table 1-1 lists the relevant values most frequently used in object_type. These values provide additional information for sysattributes, and are not for use as standalone values. For this reason, use these values only in conjunction with the class ID.

***Table 1-1: Valid values for the object_type column of sysattributes***

| Value | Description |
|---|---|
| D | Database |
| I | Index |
| L | Login |
| LR | Login Profile |
| P | Proc |
| T | Table |
| U | User |
| AP | Application |
| DC | Dump Condition |
| EL | External Login (OMNI) |
| OD | Object Definition (OMNI) |
| TC | Transaction Coordination (ASTC) |
| TG | Temporary Database Group (mult tempdb) |
| TP | Text Page (OMNI) |
| QP | Query Plans (abstract plans) |
| UR | User Role |
| GR | Group Role |
| LG | Login (for MTDB binding) |
| EG | Engine Group |
| PS | Password Security |
| SP | Keypair regeneration period |

Indexes

- Unique clustered index on class, attribute, object_type, object, object_info1, object_info2, object_info3, object_cinfo.
- Nonclustered index on object_type, object, object_info1, object_info2, object_info3, object_cinfo.

# sysauditoptions

## sybsecurity database

Description

sysauditoptions contains one row for each server-wide audit option and indicates the current setting for that option. Other types of auditing option settings are stored in other tables. For example, database-specific option settings are stored in sysdatabases, and object-specific option settings are stored in sysobjects. The default value for each option is 0, or "off." Only system security officers can access sysauditoptions.

Columns

The columns for sysauditoptions are:

| Name | Datatype | Description |
|---|---|---|
| num | smallint | Number of the server-wide option. |
| val | smallint | Current value; one of the following:<br>• 0 = off<br>• 1 = pass<br>• 2 = fail<br>• 3 = on |
| minval | smallint | Minimum valid value for this option. |
| maxval | smallint | Maximum valid value for this option. |
| name | varchar(30) | Name of option. |
| sval | varchar(30) | String equivalent of the current value: for example, "on", "off", "nonfatal". |
| comment | varchar(255) | Description of option. |

# sysaudits_01 – sysaudits_08

## sybsecurity database

Description

These system tables contain the audit trail. Only one table at a time is active. The active table is determined by the value of the current audit table configuration parameter. An installation can have as many as eight audit tables. For example, if your installation has three audit tables, the tables are named sysaudits_01, sysaudits_02, and sysaudits_03. An audit table contains one row for each audit record.

Columns

The columns for sysaudits_01 – sysaudits_08 are:

| Name | Datatype | Description |
|---|---|---|
| event | smallint | Type of event being audited. |
| eventmod | smallint | Further information about the event. Possible values are:<br>• 0 = no modifier for this event.<br>• 1 = the event passed permission checking.<br>• 2 = the event failed permission checking. |
| spid | smallint | Server process ID of the process that caused the audit record to be written. |
| | int for the Cluster Edition | |
| eventtime | datetime | Date and time of the audited event. |
| sequence | smallint | Sequence number of the record within a single event; some events require more than one audit record. |
| suid | smallint | Server login ID of the user who performed the audited event. |
| dbid | int null | Database ID in which the audited event occurred or the object/stored procedure/trigger resides, depending on the type of event. |
| objid | int null | ID of the accessed object or stored procedure/trigger. |
| xactid | binary(6) null | ID of the transaction containing the audited event. For a multidatabase transaction, this is the transaction ID from the database where the transaction originated. |
| loginname | varchar(30) null | Login name corresponding to the suid. |
| dbname | varchar(30) null | Database name corresponding to the dbid. |
| objname | varchar(255) null | Object name corresponding to the objid. |
| objowner | varchar(30) null | Name of the owner of objid. |
| extrainfo | varchar(255) null | Additional information about the audited event. This field contains a sequence of items separated by semicolons. See Table 1-2. |
| nodeid | tinyint null | Reserved for future use (not available for cluster environments) |
| instanceid | tinyint | ID of the instance (available only for cluster environments) |

**Note** Because of this change in the datatypes for the Cluster Edition, Sybase strongly recommends that you archive and truncate audit tables before you upgrade. This reduces the likelihood of a failed upgrade because of insufficient space in the sybsecurity database.

The extrainfo column contains a sequence of items separated by semicolons as shown in Table 1-2:

***Table 1-2: Items in the extrainfo column***

| Item | Contents |
|---|---|
| Roles | Lists the roles that are active. The roles are separated by blanks. |
| Keywords or options | The name of the keyword or command option that was used for the event. For example, for the alter table command, the options add column or drop constraint might be used. Multiple keywords or options are separated by commas. |
| Previous value | The value prior to the update if the event resulted in the update of a value. |
| Current value | The new value if the event resulted in the update of a value. |
| Other information | Additional security-relevant information that is recorded for the event. |
| Proxy information | The original login name, if the event occurred while a set proxy was in effect. |
| Principal information | The principal name from the underlying security mechanism, if the user's login is the secure default login, and the user logged in to Adaptive Server using unified login. The value of this field is NULL, if the secure default login is not being used. |

An example of an extrainfo column for the security-relevant event of changing an auditing configuration parameter might be:

```
sso_role;suspend auditing when full;1;0;;;;
```

This example indicates that a system security officer changed the configuration parameter suspend auditing when full from 1 (suspend all processes that involve an auditing event) to 0 (truncate the next audit table and make it the current audit table).

# syscacheinfo

## master database

Description

Provides information about data caches.

syscacheinfo is a view of the master database that provides information about the configuration of data caches and pools.

Access to the views is restricted to users with the sa_role role.

Columns

The columns for syscacheinfo are:

| Name | Datatype | Description |
|---|---|---|
| cache_name | varchar(30) | Name of the cache in which this pool is allocated. |
| cache_status | varchar(8) | Status of the cache. One of:<br>• Active<br>• Pend/Act<br>• Act/Del |
| cache_type | varchar(16) | Type of cache. One of:<br>• Mixed, HK Ignore<br>• Mixed<br>• Log Only<br>• In-Memory Storage<br>• Default |
| config_size | float | The currently configured size of the cache, in megabytes. May be different from the actual size of the cache, reported in the run_size column. |
| run_size | float | The current amount of memory, in megabytes, allocated to the cache. May be different from the configured size reported by the config_size column. |
| config_replacement | varchar(11) | Currently configured buffer replacement strategy. None, or one of:<br>• Strict LRU<br>• Relaxed LRU |
| run_replacement | varchar(11) | Current buffer replacement strategy for the cache. None, or one of:<br>• Strict LRU<br>• Relaxed LRU |
| config_partitions | int | Configured number of partitions in the data cache. |
| run_partitions | int | The current number of partitions in the data cache. |
| overhead | numeric | Amount of memory overhead for the data cache. |
| cacheid | int | ID of the data cache. |
| instanceid | int | ID of the instance (zero for non-Cluster Edition servers). |

| Name | Datatype | Description |
|---|---|---|
| scope | varchar(6) | Indicates whether the data cache is local or global for Cluster Edition. The value is always Global for nonclustered servers. |

# syscachepoolinfo

Description

Provides a row for each data cache pool that includes configuration information for the data cache. This view is a join between the syscacheinfo and syspoolinfo views.

Columns

Access to the views is restricted to users with the sa_role role.

The columns for syscacheinfo are:

| Name | Datatype | Description |
|---|---|---|
| cache_name | varchar(30) | Name of the cache in which this pool is allocated. |
| cache_status | varchar(8) | Status of the cache. One of:<br>• Active<br>• Pend/Act<br>• Act/Del |
| cache_type | varchar(16) | Type of cache. One of:<br>• Mixed, HK Ignore<br>• Mixed<br>• Log Only<br>• In-Memory Storage<br>• Default |
| cache_config_size | float | The currently configured size of the cache, in megabytes. May be different from the actual size of the cache, reported in the run_size column. |
| cache_run_size | float | The current amount of memory allocated to the cache, in megabytes. May be different from the configured size reported by the config_size column. |
| cache_config_replacement | varchar(11) | Currently configured buffer replacement strategy. None, or one of:<br>• Strict LRU<br>• Relaxed LRU |
| cache_run_replacement | varchar(11) | Current buffer replacement strategy for the cache. None, or one of:<br>• Strict LRU<br>• Relaxed LRU |
| cache_config_partitions | int | Configured number of partitions in the data cache. |
| cache_run_partitions | int | The current number of partitions in the data cache. |
| cache_overhead | numeric | Amount of memory overhead for the data cache. |
| pool_io_size | varchar(3) | The size of the buffers, in kilobytes, used to perform I/O for this pool. |
| pool_config_size | float | Configured amount of memory, in megabytes, allocated to the pool. May be different from the amount reported in the run_size column. |
| pool_run_size | float | The current amount of memory, in megabytes, allocated to the pool. |

| Name | Datatype | Description |
|---|---|---|
| pool_apf_percent | int | The percentage of buffers in the pool that can be used to hold buffers that have been read into cache by asynchronous prefetch. |
| pool_wash_size | varchar(10) | The size of the wash area, in megabytes, in the pool. |
| cacheid | int | ID of the data cache. |
| instanceid | int | ID of the instance (zero for non-Cluster Edition servers). |
| scope | varchar(6) | Indicates whether the data cache is local or global for Cluster Edition. The value is always Global for nonclustered servers. |

# syscharsets

## master database only

Description

syscharsets contains one row for each character set and sort order defined for use by Adaptive Server. One of the sort orders is marked in master..sysconfigures as the default sort order, which is the only one actually in use.

Columns

The columns for syscharsets are:

| Name | Datatype | Description |
|---|---|---|
| type | smallint | The type of entity this row represents. Numbers from 1001 to 1999 represent character sets. Numbers from 2000 to 2999 represent sort orders. |
| id | tinyint | The ID for a character set or sort order. A sort order is defined by the combination of the sort order ID and the character set ID (csid). The character set is defined by id, which must be unique. Sybase reserves ID numbers 0 – 200. |
| csid | tinyint | If the row represents a character set, this field is unused. If the row represents a sort order, this is the ID of the character set that sort order is built on. A character set row with this ID must exist in this table. |
| status | smallint | Internal system status information bits. |
| name | varchar(30) | A unique name for the character set or sort order. Can use only the 7-bit ASCII letters A – Z or a – z, digits 0 – 9, and underscores (_), and must begin with a letter. |
| description | varchar(255) | An optional description of the features of the character set or sort order. |
| definition | image | The internal definition of the character set or sort order. The structure of the data in this field depends on the type. |
| sortfile | varchar(30) null | The name of the sort order file. |

Indexes

- Unique clustered index on id, csid
- Unique nonclustered index on name

# syscolumns

## All databases

Description

syscolumns contains one row for every column in every table and view, and a row for each parameter in a procedure.

Contains one row for each computed column and function-based index key associated with a table.

Columns

The columns for syscolumns are:

| Name | Datatype | Description |
|---|---|---|
| id | int | ID of table to which this column belongs, or of procedure with which this parameter is associated. |
| number | smallint | Sub-procedure number when the procedure is grouped (0 for nonprocedure entries). |
| colid | smallint | Column ID. |
| status | tinyint | • Bits 0–2 (values 1, 2, and 4) – indicate bit positioning if the column uses the bit datatype. If the column uses the text/image datatype, bits 0 and 1 indicate replication status as follows:<br>01 = always replicate<br>10 = replicate only if changed<br>00 = never replicate<br>• Bit 3 (value 8) – indicates whether NULL values are legal in this column.<br>• Bit 4 (value 16) – indicates whether more than one check constraint exists for the column.<br>• Bits 5 and 6 – are used internally.<br>• Bit 7 (value 128) – indicates an identity column. |
| type | tinyint | Physical storage type; copied from systypes. |
| length | int | Physical length of data; copied from systypes or supplied by user. |
| offset | smallint | Offset into the row where this column appears; if negative, this is a variable-length column. |
| usertype | smallint | User type ID; copied from systypes. |
| cdefault | int | ID of the procedure that generates default value for this column. |
| domain | int | Constraint ID of the first rule or check constraint for this column. |
| name | varchar(255) not null | Column name |
| printfmt | varchar(255) null | Reserved |
| prec | tinyint null | Number of significant digits, if the column uses the numeric datatype. |
| scale | tinyint null | Number of digits to the right of the decimal point, if the column uses the numeric datatype. |

| Name | Datatype | Description |
|---|---|---|
| remote_type | int null | Maps local names to remote names. Required by the access methods of Component Integration Services to allow the software to pass native column datatype information in parameters to servers of class access_server. |
| remote_name | varchar(255) null | Maps local names to remote names. Required by the access methods of Component Integration Services to construct a query using the proper column names for a remote table. |
| xstatus | int null | The status of a column with extended datatypes. The values are:<br>0 = in row<br>1 = off row<br>xstatus is NULL for columns that do not have an extended datatype. |
| xtype | int null | ID of the class.<br>Used if a column in a table or a parameter in a procedure has a Java class as its datatype. When used, fields are not NULL, and the value of type is 0x39. See *Java in Adaptive Server Enterprise* for more information. |
| xdbid | int null | The database ID of the class. For system classes, the value is -1. Otherwise, the value is the current database ID.<br>Used if a column in a table or a parameter in a procedure has a Java class as its datatype. Fields are not NULL, and the value of type is 0x39. See *Java in Adaptive Server Enterprise* for more information. |
| accessrule | int null | The object ID of the access rule in sysprocedures. See "Row-level access control" in Chapter 11, "Managing User Permissions" of the *Security Administration Guide* for more information. |

| Name | Datatype | Description |
|---|---|---|
| remote_type | int null | Maps local names to remote names. Required by the access methods of Component Integration Services to allow the software to pass native column datatype information in parameters to servers of class access_server. |
| remote_name | varchar(255) null | Maps local names to remote names. Required by the access methods of Component Integration Services to construct a query using the proper column names for a remote table. |
| xstatus | int null | The status of a column with extended datatypes. The values are:<br>0 = in row<br>1 = off row<br>xstatus is NULL for columns that do not have an extended datatype. |
| xtype | int null | ID of the class.<br>Used if a column in a table or a parameter in a procedure has a Java class as its datatype. When used, fields are not NULL, and the value of type is 0x39. See *Java in Adaptive Server Enterprise* for more information. |
| xdbid | int null | The database ID of the class. For system classes, the value is -1. Otherwise, the value is the current database ID.<br>Used if a column in a table or a parameter in a procedure has a Java class as its datatype. Fields are not NULL, and the value of type is 0x39. See *Java in Adaptive Server Enterprise* for more information. |
| accessrule | int null | The object ID of the access rule in sysprocedures. See "Row-level access control" in Chapter 11, "Managing User Permissions" of the *Security Administration Guide* for more information. |

| Name | Datatype | Description |
|---|---|---|
| status2 | int null | Indicates the parameter mode of a SQLJ stored procedure, and the return type of a SQLJ function:<br>• 0x00000001, value 1 – parameter mode "in"<br>• 0x00000002, value 2 – parameter mode "out"<br>These internal bits support computed columns:<br>• 0x00000010, value16 – the column is a computed column.<br>• 0x00000020, value 32 – the column is a materialized computed column.<br>• 0x00000040, value 64 – the column is a computed column in a view.<br>• 0x00001000, value 4096 – the encrypted column has a decrypt default.<br>The status2 field from syscolumns uses this encoding to indicate a column's encryption properties:<br>• 0x80, value 128 – the column is encrypted.<br>• 0x100, value 256– the column is encrypted with initialization vector.<br>• 0x200, value 512 – the column is encrypted with random padding.<br>• 0x400, value 1024 – the proxy table is encrypted.<br>• 0x1000, value 4096 – the encrypted column has a decrypt default.<br>• 0x20000, value 131072– the column is explicitly defined as not compressed.<br>• 0x00040000, value 262144 – the user-specified, or derived in-row length for LOB columns created as in-row. |
| status3 | int | 0x0001, value 1 – Indicates a hidden computed column for a function-based index key. |
| computedcol | int | Stores the object ID of the computed column definition. |
| encrtype | int null | Type of data in encrypted form. |
| lobcomp_lvl | tinyint | Compression level of the columns defined for large objects. |
| encrlen | int null | Length of encrypted data. |
| encrykeyid | int null | Object ID of key. |
| encrykeydb | varchar(30) null | Name of the database where the encryption key was created; NULL if it is in the same database as the encrypted column. |
| encrdate | datetime null | Creation date of encryption key; copied from sysobjects.crdate. |
| inrowlen | smallint | Stores the user-specified, or derived in-row length for LOB columns created as in-row. |

Indexes

- Unique clustered index on id, number, colid

# syscomments

## All databases

Description

syscomments contains entries for each view, rule, default, trigger, table constraint, partition, procedure, computed column, function-based index key, and other forms of compiled objects. The text column contains the original definition statements. If the text column is longer than 255 bytes, the entries span rows. Each object can occupy as many as 65,025 rows.

It also stores the text of a computed column, function-based index, or partition definition—for example, "values <= *value_list*" for a range partition.

The create service command stores text in syscomments, as it uses the create procedure infrastructure.

Columns

The columns for syscomments are:

| Name | Datatype | Description |
|---|---|---|
| id | int | Object ID to which this text applies. |
| number | smallint | Sub-procedure number when the procedure is grouped (0 for nonprocedure entries). |
| colid | smallint | The low portion of a column counter for this procedure's comments. Can vary from 0 to 32767. If a procedure has more text than fits in that many rows, this counter works together with colid2. |
| texttype | smallint | Indicates the comment type. Values are:<br>• 0 – system-supplied comment, for views, rules, defaults, triggers, and procedures<br>• 1 – user-supplied comment for adding entries that describe an object or column |
| language | smallint | Reserved. |
| text | varchar(255) null | Actual text of SQL definition statement. |
| colid2 | smallint | The high portion of a column counter for this procedure's comments. Can vary from 0 to 32767. Is only greater than 0 for procedures containing more than 32,768 rows of procedure text. |
| status | smallint null | Bits indicating the status of the objects:<br>• 0x1 – SYSCOM_TEXT_HIDDEN indicates that the text is hidden<br>• 0x2 – Reserved for internal use<br>• 0x4 – SYSCOM_QUOTED_ID_ON indicates that quoted identifiers were on when the object was created<br>• 0x8 – SYSCOM_SHARED_INLINE_DEF indicates the text is for a sharable inline default |

| Name | Datatype | Description |
|---|---|---|
| version | smallint null | The version of encryption that encodes the algorithm used to encrypt the hidden text for this row. One of:<br>• Null – no encryption for hidden text<br>• 1 – (the default) Adaptive Server obfuscation algorithm used in versions of Adaptive Server 15.0 and earlier<br>• 2 – (optional) Advanced Encryption Standard ("AES") strong encryption |
| partitionid | int null | Partition ID. Otherwise, null. |
| encrkeyid | int null | The encryption key ID from the key object in sysencryptkeys in the current database that Adaptive Server used to encrypt the hidden text of this object when version has a value of 2. Otherwise, Adaptive Server uses a value of null for encrkeyid. |

**Note** Do not delete the definition statements from the text column of syscomments. These statements are required for the Adaptive Server upgrade process. To encrypt a definition statement, execute the system procedure sp_hidetext. To see if a statement created in version 11.5 or later was deleted, execute sp_checksource. If the statement was deleted, you must either re-create the object that created the statement or reinstall the application that created the object, which re-creates the statement.

You can protect the text of a database object against unauthorized access by restricting select permission on the text column of the syscomments table to the owner of the object and the system administrator. This restriction, which applies to direct access through select statements as well as access through stored procedures, is required to run Adaptive Server in the evaluated configuration. To enact this restriction, a system security officer must reset the parameter called select on syscomments.text using the system procedure sp_configure. For information about the evaluated configuration, see the *Security Administration Guide: Volume 1*.

Indexes

- Unique clustered index on id, number, colid2, colid, texttype

# sysconfigures

## master database only

Description — sysconfigures contains one row for each configuration parameter that can be set by the user.

Columns — The columns for sysconfigures are:

| Name | Datatype | Description |
|---|---|---|
| config | smallint | Configuration parameter number. |
| value | int | The user-modifiable value for the parameter with integer datatype. Its value is 0 for the parameters with character datatype. |
| comment | varchar(255) | Name of the configuration parameter. |
| status | int | Value that represents the type of configuration parameter. For details, see Table 1-3. |
| name | varchar(255) null | Name of the configuration parameter (the same value as comment). |
| parent | smallint null | Configuration parameter number of the parent; if more than one parent, the additional parent numbers are stored in sysattributes. |
| value2 | varchar(255) null | The user-modified value for the parameter with the character datatype. Its value is NULL for parameters with integer datatype. value2 is also used to store the pool size of a buffer pool. |
| value3 | int null | Stores the wash size of a buffer pool. |
| value4 | int null | Stores the asynchronous prefetch percents of a buffer pool, or -1 where an unspecified or default value. |
| instanceid | tinyint | ID of the instance. Available only for cluster environments. |

Table 1-3 provides information about the status column.

***Table 1-3: Status column description***

| Status type | Decimal | Hex | Description |
|---|---|---|---|
| CFG_NO_OPTIONS | 0 | 0x0 | Parameter has no options. |
| CFG_SYSTEM_OPTION | 1 | 0x01 | Parameter is a system option. |
| CFG_SYSTEM_GROUP | 2 | 0x02 | Parameter is a system group. |
| CFG_STATIC | 4 | 0x04 | Parameter is static. |
| CFG_DYNAMIC | 8 | 0x08 | Parameter is dynamic. |
| CFG_CALCULATED | 16 | 0x10 | Parameter is calculated. |
| CFG_READONLY | 32 | 0x20 | Parameter is read-only. |
| CFG_MEMORY_USED | 64 | 0x40 | Parameter consumes memory. |
| CFG_CONFIG_FILE | 128 | 0x80 | Parameter is externally visible. |
| CFG_SYSTEM_TAB | 256 | 0x100 | Parameter is externally visible only in system table. |
| CFG_EXTRAS_OPTION | 512 | 0x200 | Parameter is for CFG_EXTRAS not DS_CONFIG. |

| Status type | Decimal | Hex | Description |
|---|---|---|---|
| CFG_CFGBLK | 1024 | 0x400 | Parameter is stored in the configuration block. |
| CFG_CACHE_GROUP | 2048 | 0x800 | Parameter is a cache group. |
| CFG_CACHE_OPTION | 4096 | 0x1000 | Parameter is a cache option. |
| CFG_BUFFER_POOL_GROUP | 8192 | 0x2000 | Parameter is a buffer pool group. |
| CFG_BUFFER_POOL_OPTION | 16384 | 0x4000 | Parameter is a buffer pool option. |
| CFG_INTERNAL | 32768 | 0x8000 | Parameter is for internal use only. |
| CFG_FNOF_LPAGESIZE | 65536 | 0x10000 | Parameter entry depends on logical pagesize. |

Indexes

- Unique clustered index on name, parent, config
- Nonclustered index on config
- Nonclustered index on parent, config

# sysconstraints

## All databases

Description

Whenever a user declares a new check constraint or referential constraint using create table or alter table, Adaptive Server inserts a row into the sysconstraints table. The row remains until a user executes alter table to drop the constraint. Dropping a table by executing drop table removes all rows associated with that table from the sysconstraints table.

This table also contains one row for each check constraint, referential constraint, computed column, and function-based index key associated with a specific table.

Columns

The columns for sysconstraints are:

| Name | Datatype | Description |
|---|---|---|
| colid | smallint | Column number in the table |
| constrid | int | Object ID of the constraint |
| tableid | int | ID of the table on which the constraint is declared |
| error | int | Constraint-specific error message |
| status | int | The type of constraint:<br>• 0x0040 = a referential constraint<br>• 0x0080 = a check constraint<br>• 0x0100 = a computed column object constraint |
| spare2 | int | Unused |

Indexes

- Unique clustered index on tableid, colid
- Nonclustered index on constrid

# syscoordinations

## sybsystemdb database only

Description

syscoordinations contains information about remote Adaptive Servers participating in distributed transactions (remote participants) and their coordination states.

Columns

The columns for syscoordinations are:

| Name | Datatype | Description |
|---|---|---|
| participant | smallint | Participant ID |
| starttime | datetime | Date the transaction started |
| coordtype | tinyint | Value indicating the coordination method or protocol in the systransactions table definition |
| owner | tinyint | Row owner (for internal use) |
| protocol | smallint | Reserved for internal use |
| state | int | Value indicating the current state of the remote participant:<br>• 1 – Begun<br>• 4 – Prepared<br>• 7 – Commited<br>• 9 – In AbortTrans |
| bootcount | int | Reserved for internal use |
| dbid | smallint | Database ID at the start of the transaction. |
| logvers | tinyint | Reserved for internal use |
| spare | tinyint | Reserved for internal use |
| status | int | Reserved for internal use |
| xactkey | binary(14) | Unique Adaptive Server transaction key |
| gtrid | varchar(255) null | Global transaction ID for distributed transactions coordinated by Adaptive Server (reserved for internal use) |
| partdata | varbinary(255) null | Reserved for internal use |
| srvname | varchar(30) null | Name of local server (null for remote servers) |
| nodeid | tinyint null | Not available for non-cluster environments – reserved for future use |
| instanceid | tinyint | *Cluster environments only* – ID of the instance |

Indexes

- Unique clustered index on xactkey, participant, owner

# syscurconfigs

## master database only

Description

syscurconfigs is built dynamically when queried. It contains an entry for each of the configuration parameters, as does sysconfigures, but with the current values rather than the default values. In addition, it contains four rows that describe the configuration structure.

Columns

The columns for syscurconfigs are:

| Name | Datatype | Description |
|---|---|---|
| config | smallint | Configuration parameter number. |
| value | int | The current run value for the parameter with integer datatype. Its value is 0 for the parameters with character datatype. |
| comment | varchar(255) | Comments about the configuration parameter. For internal use.. |
| status | int | Value that represents the type of configuration parameter. See Table 1-4. |
| value2 | varchar(255) null | The current run value for the parameter with the character datatype. Its value is NULL for parameters with the integer datatype. |
| defvalue | varchar(255) null | Default value of the configuration parameter. |
| minimum_value | int null | Minimum value of the configuration parameter. |
| maximum_value | int null | Maximum value of the configuration parameter. |
| memory_used | int null | Integer value for the amount of memory used by each configuration parameter. Negative values indicate memory shared. |
| display_level | int null | Display level of the configuration parameter. The values are 1, 5, and 10. |
| datatype | int null | Datatype of the configuration parameter. |
| message_num | int null | Message number of the sp_helpconfig message for this parameter. |
| apf_percent | int null | The current run value for the asynchronous prefetch percent for a buffer pool. Valid only for rows that represent buffer pools. |
| nodeid | tinyint null | Reserved for future use (not available in cluster environments) |

| Name | Datatype | Description |
|---|---|---|
| instanceid | tinyint | ID of the instance (available only for cluster environments) |
| type | varchar(10) null | Specifies whether a configuration parameter is declared dynamic or static in its structure definition. Values are:<br>• Dynamic – takes effect immediately.<br>• Static – takes effect after restarting Adaptive Server. |

- Not applicable – parameter has no units
- Number – number of items
- Clock ticks – number of clock ticks
- Microseconds
- Milliseconds
- Seconds
- Minutes
- Hours
- Days
- Bytes
- Kilobytes
- Megabytes
- Memory pages (2K)
- Virtual pages (2K)
- Logical pages
- Percent
- Ratio
- Switch – a Boolean value
- ID – ID number
- Name
- Rows

***Table 1-4: Status column description***

| Status type | Decimal | Hex | Description |
|---|---|---|---|
| CFG_NO_OPTIONS | 0 | 0x0 | Parameter has no options. |
| CFG_SYSTEM_OPTION | 1 | 0x01 | Parameter is a system option. |
| CFG_SYSTEM_GROUP | 2 | 0x02 | Parameter is a system group. |
| CFG_STATIC | 4 | 0x04 | Parameter is static. |
| CFG_DYNAMIC | 8 | 0x08 | Parameter is dynamic. |
| CFG_CALCULATED | 16 | 0x10 | Parameter is calculated. |
| CFG_READONLY | 32 | 0x20 | Parameter is read-only. |
| CFG_MEMORY_USED | 64 | 0x40 | Parameter consumes memory. |
| CFG_CONFIG_FILE | 128 | 0x80 | Parameter is externally visible. |
| CFG_SYSTEM_TAB | 256 | 0x100 | Parameter is only externally visible in system table. |
| CFG_EXTRAS_OPTION | 512 | 0x200 | Parameter is for CFG_EXTRAS not DS_CONFIG. |
| CFG_CFGBLK | 1024 | 0x400 | Parameter is stored in the configuration block. |
| CFG_CACHE_GROUP | 2048 | 0x800 | Parameter is a cache group. |
| CFG_CACHE_OPTION | 4096 | 0x1000 | Parameter is a cache option. |
| CFG_BUFFER_POOL_GROUP | 8192 | 0x2000 | Parameter is a buffer pool group. |
| CFG_BUFFER_POOL_OPTION | 16384 | 0x4000 | Parameter is a buffer pool option. |
| CFG_INTERNAL | 32768 | 0x8000 | Parameter is for internal use only. |
| CFG_FNOF_LPAGESIZE | 65536 | 0x10000 | Parameter entry depends on logical pagesize. |

# sysdatabases

## master database only

Description

sysdatabases contains one row for each database in Adaptive Server. When Adaptive Server is installed, sysdatabases contains entries for the master database, the model database, the sybsystemprocs database, and the tempdb database. If you have installed auditing, it also contains an entry for the sybsecurity database.

Columns

The columns for sysdatabases are:

| Name | Datatype | Description |
|---|---|---|
| name | sysname | Name of the database |
| dbid | smallint | Database ID |
| suid | int | Server user ID of Database Owner |
| status | smallint | Control bits; those that the user can set with sp_dboption are so indicated in Table 1-5 |
| version | smallint | Unused |
| logptr | int | Pointer to transaction log |
| crdate | datetime | Creation date |
| dumptrdate | datetime | Date of the last dump transaction |
| status2 | smallint null | Additional control bit (see Table 1-6 on page 36) |
| audflags | int null | Audit settings for database |
| deftabaud | int null | Bit-mask that defines default audit settings for tables |
| defvwaud | int null | Bit-mask that defines default audit settings for views |
| defpraud | int null | Bit-mask that defines default audit settings for stored procedures |
| def_remote_type | smallint null | Identifies the default object type to be used for remote tables if no storage location is provided via the stored procedure sp_addobjectdef |
| def_remote_loc | varchar(349) null | Identifies the default storage location to be used for remote tables if no storage location is provided via the stored procedure sp_addobjectdef |
| status3 | int null | Additional control bits |
| status4 | int null | Additional control bits |
| audflags2 | varbinary(16) null | Reserved for future use |
| instanceid | tinyint | ID of the instance (Cluster Edition only) |
| durability | int | Durability level of the database. Values are:<br>1 – full<br>5 – at_shutdown<br>6 – no_recovery |
| lobcomp_lvl | tinyint | LOB compression level |

Table 1-5 lists the bit representations for the status column.

***Table 1-5: Status control bits in the sysdatabases table***

| Decimal | Hex | Status |
|---|---|---|
| 1 | 0x01 | Upgrade started on this database |
| 2 | 0x02 | Upgrade has been successful |
| 4 | 0x04 | select into/bulkcopy; can be set by user |
| 8 | 0x08 | trunc log on chkpt; can be set by user |
| 16 | 0x10 | no chkpt on recovery; can be set by user |
| 32 | 0x20 | Database created with for load option, or crashed while loading database, instructs recovery not to proceed |
| 64 | 0x04 | Recovery started for all databases to be recovered |
| 256 | 0x100 | • Database suspect<br>• Not recovered<br>• Cannot be opened or used<br>• Can be dropped only with dbcc dbrepair |
| 512 | 0x200 | ddl in tran; can be set by user |
| 1024 | 0x400 | read only; can be set by user |
| 2048 | 0x800 | dbo use only; can be set by user |
| 4096 | 0x1000 | single user; can be set by user |
| 8192 | 0x2000 | allow nulls by default; can be set by user |

Table 1-6 lists the bit representations for the status2 column.

***Table 1-6: status2 control bits in the sysdatabases table***

| Decimal | Hex | Status |
|---|---|---|
| 1 | 0x0001 | abort tran on log full; can be set by user |
| 2 | 0x0002 | no free space acctg; can be set by user |
| 4 | 0x0004 | auto identity; can be set by user |
| 8 | 0x0008 | identity in nonunique index; can be set by user |
| 16 | 0x0010 | Database is offline |
| 32 | 0x0020 | Database is offline until recovery completes |
| 64 | 0x0040 | The table has an auto identity feature, and a unique constraint on the identity column |
| 128 | 0x0080 | Database has suspect pages |
| 256 | 0x0100 | Table structure written to disk. If this bit appears after recovery completes, server may be under-configured for open databases. Use sp_configure to increase this parameter. |
| 512 | 0x0200 | Database is in the process of being upgraded |
| 1024 | 0x0400 | Database brought online for standby access |
| 2048 | 0x0800 | When set by the user, prevents cross-database access via an alias mechanism |
| -32768 | 0xFFFF8000 | Database has some portion of the log which is not on a log-only device |

Table 1-7 lists the bit representations for the status3 column.

***Table 1-7: status3 control bits in the sysdatabases table***

| Decimal | Hex | Status |
|---|---|---|
| 0 | 0x0000 | A normal or standard database, or a database without a proxy update in the create statement. |
| 1 | 0x0001 | You specified the proxy_update option, and the database is a user-created proxy database. |
| 2 | 0x0002 | Database is a proxy database created by high availability. |
| 4 | 0x0004 | Database has a proxy database created by high availability. |
| 8 | 0x0008 | Disallow access to the database, since database is being shut down. |
| 16 | 0x0010 | Database is a failed-over database. |
| 32 | 0x0020 | Database is a mounted database of the type master. |
| 64 | 0x0040 | Database is a mounted database. |
| 128 | 0x0080 | Writes to the database are blocked by the quiesce database command. |
| 256 | 0x0100 | User-created tempdb. |
| 512 | 0x0200 | Disallow external access to database in the server in failed-over state. |
| 1024 | 0x0400 | User-provided option to enable or disable asynchronous logging service threads. Enable through sp_dboption enbale async logging service option set to true on a particular database. |
| 4096 | 0x1000 | Database has been shut down successfully. |
| 8192 | 0x2000 | A drop database is in progress. |

Table 1-8 lists the bit representations for the status4 column.

***Table 1-8: status4 control bits in the sysdatabases table***

| Decimal | Hex | Status |
|---|---|---|
| 512 | 0x0200 | The in-memory database has a template database with it. |
| 4096 | 0x1000 | Database is an in-memory databases. |
| 16384 | 0x4000 | 64-bit atomic operations have been enabled on this database. |
| 16777216 | 0x01000000 | All tables in the database are created as page compressed. |
| 33554432 | 0x02000000 | All tables in the database are created as row compressed. |

Indexes

- Unique clustered index on name
- Nonclustered index on dbid

# sysdepends

## All databases

Description — sysdepends contains one row for each procedure, view, or table that is referenced by a procedure, view, or trigger.

Columns — The columns for sysdepends are:

| Name | Datatype | Description |
|---|---|---|
| id | int | Object ID. |
| number | smallint | Procedure number. |
| depid | int | Dependent object ID. |
| depnumber | smallint | Dependent procedure number. |
| status | smallint | Internal status information. |
| selall | bit | On if object is used in select * statement. |
| resultobj | bit | On if object is being updated. |
| readobj | bit | On if object is being read. |
| columns | varbinary | Stores a bitmap of column IDs of columns that are referenced in the body of a stored procedure.This bitmap gives column-level dependency tracking information, including predicated privileges, for compiled objects, and is decoded by sp_depends to report on column-level dependencies for stored procedures, triggers, and views. |

Indexes

- Unique clustered index on id, number, depid, depnumber

# sysdevices

## master database only

Description

sysdevices contains one row for each tape dump device, disk dump device, disk for databases, and disk partition for databases. There are four entries in sysdevices in the Adaptive Server distribution media: one for the master device (for databases), one for a disk dump device, and two for tape dump devices.

**Note** With Adaptive Server version 15.0, the device identification number is stored in the vdevno column and no longer as part of the high or low column. As a consequence, you may need to modify scripts and stored procedures that determine the device identification number based on the earlier schema.

Columns

The columns for sysdevices are:

| Name | Datatype | Description |
|---|---|---|
| low | int | (Not used for dump devices) Block offset of virtual page in 2K bytes |
| high | int | Block offset of last virtual page in 2K bytes |
| status | smallint | Bitmap indicating type of device, default, and mirror status (see Table 1-9) |
| cntrltype | smallint | Controller type:<br>• 0 = Database device<br>• 2 = Disk dump device or streaming tape<br>• 3–8 = Tape dump device |
| name | sysname | Logical name of dump device, database device, or in-memory storage cache |
| phyname | varchar(127) | Name of physical device or in-memory storage cache |
| mirrorname | varchar(127) null | Name of mirror device |
| vdevno | int | Device identification number |
| crdate | datetime null | Date on which the device was added |
| resizedate | datetime null | Date on which disk resize was most recently run for this device |
| status2 | int | Additional status bits for this device (see Table 1-10) |
| instanceid | tinyint | ID of the instance (available only for cluster environments) |
| uuid | varbinary(16) | Reserved for future use (available only for cluster environments) |

The bit representations for the status column, shown below, are additive. For example, “3” indicates a physical disk that is also a default.

***Table 1-9: Bit representations for the status column.***

| Decimal | Hex | Status |
|---|---|---|
| 1 | 0x01 | Default disk |
| 2 | 0x02 | Physical disk |

| Decimal | Hex | Status |
|---|---|---|
| 4 | 0x04 | *Not used* – logical disk |
| 8 | 0x08 | Skip header |
| 16 | 0x10 | Dump device |
| 32 | 0x20 | Serial writes |
| 64 | 0x40 | Device mirrored |
| 128 | 0x80 | Reads mirrored |
| 256 | 0x100 | Secondary mirror side only |
| 512 | 0x200 | Mirror enabled |
| 1024 | 0x400 | Master device is mirrored |
| 2048 | 0x800 | *Used internally* – mirror disabled |
| 4096 | 0x1000 | *Used internally* – primary device must be unmirrored |
| 8192 | 0x2000 | *Used internally* – secondary device must be unmirrored |
| 16384 | 0x4000 | UNIX file device uses dsync setting (writes flushed to physical media) |

Table 1-10 shows the bit representations for the status2 column.

***Table 1-10: Bit representations for the status2 column.***

| Decimal | Hex | Status |
|---|---|---|
| 1 | 0x01 | Direct I/O is enabled for this device |

Indexes

- Unique clustered index on name

# sysencryptkeys

## All databases

Description — Each key created in a database, including the default key, has an entry in the database-specific system catalog sysencryptkeys.

Columns — The columns for sysencryptkeys are:

| Field | Type | Description |
|---|---|---|
| id | int | Encryption key ID. |
| ekalgorithm | int | Encryption algorithm. |
| type | smallint | Identifies the key type. The values are:<br>• 0x1 (decimal 1) – Symmetric key<br>• 0x4 (decimal 4) – Default key<br>• 0x10 (decimal 16) – Key copy<br>• 0x40 (decimal 64) – Recovery key copy |
| status | int | Internal status information. The bit representations are:<br>• 0x1 (decimal 1) – Key uses initialization vector<br>• 0x2 (decimal 2) – Key uses random pad<br>• 0x4 (decimal 4) – Key is encrypted for lost password protection<br>• 0x8 (decimal 8) – Key copy encrypted for login access<br>• 0x10 (decimal 16) – Key copy encrypted with login password<br>• 0x20 (decimal 32) – Key copy encrypted with system encryption password<br>• 0x100 (decimal 256) – Key encrypted with user password |
| eklen | smallint | User-specified length of key. |
| value | varbinary(1282) | Encrypted value of a key. Contains a symmetric encryption of the key. To encrypt keys, Adaptive Server uses AES with a 128-bit key from the system encryption, user-specified, or login password. |
| uid | int null | User ID of key copy assignee. |
| eksalt | varbinary(20) | Random values used to validate decryption of the encryption key. |
| ekpairid | int null | Not used. |
| pwdate | datetime null | Date the password was last changed. |
| expdate | int null | Not used. |
| ekpwdwarn | int null | Not used. |

The status bits for sysencryptkeys.

***Table 1-11: sysencryptkeys status bits***

| Decimal | Hex | Status |
|---|---|---|
| | 0x00000004 | EK_KEYRECOVERY() – keys encrypted for lost password protection. |

| Decimal | Hex | Status |
|---|---|---|
| | 0x00000008 | EK_LOGINACCESS() – key encrypted for login access |
| | 0x00000010 | EK_LOGINPASS () – key encrypted with login password |
| | 0x00000100 | EK_USERPWD() – keys encrypted with user-encryption passwords |

# sysengines

## master database only

Description sysengines contains one row for each Adaptive Server engine currently online.

Columns The columns for sysengines are:

| Name | Datatype | Description |
|---|---|---|
| engine | smallint | Engine number |
| osprocid | int | • Process mode – operating system process ID<br>• Threaded mode – operating system thread (LWP) ID |
| osprocname | char(32) | Operating system process name (may be NULL) |
| status | char(12) | One of: online, in offline, in create, in destroy, debug, bad status |
| affinitied | int | Number of Adaptive Server processes with affinity to this engine |
| cur_kpid | int | Kernel process ID of process currently running on this engine, if any |
| last_kpid | int | Kernel process ID of process that previously ran on this engine |
| idle_1 | tinyint | Reserved |
| idle_2 | tinyint | Reserved |
| idle_3 | tinyint | Reserved |
| idle_4 | tinyint | Reserved |
| starttime | datetime | Date and time engine came online |
| nodeid | tinyint null | Reserved for future use (not available for cluster environments) |
| instanceid | tinyint | ID of the instance (available only for cluster environments) |

# sysgams

## All databases

Description

sysgams stores the global allocation map (GAM) for the database. The GAM stores a bitmap for all allocation units of a database, with one bit per allocation unit. You cannot select from or view sysgams.

# sysindexes

## All databases

Description

sysindexes contains one row for each clustered index, one row for each nonclustered index, one row for each table that has no clustered index, and one row for each table that contains text or image columns.This table also contains one row for each function-based index or index created on a computed column.

Columns

The columns for sysindexes are:

| Name | Datatype | Description |
|---|---|---|
| name | varchar(255) null | Index or table name. |
| id | int | ID of an index, or ID of table to which index belongs. |
| indid | smallint | Valid values are:<br>• 0 = if a table.<br>• 1 = if a clustered index on an allpages-locked table.<br>• >1 = if a nonclustered index or a clustered index on a data-only-locked table.<br>• 255 = if text, image, text chain, or Java off-row structure (large object—or LOB—structure). |
| doampg | int | Obsolete |
| ioampg | int | Obsolete |
| oampgtrips | int | Number of times OAM pages cycle in the cache without being reused, before being flushed |
| status3 | smallint | Internal system status information. |
| status2 | smallint | Internal system status information (see Table 1-13) |
| ipgtrips | int | Number of times index pages cycle in the cache, without being reused, before being flushed |
| first | int | Obsolete |
| root | int | Obsolete |
| distribution | int | Unused. Formerly used to store the page number of the distribution page for an index. |
| usagecnt | smallint | Reserved |
| segment | smallint | Number of segment in which object resides |
| status | smallint | Internal system status information (see Table 1-12) |
| maxrowsperpage | smallint | Maximum number of rows per page |
| minlen | smallint | Minimum size of a row |
| maxlen | smallint | Maximum size of a row |
| maxirow | smallint | Maximum size of a non-leaf index row |

| Name | Datatype | Description |
|---|---|---|
| keycnt | smallint | Number of keys for a clustered index on an allpages-locked table; number of keys, plus 1 for all other indexes |
| keys1 | varbinary(255) null | Description of key columns if entry is an index |
| keys2 | varbinary(255) null | Description of key columns if entry is an index |
| soid | tinyint | Sort order ID with which the index was created; 0 if there is no character data in the keys |
| csid | tinyint | Character set ID with which the index was created; 0 if there is no character data in the keys |
| base_partition | int null | Obsolete |
| fill_factor | smallint null | Value for the fillfactor of a table set with sp_chgattribute |
| res_page_gap | smallint null | Value for the reservepagegap on a table |
| exp_rowsize | smallint null | Expected size of data rows |
| keys3 | varbinary(255) null | Description of key columns if entry is an index |
| identitygap | int null | Identity gap for a table |
| crdate | datetime null | Creation date |
| partitiontype | smallint null | Values are:<br>• 1 = range<br>• 2 = hash<br>• 3 or NULL = [default] round robin<br>• 4 = list |
| conditionid | int null | ID of the partition condition. Null if partitiontype is round-robin or hash |

Table 1-12 lists the bit representations for the status column.

***Table 1-12: Status bits in the sysindexes table status column***

| Decimal | Hex | Status |
|---|---|---|
| 1 | 0x1 | Abort current command or trigger if attempt to insert duplicate key. |
| 2 | 0x2 | Unique index. |
| 4 | 0x4 | Abort current command or trigger if attempt to insert duplicate row; always 0 for data-only-locked tables. |
| 16 | 0x10 | Table is an all-pages-locked table with a clustered index. |
| 64 | 0x40 | Index allows duplicate rows, if an allpages-locked table; always 0 for data-only-locked tables. |
| 128 | 0x80 | Sorted object toggle that is being used internally. Can be set by create clustered index, reorg rebuild, or alter table locking scheme commands. |
| 512 | 0x200 | sorted data option used in create index statement. |

| Decimal | Hex | Status |
|---|---|---|
| 2048 | 0x800 | Index on primary key. |
| 32768 | 0x8000 | Suspect index; index was created under another sort order. |

Table 1-13 lists the bit representations for the status2 column.

***Table 1-13: Status bits in the sysindexes table status2 column***

| Decimal | Hex | Status |
|---|---|---|
| 1 | 0x1 | Index supports foreign-key constraint |
| 2 | 0x2 | Index supports primary key/unique declarative constraint |
| 4 | 0x4 | Index includes an IDENTITY column |
| 8 | 0x8 | Constraint name not specified |
| 16 | 0x10 | Large I/Os (prefetch) not enabled for table, index, or text chain |
| 32 | 0x20 | Most recently used (MRU) cache strategy not enabled for table, index, or text chain |
| 64 | 0x40 | Ascending inserts turned on for the table |
| 256 | 0x0100 | Index is presorted and does not need to be copied to new extents |
| 512 | 0x0200 | Index is a DOL clustered index |
| 8192 | 0x2000 | Index on a data-only-locked table is suspect |
| 32768 | 0x8000 | The index is function-based |

Indexes

- Unique clustered index on id, indid

# sysinstances

Description

A fake table that reports on the state of the instances. sysinstances includes a row for each instance defined in the cluster configuration. sysinstances contains information specific to the Cluster Edition.

Although sysinstances is a fake table, it is not impacted by the setting of set system_view, and always returns a row for each instance, regardless of the system_view setting.

Columns

The columns for sysinstances are.

| Column name | Datatype | Description |
|---|---|---|
| id | tiny int | ID of the instance |
| name | varchar(30) | Name of the instance |
| state | char(17) | State of the instance (one of `online`, `offline`, `joining`, `leaving`, and `initiating`) |
| hostname | varchar(255) | Name of the operating system host running this instance |
| starttime | datetime | Date and time the instance started |
| connections_active | int | Number of active connections on the instance |
| engines_online | smallint | Number of online engines for this instance |

Indexes

None

# sysjars

## All databases

Description

sysjars contains one row for each Java archive (JAR) file that is retained in the database.

For more information about JAR files, Java classes, and Java datatypes, see *Java in Adaptive Server Enterprise*.

Columns

The columns for sysjars are:

| Name | Datatype | Description |
|---|---|---|
| jid | int | The ID of the JAR. |
| jstatus | int | Internal status information. Unused. |
| jname | varchar(255) null | The JAR name. |
| jbinary | image null | The contents of the JAR: the Java classes. |

Indexes

- Unique clustered index on jid
- Unique nonclustered index on jname

# syskeys

## All databases

Description

syskeys contains one row for each primary, foreign, or common key.

Columns

The columns for syskeys are:

| Name | Datatype | Description |
|---|---|---|
| id | int | Object ID |
| type | smallint | Record type. Valid values are:<br>1 = primary key<br>2 = foreign key<br>3 = common key |
| depid | int null | Dependent object ID |
| keycnt | int null | Number of non-null keys |
| size | int null | Reserved |
| key1 ... key8 | smallint null | Column ID |
| depkey1 ... depkey8 | smallint null | Column ID |
| spare1 | smallint | Reserved |

Indexes

- Clustered index on id

# syslanguages

## master database only

Description

syslanguages contains one row for each language known to Adaptive Server. us_english is not in syslanguages, but it is always available to Adaptive Server.

Columns

The columns for syslanguages are:

| Name | Datatype | Description |
|---|---|---|
| langid | smallint | Unique language ID |
| dateformat | char(3) | Date order; for example, “dmy” |
| datefirst | tinyint | First day of the week—1 for Monday, 2 for Tuesday, and so on, up to 7 for Sunday |
| upgrade | int | Adaptive Server version of last upgrade for this language |
| name | varchar(30) | Official language name, for example, “french” |
| alias | varchar(30) null | Alternate language name, for example, “français” |
| months | varchar(251) | Comma-separated list of full-length month names, in order from January to December—each name is at most 20 characters long |
| shortmonths | varchar(119) | Comma-separated list of shortened month names, in order from January to December—each name is at most 9 characters long |
| days | varchar(216) | Comma-separated list of day names, in order from Monday to Sunday—each name is at most 30 characters long |

Indexes

- Unique clustered index on langid
- Unique nonclustered index on name
- Unique nonclustered index on alias

# syslisteners

## master database only

Description

syslisteners contains a row for each network protocol available for connecting with the current Adaptive Server. Adaptive Server builds syslisteners dynamically when a user or client application queries the table.

Columns

The columns for syslisteners are:

| Name | Datatype | Description |
|---|---|---|
| net_type | char(32) | Network protocol |
| address_info | char(255) | Information that uniquely identifies this Adaptive Server on the network; usually the name of the current Adaptive Server and an identifying number, such as the server's port number for the protocol |
| spare | tinyint | Unused |
| nodeid | tinyint null | Reserved for future use (not available for cluster environments) |
| instanceid | tinyint | ID of the instance (available only for cluster environments) |

# syslocks

## master database only

Description — syslocks contains information about active locks, and built dynamically when queried by a user. No updates to syslocks are allowed.

Columns — The columns for syslocks are:

| Name | Datatype | Description |
|---|---|---|
| id | int | Table ID. |
| dbid | smallint | Database ID. |
| page | unsigned int | Page number. |
| type | smallint | Type of lock (bit values for the type column are listed in Table 1-14). |
| spid | smallint | ID of process that holds the lock. |
| | int for the Cluster Edition | |
| class | varchar(30) | Name of the cursor this lock is associated with, if any. |
| fid | smallint | The family (coordinating process and its worker processes) to which the lock belongs. fid values are:<br>• 0 – the task represented by the spid is a single task executing a statement in serial<br>• Nonzero – the task (spid) holding the lock is a member of a family executing a statement in parallel.<br>If the value is equal to the spid, it indicates that the task is the coordinating process in a family executing a query in parallel. |
| | int for the Cluster Edition | |
| context | tinyint | Context type of lock request. context values are listed in Table 1-15. |
| row | smallint | Row number. |
| loid | int | Unique lock owner ID. |
| partitionid | int null | Patition ID. |
| nodeid | tinyint null | Reserved for future use (not available for cluster environments) |
| instanceid | tinyint | ID of the instance (available only for cluster environments) |

Table 1-14 lists the bit representations for the type column.

***Table 1-14: type control bits in the syslocks table***

| Decimal | Hex | Status |
|---|---|---|
| 1 | 0x1 | Exclusive table lock |
| 2 | 0x2 | Shared table lock |
| 3 | 0x3 | Exclusive intent lock |
| 4 | 0x4 | Shared intent lock |
| 5 | 0x5 | Exclusive page lock |
| 6 | 0x6 | Shared page lock |
| 7 | 0x7 | Update page lock |
| 8 | 0x8 | Exclusive row lock |
| 9 | 0x9 | Shared row lock |
| 10 | 0xA | Update row lock |
| 11 | 0xB | Shared next key lock |
| 256 | 0x100 | Lock is blocking another process |
| 512 | 0x200 | Demand lock |

Table 1-15 lists the values for the context column:

***Table 1-15: context column values in the syslocks table***

| Value | Interpretation |
|---|---|
| null | The task holding this lock is either executing a query in serial, or it is a query being executed in parallel in transaction isolation level 1. |
| 0x1 | The task holding the lock will hold the lock until the query is complete. A lock's context may be FAM_DUR (0x1H) when the lock is:<br>• A table lock held as part of a parallel query.<br>• Held by a worker process at transaction isolation level 3.<br>• Held by a worker process in a parallel query and must be held for the duration of the transaction. |
| 0x2 | Range lock held by serializable read task. |
| 0x4 | Infinity key lock. |
| 0x8 | Lock acquired on an index pages of an allpages-locked table. |
| 0x10 | Lock on a page or row acquired to delete a row. |
| 0x20 | Address lock acquired on an index page during a shrink or split operation. |
| 0x40 | Intent lock held by a transaction performing repeatable reads. Valid only for shared intent and exclusive intent locks on data-only-locked tables. |

# sysloginroles

## master database only

Description

sysloginroles contains a row for each instance of a server login or login profile possessing a role. One row is added for each role granted to each login. For example, if a single server user is granted sa_role, sso_role, and oper_role, three rows are added to sysloginroles associated with that user's system user ID (suid).

Columns

The columns for sysloginroles are:

| Name | Datatype | Description |
|---|---|---|
| suid | int | Server user ID or login profile ID |
| srid | int | Server role ID; one of the following:<br>• 0 = sa_role<br>• 1 = sso_role<br>• 2 = oper_role<br>• 4 = navigator_role<br>• 5 = replication_role<br>• 6 = Currently unused<br>• 7 = dtm_tm_role<br>• 8 = ha_role<br>• 8 = ha_role<br>• 9 = Used internally<br>• 10 = mon_role<br>• 11 = js_admin_role<br>• 12 = messaging_role<br>• 13 = js_client_role<br>• 14 = js_user_role<br>• 15 = webservices_role |
| status | smallint | Status bit that indicates whether the various server roles are set to their defaults at login:<br>• 0 = default off<br>• 1 = default on |

**Note** When you change the status bit using alter login, you must log out and relog for the change to take effect. To see immediate results, use set role role_name off.

Indexes

- Clustered index on suid

# syslogins

## master database only

Description — syslogins contains one row for each valid Adaptive Server user account or login profile.

Columns — The columns for syslogins are:

| Name | Datatype | Description |
|---|---|---|
| suid | int | Server user ID or login profile ID. |
| status | smallint | Status of the account (see Table 1-16). |
| accdate | datetime | Date totcpu and totio were last cleared. |
| totcpu | int | CPU time accumulated by login. |
| totio | int | I/O accumulated by login. |
| spacelimit | int | Reserved. |
| timelimit | int | Reserved. |
| resultlimit | int | Reserved. |
| dbname | sysname null | Name of database in which to put user when connection established. Column is not applicable for a login row if a login profile is associated with the login account. |
| name | sysname | Login name of user. |
| password | varbinary(128) null | One-way hash of user password. The contents of syslogins.password depend on the value for sp_passwordpolicy allow password downgrade. |
| language | varchar(30) null | User's default language. If a login profile is associated with the login account, this column is not applicable for a login row. |
| pwdate | datetime null | Date the password was last changed. |
| audflags | int null | User's audit settings. One of:<br>• 0x00000001 – successful reference to a user-created table<br>• 0x00000002 – failure<br>• 0x00000004 – successful reference to a user-created view<br>• 0x00000008 – failure<br>• 0x00000010 – user cmdtext auditing<br>• 0x00000020 – required padding<br>• 0x00000040 – all successful user action auditing<br>• 0x00000080 – all failed user action auditing |
| fullname | varchar(30) null | Full name of the user. |
| srvname | varchar(30) null | Name of server to which a passthrough connection must be established if the AUTOCONNECT flag is turned on. |
| logincount | smallint null | Number of failed login attempts; reset to 0 by a successful login. |

| Name | Datatype | Description |
|---|---|---|
| procid | int null | Stores the login trigger registered with the login script. If a login profile is associated with the login account, this column is not applicable for a login row. |
| lastlogindate | datetime | Timestamp for the user's last login. |
| crdate | datetime | Timestamp when the login or login profile was created. |
| locksuid | int | The server user ID (suid) responsible for locking the login. |
| lockreason | int | Reasons for lock; one of:<br>NULL – account has not been locked<br>0 – locked by locksuid by executing sp_locklogin<br>1 – inactive account locked by executing sp_locklogin 'all', 'lock', 'ndays'<br>2 – Adaptive Server locked the account because number of failed login attempts reached max failed logins.<br>3 – locked by locksuid because the password downgrade period has ended and a login or role was not transitioned to SHA-256<br>4 – automatically locked by locksuid due to inactivity. |
| lockdate | datetime | If:<br>• The login account is locked – syslogins.lockdate specifies the timestamp when the login was locked.<br>• The login account is not locked, and:<br>  • syslogins.lockdate is non-NULL – specifies the timestamp when the login was unlocked.<br>  • syslogins.lockdate is NULL – specifies that the login was never locked. |
| crsuid | int | Server user ID of the creator of login or login profile. |
| lpid | int | Login profile ID. One of:<br>• null – login account is associated with default login profile, if any<br>• -1– login profile is ignored for login account.<br>• suid – the login profile ID. |

On the Adaptive Server distribution media, syslogins contains an entry in which the name is "sa", the suid is 1, and the password is null. It also contains the entry "probe" with an unpublished password. The login "probe" and the user "probe" exist for the two-phase commit probe process, which uses a challenge and response mechanism to access Adaptive Server.

***Table 1-16: status control bits in the syslogins table***

| Decimal | Hex | Status |
|---|---|---|
| 2 | 0x2 | Account is locked. |
| 4 | 0x4 | Password has expired. |
| 8 | 0x8 | Indicates that the value of exempt inactive lock is set to TRUE. It is not applicable for login profile rows. |
| 16 | 0x10 | OMNI:autoconnect mode is enabled. |

| Decimal | Hex | Status |
|---|---|---|
| 32 | 0x20 | May use Adaptive Server internal authentication mechanism – syslogins. |
| 64 | 0x40 | May use LDAP external authentication. |
| 128 | 0x80 | May use PAM external authentication. |
| 256 | 0x100 | May use Kerberos external authentication. |
| 512 | 0x200 | Indicates a login profile. |
| 1536 | 0x200 \| 0x400 | Indicates a default login profile. |
| 2048 | 0x800 | Indicates an authentication mechanism specified in a login profile. |

Indexes

- Unique clustered index on suid
- Unique nonclustered index on name

# syslogs

## All databases

Description

syslogs contains the transaction log. It is used by Adaptive Server for recovery and roll forward. It is not useful to users.

You cannot delete from, insert into, or update syslogs. Every data modification operation is logged, so before you can change syslogs, the change must be logged. This means that a change operation on syslogs adds a row to syslogs, which then must be logged, adding another row to syslogs, and so on, producing an infinite loop. The loop continues until the database becomes full.

Columns

The columns for syslogs are:

| Name | Datatype | Description |
|---|---|---|
| xactid | binary(6) | Transaction ID |
| op | tinyint | Number of update operation |

# syslogshold

## master database only

Description

syslogshold contains information about each database's oldest active transaction (if any) and the Replication Server truncation point (if any) for the transaction log, but it is not a normal table. Rather, it is built dynamically when queried by a user. No updates to syslogshold are allowed.

Columns

The columns for syslogshold are:

| Name | Datatype | Description |
|---|---|---|
| dbid | smallint | Database ID. |
| reserved | int | Unused. |
| spid | smallint | Server process ID of the user that owns the oldest active transaction (always 0 for Replication Server). |
| | int for cluster environments | |
| page | unsigned int | Starting page number of active portion in syslogs defined by oldest transaction (or the truncation page in syslogs for Replication Server). |
| xactid | binary(6) | ID of the oldest active transaction (always 0x000000 for Replication Server). |
| masterxactid | binary(6) | ID of the transaction's master transaction (if any) for multidatabase transactions; otherwise 0x000000 (always 0x000000 for Replication Server). |
| starttime | datetime | Date and time the transaction started (or when the truncation point was set for Replication Server). |
| name | char(67) | Name of the oldest active transaction. It is the name defined with begin transaction, "$user_transaction" if no value is specified with begin transaction, or "$chained_transaction" for implicit transactions started by the ANSI chained mode. Internal transactions started by Adaptive Server have names that begin with the dollar sign ($) and are named for the operation, or are named "$replication_truncation_point" for Replication Server. |
| xloid | int null | Lock ownership ID based on spid if the owner is a task, or on xdes if the owner is a transaction. |

**Note** Because of this change in the datatypes for the Cluster Edition, Sybase strongly recommends that you archive and truncate audit tables before you upgrade. This reduces the likelihood of a failed upgrade because of insufficient space in the sybsecurity database.

# sysmessages

## master database only

Description

sysmessages contains one row for each system error or warning that can be returned by Adaptive Server. Adaptive Server displays the error description on the user's screen.

Columns

The columns for sysmessages are:

| Name | Datatype | Description |
|---|---|---|
| error | int | Unique error number |
| severity | smallint | Severity level of error |
| dlevel | smallint | Reserved |
| description | varchar(1024) | Explanation of error with placeholders for parameters |
| langid | smallint null | Language; null for us_english |
| sqlstate | varchar(5) null | SQLSTATE value for the error |

Indexes

- Clustered index on error, dlevel
- Nonclustered index on error, dlevel, langid

# sysmonitors

## master database only

Description sysmonitors contains one row for each monitor counter.

Columns The columns for sysmonitors are:

| Name | Datatype | Description |
|---|---|---|
| field_name | char(79) | Name of the counter |
| group_name | char(25) | Group to which this counter belongs |
| field_id | smallint | Unique identifier for the row |
| value | int | Current value of the counter |
| description | varchar(255) null | Description of the counter; not used |
| nodeid | tinyint null | Reserved for future use (not available for cluster environments) |
| instanceid | tinyint | ID of the instance (available only for cluster environments) |

# sysobjects

## All databases

Description

sysobjects contains one row for each table, view, stored procedure, extended stored procedure, log, rule, default, trigger, check constraint, referential constraint, computed column, function-based index key, encryption key, predicated privilege, and (in tempdb only) temporary object, and other forms of compiled objects. It also contains one row for each partition condition ID when object type is N.

For cross-database key references, syscolumns.encrdate matches sysobjects.crdate.

encrkeyid in *sysencryptkeys* matches the id column in *sysobjects*.

Columns

The columns for sysobjects are:

| Name | Datatype | Description |
|---|---|---|
| name | varchar(255) not null | Object name. |
| id | int | Object ID. |
| uid | int | User ID of object owner. |
| type | char(2) | One of the following object types:<br>• C – computed column<br>• D – default<br>• DD – decrypt default<br>• EK – encryption key<br>• F – SQLJ function<br>• N – partition condition<br>• P – Transact-SQL or SQLJ procedure<br>• PP – the predicate of a privilege<br>• PR – prepare objects (created by Dynamic SQL)<br>• R – rule<br>• RI – referential constraint<br>• RS – precomputed result set<br>• S – system table<br>• TR – trigger<br>• U – user table<br>• V – view<br>• XP – extended stored procedure. |

| Name | Datatype | Description |
|---|---|---|
| userstat | smallint | Application-dependent type information (32768 decimal [0x8000 hex] indicates to Data Workbench® that a procedure is a report). |
| sysstat | smallint | Internal status information (256 decimal [0x100 hex] indicates that table is read-only) |
| indexdel | smallint | Recounts the changes in the schema of an object and and updates schemacnt. |
| schemacnt | smallint | Count of changes in the schema of an object (incremented if a rule or default is added) |
| sysstat2 | int | Additional internal status information (see Table 1-18) |
| systat3 | unsigned smallint | Additional internal status information (see Table 1-19) |
| crdate | datetime | Date the object was created |
| expdate | datetime | Reserved |
| deltrig | int | Stored procedure ID of a delete trigger if the entry is a table. Table ID if the entry is a trigger. |
| instrig | int | Stored procedure ID of a table's insert trigger if the entry is a table |
| updtrig | int | Stored procedure ID of a table's update trigger if the entry is a table |
| seltrig | int | Reserved |
| ckfirst | int | ID of first check constraint on the table |
| cache | smallint | Reserved |
| audflags | int null | Object's audit settings |
| objspare | smallint | Spare |
| versionts | binary(6) null | The version timestamp of the last schema change for this object (used by Replication Server) |
| loginame | varchar(30) null | Login name of the user who created the object |
| identburnmax | numeric(17) null | Maximum burned value for identity column if any in this object<br>**Note** The identburnmax column is stored in an internal format. Use the identity_burn_max() function if you need a value. |
| spacestate | smallint null | For internal use only |
| erlchgts | binary(8) null | For internal use only |
| lobcomp_lvl | tinyint | LOB compression level |

Table 1-17 lists the bit representations for the sysstat column:

***Table 1-17: sysstat control bits in the sysobjects table***

| Decimal | Hex | Description |
|---|---|---|
| 0 | 0x0 | Any illegal object |
| 1 | 0x1 | System object |

| Decimal | Hex | Description |
|---|---|---|
| 2 | 0x2 | View |
| 3 | 0x3 | User object |
| 4 | 0x4 | Stored procedure |
| 5 | 0x5 | Predicated privilege |
| 6 | 0x6 | Default value spec |
| 7 | 0x7 | Domain rule |
| 8 | 0x8 | Trigger procedure |
| 9 | 0x9 | Referential integrity constraint |
| 10 | 0xA | SQL Function |
| 11 | 0xB | Extended type |
| 12 | 0xC | Stored function |
| 13 | 0xD | Computed column |
| 14 | 0xE | Partition condition |
| 15 | 0xF | Encryption key |
| 16 | 0x10 | Has clustered index |
| 32 | 0x20 | Has nonclustered index |
| 64 | 0x40 | If the object is a table, changes to the object are logged. If the object is a procedure, indicates that replication can subscribe to executions of the procedure. |
| 128 | 0x80 | The object is being created |
| 256 | 0x100 | The object contains suspect indexes and can only be used for read-only purposes until you have run dbcc reindex. |
| 512 | 0x200 | The object flagged by recovery as possibly damaged; run dbcc. Checked by opentable. |
| 1024 | 0x400 | The object is “fake”; that is, it resides in tempdb and is redefined for every query step that uses it |
| 2048 | 0x800 | The object is a definition time object created for query compilation. |
| 4096 | 0x1000 | Tags a system table that will have its index(es) re-created. |
| 8192 | 0x2000 | The object contains text/image fields |
| 16384 | 0x4000 | Unused |
| 32768 | 0x8000 | The table or procedure is replicated |

Table 1-18 lists the bit representations for the sysstat2 column:

***Table 1-18: sysstat2 control bits in the sysobjects table***

| Decimal | Hex | Status |
|---|---|---|
| 0 | 0x00 | Unchained transaction mode. |
| 1 | 0x1 | Table has a referential constraint. |
| 2 | 0x2 | Table has a foreign-key constraint. |
| 4 | 0x4 | Table has more than one check constraint. |

| Decimal | Hex | Status |
|---|---|---|
| 8 | 0x8 | Table has a primary-key constraint. |
| 16 | 0x10 | Stored procedure can execute only in chained transaction mode. |
| 32 | 0x20 | Stored procedure can execute in any transaction mode. |
| 64 | 0x40 | Table has an IDENTITY field. |
| 128 | 0x80 | Object is s virtually hashed table. |
| 256 | 0x100 | Allow implicit grant in execute immediate calls inside the stored procedure (Dynamic ownership chain). |
| 512 | 0x200 | Table does not contain variable-length columns. |
| 1024 | 0x400 | Table is remote. |
| 2048 | 0x800 | Table is a proxy table created with the existing keyword. |
| 4096 | 0x1000 | Object should be replicated with owner name. |
| 8192 | 0x2000 | Table uses allpages locking scheme. |
| 16384 | 0x4000 | Table uses datapages locking scheme. |
| 32768 | 0x8000 | Table uses datarows locking scheme. |
| 65536 | 0x10000 | Table was created in a version 11.9 or later server. |
| 131072 | 0x20000 | Table has a clustered index. |
| 262144 | 0x40000 | Object represents an Embedded SQL procedure. |
| 524288 | 0x80000 | Hybrid table. |
| 16777216 | 0x1000000 | An access rule. |
| 33554432 | 0x2000000 | Object represents a SQLJ stored procedure. |
| 67108864 | 0x4000000 | Object represents an OR access rule. |
| 1073741824 | 0x40000000 | Table contains one or more function-based indexes. |
| 2147483648 | 0x80000000 | Object has an extended index |

Table 1-19 lists the bit representations for the sysstat4 column:

***Table 1-19: sysstat3 control bits in the sysobjects table***

| Decimal | Hex | Status |
|---|---|---|
| 256 | 0x0100 | Stored procedure created with execute as owner clause |
| 512 | 0x0200 | Stored procedure created with execute as caller clause |
| 2048 | 0x0800 | Table contains LOB compressed data |
| 4096 | 0x1000 | Table uses row-level compression |
| 8192 | 0x2000 | Table uses page-level compression |
| 16384 | 0x4000 | Table contains compressed data |
| 32768 | 0x8000 | Table participates in incremental transfer |

Indexes

- Unique clustered index on id
- Nonclustered index on name, uid

# sysoptions

## All databases

Description

sysoptions is a fake table queried by sp_options. When you are querying sysoptions, the names of the rows are case sensitive.

Columns

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| spid | int | | Contains the process ID. |
| name | varchar(100) | | Contains the name of the option. |
| category | varchar(100) | | Contains the name of the category to which the option belongs. |
| currentsetting | varchar(100) | NULL | Contains the current setting of the option. |
| defaultsetting | varchar(100) | NULL | Contains the default setting of the option. |
| scope | int | | Contains the bitmap used to capture information about options. The bits are ordered as follows:<br>• Bit 1 – compiled time options<br>• Bit 2 – stored procedure specific options<br>• Bit 3 – binary options |
| number | int | | The switch ID as an integer. |

sysoptions shows:

- Trace flags set in the runserver file with the –T options
- Trace flags set with dbcc traceon(*flag_number*) or set switch serverwide on
- Trace flags and switches set by a specific system process ID (SPID) using set switch on

sysoptions displays only the switches that are visible to the user querying the sysoptions table. That is, the user cannot see switches set privately by other SPIDs with set switch on. However, traceflags enabled using the runserver file –T option, dbcc traceon, or set switch serverwide on are visible to all users.

Query sysoptions using sp_options. The datatype for the current and default value is varchar so settings with varchar values can be used directly. Settings with integer values can be used after typecasting.

You do not need special privileges to query sysoptions. For example:

```
select * from sysoptions
where spid = 13
go
```

You can also use string manipulation or typecasting. For example, if an option is numeric, you can query sysoptions by entering:

```
if (isnumeric(currentsetting))
    select@int_val = convert(int, currentsetting)
    ...
else
    select@char_val = currentsetting
    ...
```

# syspartitionkeys

## All databases

Description

syspartitionkeys contains one row for each partition key for hash, range, and list partitioning of a table. All columns are not null.

Columns

The columns for syspartitionkeys are:

| Name | Datatype | Description |
|---|---|---|
| indid | smallint | Type of index. Values are:<br>• 0 = table<br>• 1 = clustered index<br>• >1 = nonclustered index |
| id | int | Object ID of the partitioned table |
| colid | smallint | Column ID of the partition key of the partitioned table |
| position | smallint | Position of key among key positions |

Indexes

- Unique clustered index on id, indid, colid

# syspartitions

## All databases

Description

syspartitions contains one row for each data partition and one row for each index partition.

For each database, syspartitions contains one row for:

- Each table partition. indid is 0.
- Each clustered index partition. indid is 1.
- Each nonclustered index partition. indid is >1.
- Each single-partitioned (unpartitioned) table.
- Each single-partitioned (unpartitioned) clustered or nonclustered index.

If an index is local, the value for partitionid (data partition row) and data_partitionid (associated index row) are the same. If the index is not local, the value for data_partitionid (index row) is zero (0), and it does not equal that for partitionid (data partition row).

**Note** The syspartitions table in versions of Adaptive Server earlier than 15.0 has been renamed sysslices and made obsolete. With Adaptive Server version 15.0, syspartitions is completely redefined, and now supports data and index partitioning.

Columns

The columns for syspartitions are:

| Name | Datatype | Description |
|---|---|---|
| name | varchar(255) | Partition name. |
| indid | smallint | on an allpages-locked table<br>Index ID. Values are:<br>• 0 = data pages (table)<br>• 1 = clustered index on an allpages-locked table<br>• >1 and <255 = nonclustered index or a clustered index on a data-only-locked table<br>• 255 = text chain |
| id | int | Table ID. |
| partitionid | int | ID of data or index partition. |
| segment | smallint | ID of segment on which partition resides. |
| status | int | Internal status information. |
| datoampage | unsigned int | Page number for the object allocation map of a data partition. |

| Name | Datatype | Description |
|---|---|---|
| indoampage | unsigned int | Page number of the object allocation map of an index partition. |
| firstpage | unsigned int | Page number of the first data or leaf page. |
| rootpage | unsigned int | Page number of:<br>• Root page if entry is an index partition<br>• Last page if entry is a data partition |
| data_partitionid | int | ID of data partition this index spans. Values are:<br>• 0 = for global indexes spanning the entire table<br>• 1 = partition ID of the data partition that a local index's partition spans. |
| crdate | datetime | Date the partition created. |
| cdataptnname | varchar(255) null | Name of data partition. |
| lobcomp_lvl | tinyint | LOB compression level |
| ptndcompver | tinyint | Version of datacompression algorithm used |

Indexes

- Unique clustered index on id, indid, partitionid
- Unique nonclustered index on id, indid, name
- Unique nonclustered index on partitionid, indid

# syspoolinfo

## master database

Description — Provides information about data caches and pools.

Access to the views is restricted to users with the sa_role role.

Columns — The columns for syspoolinfo are:

| Name | Datatype | Description |
|---|---|---|
| cache_name | varchar(30) | Name of the cache in which this pool is allocated. |
| io_size | varchar(3) | The size of the buffers, in kilobytes, used to perform I/O for this pool. |
| config_size | float | Configured amount of memory, in megabytes, allocated to the pool. May be different from the amount reported in the run_size column. |
| run_size | float | The current amount of memory, in megabytes, allocated to the pool. |
| apf_percent | int | The percentage of buffers in the pool that can be used to hold buffers that have been read into cache by asynchronous prefetch. |
| wash_size | varchar(10) | The size of the wash area, in megabytes, in the pool. |
| cacheid | int | ID of the data cache. |
| instanceid | int | ID of the instance (zero for non-Cluster Edition servers). |
| scope | varchar(6) | Indicates whether the data cache is local or global for Cluster Edition. The value is always Global for nonclustered servers. |

# sysprocedures

## All databases

Description

sysprocedures contains entries for each view, default, rule, trigger, procedure, declarative default, partition condition, check constraint, comuted column, function-based index key, and other forms of compiled objects. The sequence tree for each object, including computed columns or function-based index definition, is stored in binary form. If the sequence tree does not fit into one entry, it is broken into more than one row. The sequence column identifies the sub-rows.

Columns

The columns for sysprocedures are:

| Name | Datatype | Description |
|---|---|---|
| type | smallint | Object type (see Table 1-20) |
| qp_setting | varbinary(6) null | For future use only |
| id | int | Object ID |
| sequence | int | Sequence number if more than one row is used to describe this object |
| status | smallint | Internal system status |
| number | smallint | Sub-procedure number when the procedure is grouped (0 for nonprocedure entries) |
| version | int null | The version of Adaptive Server that created the sequence tree stored in this catalog for a given object |

Table 1-20 lists the bit representations for the type column.

***Table 1-20: type control bits in the sysprocedures table***

| Decimal | Hex | Status |
|---|---|---|
| 1 | 0x1 | Entry describes a plan (reserved) |
| 2 | 0x2 | Entry describes a tree |

Indexes

- Unique clustered index on id, number, type, sequence

# sysprocesses

## master database only

Description

sysprocesses contains information about Adaptive Server processes, but it is not a normal table. It is built dynamically when queried by a user. No updates to sysprocesses are allowed. Use the kill statement to kill a process.

Columns

The columns for sysprocesses are:

| Name | Datatype | Description |
|---|---|---|
| spid | smallint | Process ID. |
| | int for the Cluster Edition | |
| kpid | int | Kernel process ID. |
| enginenum | int | Number of engine on which process is being executed. |
| status | char(12) | Process ID status (see Table 1-21). |
| suid | int | Server user ID of user who issued command. |
| hostname | varchar(30) null | Name of host computer. |
| program_name | varchar(30) null | Name of front-end module. |
| hostprocess | varchar(30) null | Host process ID number.. |
| cmd | varchar(30) null | Command or process currently being executed. Evaluation of a conditional statement, such as an if or while loop, returns `cond`. |
| cpu | int | Cumulative CPU time for process in ticks |
| physical_io | int | Number of disk reads and writes for current command. |
| memusage | int | Amount of memory allocated to process. |
| blocked | smallint | Process ID of blocking process, if any. |
| | int for the Cluster Edition | |
| dbid | smallint | Database ID. |
| uid | int | ID of user who executed command. |
| gid | int | Group ID of user who executed command. |
| tran_name | varchar(64) null | Name of the active transaction. |
| time_blocked | int null | Time blocked in seconds. |
| network_pktsz | int null | Current connection's network packet size. |
| fid | smallint | Process ID of the worker process' parent. |
| | int for the Cluster Edition | |
| execlass | varchar(30) null | Execution class that the process is bound to. |
| priority | varchar(10) null | Base priority associated with the process. |
| affinity | varchar(30) null | Name of the engine to which the process has affinity. |
| id | int null | Object ID of the currently running procedure (or 0 if no procedure is running). |
| stmtnum | int null | The current statement number within the running procedure (or the SQL batch statement number if no procedure is running). |

| Name | Datatype | Description |
|---|---|---|
| linenum | int null | The line number of the current statement within the running stored procedure (or the line number of the current SQL batch statement if no procedure is running). |
| origsuid | int null | Original server user ID. If this value is not NULL, a user with an suid of origsuid executed set proxy or set session authorization to impersonate the user who executed the command. |
| block_xloid | int null | Unique lock owner ID of a lock that is blocking a transaction. |
| clientname | varchar(30) null | (Optional) Name by which the user is know for the current session.<br>**Note** Adaptive Server automatically stores one or more spaces in clientname, clienthostname, and clientapplname columns. For this reason, a query using any of these three columns that includes "`is null`" does not return an expected result set. |
| clienthostname | varchar(30) null | Optional – name by which the host is known for the current session. |
| clientapplname | varchar(30) null | Optional – name by which the application is known for the current session. |
| sys_id | smallint null | Unique identity of companion node. |
| ses_id | int null | Unique identity of each client session. |
| loggedindatetime | datetime null | Shows the time and date when the client connected to Adaptive Server. See "Row-level access control" in Chapter 11, "Managing User Permissions" of the *Security Administration Guide* for more information. |
| ipaddr | varchar(64) null | IP address of the client where the login is made. See "Row-level access control" in Chapter 11, "Managing User Permissions" of the *Security Administration Guide* for more information. |
| nodeid | tinyint null | Reserved for future use (not available for cluster environments). |
| instanceid | tinyint | ID of the instance (available only for cluster environments). |
| pad | smallint | (Cluster Edition) Column added for alignment purposes. |
| lcid | int | (Cluster Edition) ID of the cluster. |

**Note** Because of this change in the datatypes for the Cluster Edition, Sybase strongly recommends that you archive and truncate audit tables before you upgrade. This reduces the likelihood of a failed upgrade because of insufficient space in the sybsecurity database.

Table 1-21 lists the values for the status column:

***Table 1-21: sysprocesses status column values***

| Status | Meaning |
|---|---|
| alarm sleep | Waiting for alarm to wake process up (user executed a waitfor delay command) |
| background | A process, such as a threshold procedure, run by Adaptive Server rather than by a user process |

| Status | Meaning |
|---|---|
| infected | Server has detected a serious error condition; extremely rare |
| latch sleep | Waiting on a latch acquisition |
| lock sleep | Waiting on a lock acquisition |
| PLC sleep | Waiting to access a user log cache |
| recv sleep | Waiting on a network read |
| remote i/o | Performing I/O with a remote server |
| runnable | In the queue of runnable processes |
| running | Actively running on one of the server engines |
| send sleep | Waiting on a network send |
| sleeping | Waiting on a disk I/O, or some other resource (often indicates a process that is running, but doing extensive disk I/O) |
| stopped | Stopped process |
| sync sleep | Waiting on a synchronization message from another process in the family |

# sysprotects

## All databases

Description — sysprotects contains information on permissions that have been granted to, or revoked from, users, groups, and roles.

Columns — The columns for sysprotects are:

| Name | Datatype | Description |
|---|---|---|
| id | int | ID of the object to which this permission applies. Has an ID of 0 when the permission granted is create table, create default, and so on. |
| uid | int | ID of the user, group, or role to which this permission applies. |
| action | smallint | See Table 1-22 for permissions. |
| protecttype | tinyint | One of the following values:<br>0 = grant with grant<br>1 = grant<br>2 = revoke |
| columns | varbinary(133) | Bitmap of columns to which this select, update, decrypt, or references permission applies. The bits indicate the following:<br>0 = indicates all columns.<br>1 = means permission applies to that column.<br>NULL = means no information.<br>columns is also a bitmap of permitted roles for set session authorization. |
| grantor | int | User ID of the grantor. If the grantor is a system administrator, the user ID of the object owner is used. |
| predid | int | Object ID of predicated privileg e |
| status | smallint | 0x0001 – indicates that the privilege (or denial) is predicated |

***Table 1-22: sysprotects action column values***

1 = alter any object owner *
2 = alter any table *
3 = change password *
4 = checkpoint any database *
5 = select builtin
6 = checkpoint *
7 = create any default *
8 = create any function *
9 = create any index *
10 = create any object *
11 = create any procedure *
12 = create any rule *
13 = create any table *
14 = create any trigger *
15 = create any view *
16 = allow exceptional login *
17 = dbcc checkalloc
18 = dbcc checkalloc any database
19 = map external file *
20 = manage dump configuration *
21 = dbcc checkcatalog
22 = dbcc checkcatalog any database
25 = dbcc checkdb
26 = dbcc checkdb any database
29 = dbcc checkindex
30 = dbcc checkindex any database
33 = dbcc checkstorage
34 = dbcc checkstorage any database
37 = dbcc checktable
38 = dbcc checktable any database
41 = dbcc checkverify
42 = dbcc checkverify any database
45 = dbcc fix_text
46 = dbcc fix_text any database
49 = dbcc indexalloc
50 = dbcc indexalloc any database
53 = dbcc reindex
54 = dbcc reindex any database
57 = dbcc tablealloc
58 = dbcc tablealloc any database
61 = dbcc textalloc
62 = dbcc textalloc any database
65 = dbcc tune
66 = delete any table *
67 = drop any default *
68 = drop any function *
70 = drop any object *
71 = drop any procedure *
72 = drop any rule *
73 = drop any table *
74 = drop any trigger *
75 = drop any view *
76 = dump database *
77 = dump any database *
79 = execute any function *
80 = execute any procedure *
80 = identity_insert any table *
82 = identity_update any table *
81 = identity_insert *
82 = identity_update *
85 = insert any table *
86 = kill *
87 = kill any process *
88 = load database *
89 = load any database *
90 = manage service key *
91 = manage abstract plans *
92 = manage any encryption key *
93 = manage any esp *
94 = manage any execution class *
95 = manage any login *
96 = manage any login profile *
97 = manage any object permission *
98 = manage any remote login *
99 = manage any statistics *
100 = manage any user *
101 = manage auditing *
102 = manage checkstorage *
103 = manage cluster *
104 = manage data cache *
105 = manage database *
106 = manage database permissions *
107 = manage disk *
108 = manage lock promotion threshold *
109 = manage master key *
110 = manage replication *
111 = manage resource limit *
112 = manage roles *

113 = manage security configuration *
114 = manage security permissions *
115 = manage server *
116 = manage server configuration *
117 = manage server permissions *
118 = monitor qp performance *
119 = monitor server replication *
120 = mount any database *
121 = online any database *
122 = online database *
123 = own any database *
125 = own database *
126 = quiesce any database *
129 = references any table *
130 = report checkstorage *
131 = reorg any table *
132 = select any audit table *
133 = select any system catalog *
134 = select any table *
135 = set tracing any process *
136 = setuser
137 = shutdown *
138 = transfer any table *
139 = manage any thread pool *
140 = truncate any table *
141 = unmount any database *
144 = update any security catalog *
145 = update any table *
146 = use any database *
148 = use database *
149 = set switch *
150 = show switch *
151 = references
152 = truncate any audit table *
153 = decrypt any table *
155 = manage column encryption key *
156 = manage any database *
167 = set proxy
193 = select
195 = insert
196 = delete
197 = update
198 = create table
203 = create database
207 = create view
221 = create trigger
222 = create procedure
224 = execute
233 = create defaulr
235 = dump transaction
236 = create rule
253 = connect
280 = create function
282 = delete statistics
320 = truncate table
326 = update statistics
347 = set tracing
353 = decrypt
354 = create encryption key
368 = transfer table

**Note** Permissions for the action column marked with an asterisk (*) take effect only when granular permissions is enabled.

Indexes

- Unique clustered index on id, action, grantor, uid, protecttype, predid

# sysquerymetrics

## All databases

Description — Presents aggregated historical query processing metrics for individual queries from persistent data. In addition to monitoring tables, use performance metrics information from this catalog.

Columns — The columns for sysquerymetrics are:

| Name | Datatype | Description |
|---|---|---|
| uid | int | User ID |
| gid | int | Group ID |
| hashkey | int | Hashkey over the SQL query text |
| id | int | Unique ID |
| sequence | smallint null | Sequence number for a row when multiple rows are required for the text of the SQL |
| exec_min | int null | Minimum execution time |
| exec_max | int null | Maximum execution time |
| exec_avg | int null | Average execution time |
| elap_min | int null | Minimum elapsed time |
| elap_max | int null | Maximum elapsed time |
| elap_avg | int null | Average elapsed time |
| lio_min | int null | Minimum logical IO |
| lio_max | int null | Maximum logical IO |
| lio_avg | int null | Average logical IO |
| pio_min | int null | Minimum physical IO |
| pio_max | int null | Maximum physical IO |
| pio_avg | int null | Average physical IO |
| cnt | int null | Number of times the query has been executed. |
| abort_cnt | int null | Number of times a query is aborted by the Resource Governor when a resource limit is exceeded |
| qtext | varchar(255) null | Query text |

The number of metrics shared among user IDs increased for Adaptive Server release 15.0.2 and later, reducing the number of entries in sysquerymetrics (a view of sysqueryplans), and automatically aggregates the metrics for identical queries across different user IDs.

The user ID (uid) of sysquerymetrics is 0 when all table names in a query that are not qualified by user name are owned by the DBO.

For example, if table t1 is owned only by the DBO and shared by different users:

```
select * from t1 where c1 = 1
```

Adaptive Server uses 0 as the uid for the sysquerymetrics entry for all users executing this query who do not have a private table named t1.

In this example, if table t2 is owned and qualified by "user1," Adaptive Server also uses an UID of 0:

```
selet * from user1.t2 where c1 = 1
```

However, if table t3 is owned only by "user1," but is unqualified and not owned by the DBO, the UID of "user1" is used in the sysquerymetrics entry:

```
select * from t3 where c1 = 1
```

# sysqueryplans

## All databases

Description — sysqueryplans contains two or more rows for each abstract query plan. Uses datarow locking.

Columns — The columns for sysqueryplans are:

| Name | Datatype | Description |
|---|---|---|
| uid | int | User ID of user who captured the abstract plan. |
| dbid | int null | For future use only |
| qpdate | datetime null | For future use only |
| sprocid | int null | For future use only |
| hashkey2 | int null | For future use only |
| key1 | int null | For future use only |
| key2 | int null | For future use only |
| key3 | int null | For future use only |
| gid | int | The abstract plan group ID under which the abstract plan was saved. |
| hashkey | int | The hash key over the SQL query text. |
| id | int | The unique ID if the abstract plan. |
| type | smallint | 10 if the text column contains query text, or 100 if the text column contains abstract plan text. |
| sequence | smallint | Sequence number if multiple rows are required for the text of the SQL query or abstract plan. |
| status | int null | Reserved. |
| text | varchar(255) null | The SQL text, if type is 10, or the abstract query plan text, if the type is 100. |

Indexes

- Unique clustered index on uid, gid, hashkey, id, type, sequence
- Nonclustered index on id, type, sequence

# sysreferences

## All databases

Description — sysreferences contains one row for each referential integrity constraint declared on a table or column.

Columns — The columns for sysreferences are:

| Name | Datatype | Description |
|---|---|---|
| indexid | smallint | ID of the unique index on referenced columns |
| constrid | int | Object ID of the constraint from sysobjects |
| tableid | int | Object ID of the referencing table |
| reftabid | int | Object ID of the referenced table |
| keycnt | smallint | Number of columns in the foreign key |
| status | smallint | Options and indicators |
| frgndbid | smallint null | Database ID of the database that includes the referencing table. |
| pmrydbid | smallint | Database ID of the database that includes the referenced table (the table with the primary key) |
| spare2 | int | Reserved |
| fokey1 ... fokey16 | smallint | Column ID of the first to the 16th referencing column |
| refkey1 ... refkey16 | smallint | Column ID of the first to the 16th referenced column |
| frgndbname | varchar(30) null | Name of the database that includes the referencing table (the table with the foreign key); NULL if the referencing table is in the current database |
| pmrydbname | varchar(30) null | Name of the database that includes the referenced table (the table with the primary key); NULL if the referenced table is in the current database |

The status bit in sysreferences is:

| Decimal | Hex | Status |
|---|---|---|
| 2 | 0x2 | The referential constraint has a match full option |

Indexes

- Clustered index on tableid, frgndbname
- Nonclustered index on constrid, frgndbname
- Nonclustered index on reftabid, indexid, pmrydbname

# sysremotelogins

## master database only

Description

sysremotelogins contains one row for each remote user that is allowed to execute remote procedure calls on this Adaptive Server.

Columns

The columns for sysremotelogins are:

| Name | Datatype | Description |
|---|---|---|
| remoteserverid | smallint | Identifies the remote server |
| remoteusername | varchar(30) null | User's login name on remote server |
| suid | int | Local server user ID |
| status | smallint | Bitmap of options |

Indexes

- Unique clustered index on remoteserverid, remoteusername

# sysresourcelimits

## master database only

Description

sysresourcelimits contains a row for each resource limit defined by Adaptive Server. Resource limits specify the maximum amount of server resources that can be used by an Adaptive Server login or an application to execute a query, query batch, or transaction.

Columns

The columns for sysresourcelimits are:

| Name | Datatype | Description |
|---|---|---|
| name | varchar(30) null | Login name |
| appname | varchar(30) null | Application name |
| rangeid | smallint | id column from systimeranges |
| limitid | smallint | id column from spt_limit_types |
| enforced | tinyint | Subset of the enforced column from spt_limit_types:<br>• 1 = prior to execution<br>• 2 = during execution<br>• 3 = both |
| action | tinyint | Action to take on a violation:<br>• 1 = issue warning<br>• 2 = abort query batch<br>• 3 = abort transaction<br>• 4 = kill session |
| limitvalue | int | Value of limit |
| scope | tinyint | Scope of user limit (a bitmap indicating one or more of the following):<br>• 1 = query<br>• 2 = query batch<br>• 4 = transaction |
| spare | tinyint | Reserved |

Indexes

- Clustered index on name, appname

# sysroles

## All databases

Description

sysroles maps server role IDs to local role IDs.

Columns

The columns for sysroles are:

| Name | Datatype | Description |
|---|---|---|
| id | int | Server role ID (srid) |
| lrid | int | Local role ID |
| type | smallint | Unused |
| status | int | Unused |

When a database permission is granted to a role, if an entry for the role does not exist in sysroles, Adaptive Server adds an entry to sysroles to map the local role ID (lrid) to the server-wide role ID (srid) in syssrvroles.

Indexes

- Unique clustered index on lrid

# syssecmechs

master database only

Description

syssecmechs contains information about the security services supported by each security mechanism that is available to Adaptive Server. syssecmechs is not created during installation, rather, it is built dynamically when queried by a user.

Columns

The columns for syssecmechs are:

| Name | Datatype | Description |
|---|---|---|
| sec_mech_name | varchar(30) | Name of the security mechanism; for example, “NT LANMANAGER” |
| available_service | varchar(30) | Name of the security service supported by the security mechanism; for example, “unified login” |

# syssegments

## All databases

Description

syssegments contains one row for each segment (named collection of disk pieces). In a newly created database, the entries are: segment 0 (system) for system tables; segment 2 (logsegment) for the transaction log; and segment 1 (default) for other objects.

Columns

The columns for syssegments are:

| Name | Datatype | Description |
|---|---|---|
| segment | smallint | Segment number |
| name | sysname | Segment name |
| status | smallint null | Indicates which segment is the default segment |

# sysservers

## master database only

Description — sysservers contains one row for each remote Adaptive Server, Backup Server™, or Open Server™ on which this Adaptive Server can execute remote procedure calls.

Columns — The columns for sysservers are:

| Name | Datatype | Description |
|---|---|---|
| srvid | smallint | ID number (for local use only) of the remote server. |
| srvstatus | smallint | Bitmap of options (see Table 1-23). |
| srvstatus2 | unsigned int | Bitmap of options (see Table 1-24). |
| srvstat2 | unsigned int | Bitmap of server options |
| srvname | varchar(30) | Server name. |
| srvnetname | varchar (255) | Interfaces file name for the server. |
| srvclass | smallint null | Server category defined by the class parameter of sp_addserver (see Table 1-25). |
| srvsecmech | varchar(30) null | Security mechanism. |
| srvcost | smallint null | Provides the network cost in milliseconds for accessing a server over a network. Used only by the Adaptive Server query optimizer for evaluating the cost of a query when accessing a proxy table, the default is set to 1,000 ms. |
| srvprincipal | varchar(255) null | Specifies the Kerberos principal name for the server. Default value is NULL. |

Table 1-23 lists the bit representations for the srvstatus column:

***Table 1-23: Status control bits for srvstatus column***

| Decimal | Hex | Status |
|---|---|---|
| 0 | 0x0 | Timeouts are enabled |
| 1 | 0x1 | Timeouts are disabled |
| 2 | 0x2 | Network password encryption is enabled |
| 4 | 0x4 | Remote server is read-only |
| 64 | 0x40 | Use message confidentiality |
| 128 | 0x80 | Use message integrity |
| 256 | 0x100 | Mutual authentication |

***Table 1-24: srvstatus2 control bits in the sysservers table***

| Decimal | Hex | Status |
|---|---|---|
| 1 | 0x01 | Supports fully qualified table names |
| 2 | 0x02 | Reserved for future use |

Table 1-25 lists the server categories for the srvclass column:

***Table 1-25: Server categories is the sysservers table***

| srvclass | Server category |
|---|---|
| 0 | Local server |
| 1 | sql_server class server |
| 3 | direct_connect class server |
| 4 | DB2 class server |
| 6 | sds class server |
| 7 | Adaptive Server Enterprise class server |
| 8 | Adaptive Server Anywhere class server |
| 9 | ASIQ class server |

Indexes

- Unique clustered index on srvid
- Nonclustered index on srvname

# syssessions

## master database only

Description

syssessions is used only when Adaptive Server is configured for Sybase Failover in a high availability system. syssessions contains one row for each client that connects to Adaptive Server with the failover property. Clients that have an entry in syssessions during failover are moved to the secondary companion. Clients that do not have an entry in syssessions are dropped during failover. Clients that have an entry in syssessions during failback are moved to the primary companion. Clients that do not have an entry in syssessions during failback are dropped.

Columns

The columns for syssessions are:

| Name | Datatype | Description |
|---|---|---|
| sys_id | smallint | Unique identity of companion node |
| ses_id | int | Unique identity of each client session |
| state | tinyint | Describes whether the session is active or inactive |
| spare | tinyint | Reserved for future use |
| status | smallint | Reserved for future use |
| dbid | smallint | Reserved for future use |
| name | varchar(30) null | Same as client's login name as specified in syslogins |
| nodeid | tinyint null | Reserved for future use (not available for cluster environments) |
| instanceid | tinyint | ID of the instance (available only for cluster environments) |
| ses_data | image null | Reserved for future use |

# sysslices

## All databases

Description

sysslices contains one row for each slice (page chain) of a sliced table. sysslices is used only during the Adaptive Server upgrade process. After the upgrade is complete, all the data is removed.

**Note** In versions of Adaptive Server earlier than 15.0 syspartitions was the name of the catalog that stored partition-related *information*. This has been renamed to sysslices for Adaptive Server 15.0, with syspartitions now referring to the catalog that tracks all partition-related *data* in Adaptive Server.

Columns

The columns for sysslices are:

| Name | Datatype | Description |
|---|---|---|
| state | smallint | Internal information about the state of the partition |
| id | int | Object ID of the partitioned table |
| partitionid | int | Partition ID number |
| firstpage | int | Page number of the partition's first page |
| controlpage | int | Page number of the partition's control page |
| spare | binary(32) | Reserved |

Indexes

- Unique clustered index on id, partitionid

# syssrvroles

## master database only

Description — syssrvroles contains a row for each system or user-defined role.

Columns — The columns for syssrvroles are:

| Name | Datatype | Description |
|---|---|---|
| srid | int | Server role ID |
| name | varchar(30) | Name of the role |
| password | varbinary(128) null | Password for the role (encrypted) and readable only by a user with sso_role |
| pwdate | datetime null | Date the password was last changed |
| status | smallint null | Bitmap for role status (see Table 1-26) |
| logincount | smallint null | Number of failed login attempts; reset to 0 by a successful login |
| locksuid | int null | The user who locked the role. |
| lockreason | int null | The reason why a role was locked. |
| lockdate | datetime null | The date and time a role was locked. |

Table 1-26 lists the bit representations for the status column:

***Table 1-26: status control bits in the syssrvroles table***

| Decimal | Hex | Status |
|---|---|---|
| 2 | 0x2 | Role is locked |
| 4 | 0x4 | Role is expired |

Indexes

- Unique clustered index on srid

# sysstatistics

## All databases

Description — sysstatistics contains one or more rows for each indexed column on a user table and for each partition. May also contain rows for unindexed column.

Columns — The columns for sysstatistics are:

| Name | Datatype | Description |
|---|---|---|
| statid | smallint | Reserved |
| id | int | Object ID of table |
| sequence | int | Sequence number if multiple rows are required for this set of statistics |
| moddate | datetime | Date this row was last modified |
| formatid | tinyint | Type of statistics represented by this row |
| usedcount | tinyint | Number of fields c0 to c79 used in this row |
| colidarray | varbinary(100) | An ordered list of column IDs |
| c0...c79 | varbinary(255) | Statistical data |
| indid | smallint | Index ID of partition |
| ststatus | smallint | Status bits for this statistics row; possible values vary according to the type of row. |
| partitionid | int | Partition ID |
| spare2 | smallint | For future use |
| spare3 | int | For future use |

Indexes

- Unique clustered index csysstatistics on id, indid, partitionid, statid, colidarray, formatid, sequence

# systabstats

## All databases

Description — systabstats contains one row for each clustered index, one row for each nonclustered index, one row for each table that has no clustered index, and one row for each partition.

Columns — The columns for systabstats are:

| Name | Datatype | Description |
|---|---|---|
| indid | smallint | • 0 =if a table<br>• 1 = if a clustered index on an allpages-locked table<br>• >1 = if a nonclustered index or a clustered index on a data-only-locked table<br>systabstats does not maintain statistics on text or image objects (255) |
| id | int | ID of table to which index belongs |
| activestatid | smallint | Reserved |
| indexheight | smallint | Height of the index; maintained if indid is greater than 1 |
| leafcnt | unsigned int | Number of leaf pages in the index; maintained if indid is greater than 1 |
| pagecnt | unsigned int | Number of pages in the table or index |
| rowcnt | float | Number of rows in the table; maintained for indid of 0 or 1 |
| forwrowcnt | float | Number of forwarded rows; maintained for indid of 0 or 1 |
| delrowcnt | float | Number of deleted rows |
| dpagecrcnt | float | Number of extent I/Os that need to be performed to read the entire table |
| ipagecrcnt | float | Number of extent I/Os that need to be performed to read the entire leaf level of a nonclustered index |
| drowcrcnt | float | Number of page I/Os that need to be performed to read an entire table |
| oamapgcnt | int | Number of OAM pages for the table, plus the number of allocation pages that store information about the table |
| extent0pgcnt | int | Count of pages that are on the same extent as the allocation page |
| datarowsize | float | Average size of the data row |
| leafrowsize | float | Average size of a leaf row for nonclustered indexes and clustered indexes data-only-locked tables |
| status | int | Internal system status information (see Table 1-27) |
| pljoindegree | int | The degree of parallelism used for a nested loop join operation, pljoindegree is the parallel scan degree of the table (whose systabstats has this field) that is the inner table in a nested loop join. |
| rslastoam | int | Last OAM page visited by a reorg reclaim_space or reorg compact command |
| rslastpage | int | Last data or leaf page visited by a reorg reclaim_space or reorg compact command |

| Name | Datatype | Description |
|---|---|---|
| frlastoam | int | Last OAM page visited by the reorg forwarded_rows command |
| frlastpage | int | Last data page visited by the reorg forwarded_rows command |
| conopt_thld | smallint | Concurrency optimization threshold |
| plldegree | int16 | Maximum degree of parallelism possible on table or index for data manipulation languages (DMLs). A value of 0 (zero) indicates a nonexistent maximum; the query processor configures maximum degree of parallelism. |
| emptypgcnt | unsigned int | Number of empty pages in extents allocated to the table or index |
| partitionid | int | Partition ID |
| warmcachepgcnt | unsigned int | |
| statmoddate | datetime | Last time the row was flushed to disk |
| unusedpgcnt | unsigned int | Number of unused pages |
| oampagecnt | unsigned int | Number of allocation pages listed in the object allocation map |
| pioclmdata | real | |
| pioclmindex | real | |
| piocsmdata | real | |
| piocsmindex | real | |
| spare2 | float | Reserved |
| spare4 | float | Reserved |
| spare5 | int | Spare field for alignment |

The status bit for systabstats is:

***Table 1-27: Status bit in the systabstats table status column***

| Decimal | Hex | Status |
|---|---|---|
| 1 | 0x1 | Statistics are the result of upgrade (not update statistics) |

Indexes

- Unique clustered index on id, indid, partitionid

# systhresholds

## All databases

Description — systhresholds contains one row for each threshold defined for the database.

Columns — The columns for systhresholds are:

| Name | Datatype | Description |
|---|---|---|
| segment | smallint | Segment number for which free space is being monitored. |
| free_space | unsigned int | Size of threshold, in logical pages. |
| status | smallint | Bit 1 equals 1 for the log segment's last-chance threshold, 0 for all other thresholds. |
| proc_name | varchar(255) | Name of the procedure that is executed when the number of unused pages on segment falls below free_space. |
| suid | int null | The server user ID of the user who added the threshold or modified it most recently. |
| currauth | varbinary(255) null | A bitmask that indicates which roles were active for suid at the time the threshold was added or most recently modified. When the threshold is crossed, proc_name executes with this set of roles, less any that have been deactivated since the threshold was added or last modified. |

**Determining the active roles from currauth** Rhe possible bitmasks you might see, individually or in combination, in the currauth column.

| Decimal | Hex | Description |
|---|---|---|
| 1 | 0x1 | sa_role |
| 2 | 0x2 | sso_role |
| 4 | 0x4 | oper_role |
| 8 | 0x8 | sybase_ts_role |
| 16 | 0x10 | sybase_ts_role |
| 32 | 0x20 | navigator _ole |
| 128 | 0x80 | replication_role |
| 256 | 0x100 | dtm_tm_role |
| 1024 | 0x400 | ha_role |
| 2048 | 0x800 | mon_role |
| 4096 | 0x1000 | js_admin_role |
| 16384 | 0x4000 | messaging_role |
| 32768 | 0x8000 | web_services |

To find out what role ID is associated with the bitmask output in currauth in your Adaptive Server, perform the following select statement:

```
1> select (c.number - 1) as role_id,role_name(c.number - 1) as role_name
2> from systhresholds ,master.dbo.spt_values c
```

```
3> where   convert(tinyint,substring(isnull(currauth,0x1), c.low,1)) &
4> c.high != 0
5> and c.type = "P"
6> and c.number <= 1024
7> and c.number >0
8> and role_name(c.number - 1) is not null
9> go
```

Adaptive Server returns something similar to the following:

```
 role_id     role_name
----------- ------------------------------
           0  sa_role
           1  sso_role
           2  oper_role
           3  sybase_ts_role
           4  navigator_role
           7  dtm_tm_role
          10  mon_role
          11  js_admin_role
          12  messaging_role
          13  js_client_role
```

# systimeranges

## master database only

Description — systimeranges stores named time ranges, which are used by Adaptive Server to control when a resource limit is active.

Columns — The columns for systimeranges are:

| Name | Datatype | Description |
|---|---|---|
| name | varchar(255) | Unique name of the time range. |
| id | smallint | Unique identifier for the time range. 1 represents the “at all times” limit. |
| startday | tinyint | Day of week (1 – 7) for the beginning of the range. Monday = 1, Sunday = 7. |
| endday | tinyint | Day of week (1 – 7) for the end of the range. Monday = 1, Sunday = 7. |
| starttime | varchar(10) | Time of day for the beginning of the range. |
| endtime | varchar(10) | Time of day for the end of the range. |

Indexes

- Clustered index on id

# systransactions

## master database only

Description

systransactions contains information about Adaptive Server transactions, but it is not a normal table. Portions of the table are built dynamically when queried by a user, while other portions are stored in the master database. Updates to the dynamically-built columns of systransactions are not allowed.

Columns

The columns for systransactions are:

| Name | Datatype | Description |
|---|---|---|
| xactkey | binary(14) | Unique Adaptive Server transaction key |
| starttime | datetime | Date the transaction started |
| failover | int | Value indicating the transaction failover state. Valid values are:<br>0 – Resident Tx<br>1 – Failed-over Tx<br>2 – Tx by Failover-Conn |
| type | int | Value indicating the type of transaction. Valid values are:<br>1 – Local<br>3 – External<br>98 – Remote<br>99 – Dtx_State |
| coordinator | int | Value indicating the coordination method or protocol. Valid values are:<br>0 – None<br>1 – Syb2PC<br>2 – ASTC<br>3 – XA<br>4 – DTC |
| state | int | Value indicating the current state of the transaction (see Table 1-28) |
| connection | int | Value indicating the connection state. The connection values and states are:<br>1 – Attached<br>2 – Detached |
| status | int | Internal transaction status flag |
| status2 | int | Additional internal transaction status flags |
| spid | smallint | Server process ID, or 0 if the process is detached |
| | int for the Cluster Edition | |
| masterdbid | smallint | Starting database of the transaction |
| loid | int | Lock owner ID |
| namelen | smallint | Length of xactname |
| xactname | varchar(255) null | Transaction name or *XID* |
| srvname | varchar(30) null | Name of the remote server (null for local servers) |

| Name | Datatype | Description |
|---|---|---|
| nodeid | tinyint null | Reserved for future use (not available for cluster environments) |
| instanceid | tinyint | ID of the instance (available only for cluster environments) |

**Note** Because of this change in the datatypes for the Cluster Edition, Sybase strongly recommends that you archive and truncate audit tables before you upgrade. This reduces the likelihood of a failed upgrade because of insufficient space in the sybsecurity database.

***Table 1-28: systransactions state column values***

| *state* value | Transaction state |
|---|---|
| 1 | Begun |
| 2 | Done Command |
| 3 | Done |
| 4 | Prepared |
| 5 | In Command |
| 6 | In Abort Cmd |
| 7 | Committed |
| 8 | In Post Commit |
| 9 | In Abort Tran |
| 10 | In Abort Savept |
| 65537 | Begun-Detached |
| 65538 | Done Cmd-Detached |
| 65539 | Done-Detached |
| 65540 | Prepared-Detached |
| 65548 | Heur Committed |
| 65549 | Heur Rolledback |

# systypes

## All databases

Description

systypes contains one row for each system-supplied and user-defined datatype. Domains (defined by rules) and defaults are given, if they exist.

You cannot alter the rows that describe system-supplied datatypes.

Columns

The columns for systypes are:

| Name | Datatype | Description |
|---|---|---|
| uid | int | User ID of datatype creator |
| usertype | smallint | User type ID |
| variable | bit | 1 if datatype is of variable length; 0 otherwise |
| allownulls | bit | Indicates whether nulls are allowed for this datatype |
| type | tinyint | Physical storage datatype |
| length | int | Physical length of datatype |
| tdefault | int | ID of system procedure that generates default for this datatype |
| domain | int | ID of system procedure that contains integrity checks for this datatype |
| name | varchar(255) | Datatype name |
| printfmt | varchar(255) null | Reserved |
| prec | tinyint null | Number of significant digits |
| scale | tinyint null | Number of digits to the right of the decimal point |
| ident | tinyint null | 1 if column has the IDENTITY property; 0 if it does not |
| hierarchy | tinyint null | Precedence of the datatype in mixed-mode arithmetic |
| xtypeid | int null | The internal class ID |
| xdbid | int null | The dbid where a class is installed:<br>• -1 = the system database<br>• -2 = the current database |
| accessrule | int null | The object ID of the access rule in sysprocedures |

Table 1-29 lists each system-supplied datatype's name, hierarchy, type (not necessarily unique), and usertype (unique). The datatypes are ordered by hierarchy. In mixed-mode arithmetic, the datatype with the lowest hierarchy takes precedence:

***Table 1-29: Datatype names, hierarchy, types, and usertypes***

| Name | Hierarchy | Type | Usertype |
|---|---|---|---|
| floatn | 1 | 109 | 14 |
| float | 2 | 62 | 8 |
| datetimn | 3 | 111 | 15 |
| datetime | 4 | 61 | 12 |
| real | 5 | 59 | 23 |
| numericn | 6 | 108 | 28 |
| numeric | 7 | 63 | 10 |
| decimaln | 8 | 106 | 27 |
| decimal | 9 | 55 | 26 |
| moneyn | 10 | 110 | 17 |
| money | 11 | 60 | 11 |
| smallmoney | 12 | 122 | 21 |
| smalldatetime | 13 | 58 | 22 |
| intn | 14 | 38 | 13 |
| uintn | 15 | 68 | 47 |
| bigint | 16 | 191 | 43 |
| ubigint | 17 | 67 | 46 |
| int | 18 | 56 | 7 |
| uint | 19 | 66 | 45 |
| smallint | 20 | 52 | 6 |
| usmallint | 21 | 65 | 44 |
| tinyint | 22 | 48 | 5 |
| bit | 23 | 50 | 16 |
| univarchar | 24 | 155 | 35 |
| unichar | 25 | 135 | 34 |
| unitext | 26 | 174 | 36 |
| varchar | 27 | 39 | 2 |
| sysname | 27 | 39 | 18 |
| nvarchar | 27 | 39 | 25 |
| longsysname | 27 | 39 | 42 |
| char | 28 | 47 | 1 |
| nchar | 28 | 47 | 24 |
| varbinary | 29 | 37 | 4 |
| timestamp | 29 | 37 | 80 |
| binary | 30 | 45 | 3 |
| text | 31 | 35 | 19 |

| Name | Hierarchy | Type | Usertype |
|---|---|---|---|
| image | 32 | 34 | 20 |
| date | 33 | 49 | 37 |
| time | 34 | 51 | 38 |
| daten | 35 | 123 | 39 |
| timen | 36 | 147 | 40 |
| extended type | 99 | 36 | -1 |

Indexes

- Unique clustered index on name
- Unique nonclustered index on usertype

# sysusages

## master database only

Description

sysusages contains one row for each **disk allocation piece** assigned to a database. Each database contains a specified number of database (logical) page numbers.

The create database command checks sysdevices and sysusages to find available disk allocation pieces. One or more contiguous disk allocation pieces are assigned to the database, and the mapping is recorded in sysusages.

See "System tables that manage space allocation" in Chapter 21, "Creating and Managing User Databases" of the *System Administration Guide: Volume 2* for more information on sysusages.

**Note** In Adaptive Server version 15.0 and later, the device identification number is stored in the vdevno column and not as part of the vstart column. As a consequence, you may need to modify scripts and stored procedures that determine the device identification number based on the earlier schema.

Columns

The columns for sysusages are:

| Name | Datatype | Description |
|---|---|---|
| dbid | smallint | Database ID |
| segmap | int | Bitmap of possible segment assignments |
| lstart | unsigned int | First database (logical) page number |
| size | unsigned int | Number of contiguous database (logical) pages |
| vstart | int | Starting virtual page number |
| unreservedpgs | unsigned int | Free space not part of an allocated extent |
| crdate | datetime null | Creation date |
| vdevno | int | Device identification number |

Indexes

- Unique clustered index on dbid, lstart
- Unique nonclustered index on vdevno, vstart

# sysusermessages

## All databases

Description — sysusermessages contains one row for each user-defined message that can be returned by Adaptive Server.

Columns — The columns for sysusermessages are:

| Name | Datatype | Description |
|---|---|---|
| error | varchar(1024) | Unique error number. Must be 20,000 or higher. |
| uid | int | Server user ID (suser_id) of the message creator. |
| description | varchar(1024) | User-defined message with optional placeholders for parameters. |
| langid | smallint null | Language ID for this message; null for us_english. |
| dlevel | smallint null | Stores the with_log bit, which is used to call the appropriate routine to log a message. |

Indexes

- Clustered index on error
- Unique nonclustered index on error, langid

# sysusers

## All databases

Description

sysusers contains one row for each user allowed in the database, and one row for each group or role.

Columns

The columns for sysusers are:

| Name | Datatype | Description |
|---|---|---|
| suid | int | Server user ID, copied from syslogins. |
| uid | int | User ID, unique in this database, is used for granting and revoking permissions. User ID 1 is "dbo". |
| gid | int | Group ID to which this user belongs. If uid = gid, this entry defines a group. Negative values may be used for user IDs (uid). Every suid associated with a group or a role in sysusers is set to -2 (INVALID_SUID). |
| name | sysname | User or group name, unique in this database. |
| environ | varchar(255) null | Reserved. |

On the Adaptive Server distribution media, master..sysusers contains some initial users: "dbo", whose suid is 1 and whose uid is 1; "guest", whose suid is -1 and whose uid is 2; and "public", whose suid is -2 and whose uid is 0. In addition, both system-defined and user-defined roles are listed in sysusers.

The user "guest" provides a mechanism for giving users not explicitly listed in sysusers access to the database with a restricted set of permissions. The "guest" entry in master means any user with an account on Adaptive Server (that is, with an entry in syslogins) can access master.

The user "public" refers to all users. The keyword public is used with the grant and revoke commands to signify that permission is being given to or taken away from all users.

Indexes

- Nonunique clustered index with "allow duplicate rows" on suid
- Unique nonclustered index on name
- Unique nonclustered index on uid

# sysxtypes

## All databases

Description

sysxtypes contains one row for each extended, Java-SQL datatype.

See *Java in Adaptive Server Enterprise* for more information about Java-SQL classes and datatypes.

Columns

The columns for sysxtypes are:

| Name | Datatype | Description |
|---|---|---|
| xtid | int | System-generated ID for the extended type. |
| xtstatus | int | Internal status information. Unused. |
| xtmetatype | int | Unused. |
| xtcontainer | int | The ID of the JAR file containing the class. Can be NULL. |
| xtname | varchar(255) null | The name of the extended type. |
| xtsource | text null | Source code for the extended type. Unused. |
| xtbinaryinrow | varbinary(255) null | Object code for the extended type. For Java classes, it contains the class file. Data is stored in-row up to a length of 255 bytes. |
| xtbinaryoffrow | image | Object code for the extended type. For Java classes, it contains the class file. Data is stored off-row as an image column. |

Indexes

- Unique clustered index on xtid
- Unique nonclustered index on xtname

CHAPTER 2

# *dbccdb* Tables

In addition to the standard system tables included in all databases, the dbcc management database, dbccdb, contains seven tables that define inputs to and outputs from dbcc checkstorage. It also contains at least two workspaces.



## dbccdb workspaces

Workspaces are special tables in dbccdb that store intermediate results of the dbcc checkstorage operation. Workspaces differ from worktables in that they:

- Are preallocated contiguously to improve I/O performance
- Are persistent
- Do not reside in the tempdb database

When you create dbccdb, two workspaces are created automatically. They are preallocated as follows:

- *Scan workspace* – contains a row for each page of the target database. The allocation is approximately 1 percent of the database size. Each row consists of a single binary(18) column.
- *Text workspace* – contains a row for each table in the target database that contains text or image columns. The size of this table depends on the design of the target database, but it is usually significantly smaller than the scan workspace. Each row consists of a single binary(22) column.

If either allocation is larger than needed by dbcc checkstorage, the operation uses only what is required. The allocation does not change. If the text workspace allocation is too small, dbcc checkstorage reports this, recommends a new size, and continues checking; however, not all text chains are checked. If the scan workspace allocation is too small, the dbcc checkstorage operation fails immediately.

You must have at least one scan and one text workspace, but you may create as many as you need. While in use, the workspaces are locked so that only one dbcc checkstorage operation can use them at any given time. You can execute concurrent dbcc checkstorage operations by supplying each one with a separate scan and text workspace.

For more information on creating workspaces, see the *System Administration Guide* and the *Adaptive Server Reference Manual*.

Ideally, you should access workspaces only through dbcc checkstorage, but this is not a requirement. dbcc checkstorage exclusively locks the workspaces it uses, and the content of the workspaces is regenerated with each execution of dbcc checkstorage. The workspaces do not contain any secure data.

---

**Note** While the contents of the workspaces are accessible through SQL, no interpretation of the binary values is available. Access through SQL might return data from different dbcc checks mixed together. The presence of a row in these tables does not ensure that it contains valid data. dbcc tracks valid rows only during execution. That information is lost when the operation completes.

---

Most of the update activity in dbccdb is performed in the text and scan workspaces. The workspaces are preallocated, and only one dbcc checkstorage operation can use the workspaces at any given time, so the workspaces are less susceptible to corruption than most user tables. Corruption in a workspace can cause the dbcc checkstorage operation to fail or behave erratically. If this happens, drop and re-create the corrupt workspace.

Checks of databases using different workspaces can proceed simultaneously, but the performance of each operation might be degraded as it competes for disk throughput.

To delete a workspace, in dbccdb, enter:

```
drop table workspace_name
```

# dbccdb log

The results of each dbcc checkstorage operation are recorded in the dbccdb log. Updates to the text and scan workspaces are not recorded there.

You must size the dbccdb log to handle updates to the tables. The log requirement is related to the number of tables and indexes in the target database. It is not related to the target database size.

To minimize the log requirement and the recovery time, use the truncate log on checkpoint option with dbccdb.

# dbcc_config

Description

The dbcc_config table describes the currently executing or last completed dbcc checkstorage operation. It defines:

- The location of resources dedicated to the dbcc checkstorage operation
- Resource usage limits for the dbcc checkstorage operation

Columns

The columns for dbcc_config are:

| Column name | Datatype | Description |
|---|---|---|
| dbid | smallint | Matches the dbid from a row in sysindatabases. |
| type_code | int | Matches the type_code from a row in the dbcc_types table. Valid values are 1 – 9. |
| value | int null | Specifies the value of the item identified by type_code. Can be null only if the value of stringvalue is not null. |
| stringvalue | varchar(255) null | Specifies the value of the item identified by type_code. Can be null only if the value of value is not null. |

Primary key

Combination of dbid and type_code

See also

For information on initializing and updating dbcc_config, see the *System Administration Guide*.

# dbcc_counters

Description

The dbcc_counters table stores the results of the analysis performed by dbcc checkstorage. Counters are maintained for each database, table, index, partition, device, and invocation of dbcc.

Columns

The columns for dbcc_counters are:

| Column name | Datatype | Description |
|---|---|---|
| dbid | smallint | Identifies the target database. |
| id | int | Identifies the table. The value is derived from sysindexes and sysobjects. |
| indid | smallint | Identifies the index. The value is derived from sysindexes. |
| partitionid | int | Identifies the defined object-page affinity. The value is derived from sysindexes and syspartitions. |
| devid | int | Identifies the disk device. The value is derived from sysdevices. |
| opid | smallint | Identifies the dbcc operation that was performed. |
| type_code | int | Matches the type_code column of a row in the dbcc_types table. Valid values are 5000 through 5024. |
| value | real null | Matches the appropriate type_name for the given type_code as described in dbcc_types. |

Primary key

Combination of dbid, id, indid, partitionid, devid, opid, and type_code

# dbcc_exclusions

Description — The dbcc_exclusions table stores the faults, tables or a combination of them that should be excluded from processing by checkverify and fault reporting via sp_dbcc_faultreport.

Columns — The columns for dbcc_exclusionss are:

| Column name | Datatype | Description |
|---|---|---|
| dbid | smallint | Identifies the target database. |
| type | tinyint | Exclusion type code. The valid values are:<br>• 1 – faults<br>• 2 – tables<br>• 3 – combo |
| fault_type | int null | The fault type to be excluded when type is 1 (faults) or 3 (combo). See "dbcc types" on page 119 for more information. |
| table_name | varchar(30) null | The table name to be excluded when type is 2 (faults) or 3 (combo). See "dbcc types" on page 119 for more information. |

Primary key — Combination of dbid, fault_type, and table_name

# dbcc_fault_params

Description — The dbcc_fault_params table provides additional descriptive information for a fault entered in the dbcc_faults table.

Columns — The columns for dbcc_fault_params are:

| Column name | Datatype | Description |
|---|---|---|
| dbid | smallint | Identifies the target database. |
| opid | smallint | Identifies the dbcc operation that was performed. |
| faultid | int | Identifies the fault ID. |
| type_code | int | Defines the interpretation of the value, which is provided by the "value" columns. Valid values are 1000 – 1009. They are described in dbcc_types. |
| intvalue | int null | Specifies the integer value. |
| realvalue | real null | Specifies the real value. |
| binaryvalue | varbinary(255) null | Specifies the binary value. |
| stringvalue | varchar(255) null | Specifies the string value. |
| datevalue | datetime null | Specifies the date value. |

Each "value" column (intvalue, realvalue, binaryvalue, stringvalue, and datevalue) can contain a null value. At least one must be not null. If more than one of these columns contains a value other than null, the columns provide different representations of the same value.

Primary key — Combination of dbid, opid, faultid, and type_code

# dbcc_faults

Description — The dbcc_faults table provides a description of each fault detected by dbcc checkstorage.

Columns — The columns for dbcc_faults are:

| Column name | Datatype | Description |
|---|---|---|
| dbid | smallint | Identifies the target database. |
| id | smallint | Identifies the table. The value is derived from sysindexes and sysobjects. |
| indid | smallint | Identifies the index. The value is derived from sysindexes. |
| partitionid | int | Identifies the partition. The value is derived from sysindexes and syspartitions. Counters are maintained for page ranges, so "partition" refers to the defined object-page affinity, rather than the actual object page chain. |
| devid | int | Identifies the disk device. The value is derived from sysdevices. |
| opid | smallint | Identifies the dbcc operation that was performed. |
| faultid | int | Provides a unique sequence number assigned to each fault recorded for the operation. |
| type_code | int | Identifies the type of fault. Valid values are 100000 – 100032. They are described in Table 2-1 on page 119. |
| status | int | Classifies the fault. Valid values are:<br>• 0 – soft fault, possibly transient.<br>• 1– hard fault.<br>• 2 – soft fault that proved to be transient.<br>• 3 – soft fault upgraded to a hard fault.<br>• 5 – repaired hard fault.<br>• 7 – repaired upgraded hard fault.<br>• 9 – hard fault not rapirable.<br>• 11 – soft fault upgraded to a hard fault and not repairable.<br>• 16 – soft fault, object dropped (inaccessible).<br>• 17 – hard fault, object dropped (inaccessible).<br>• 18 – transient soft fault, object dropped (inaccessible).<br>• 19 – soft fault upgraded to a hard fault and object dropped (inaccessible).<br>For more information, see the *System Administration Guide*. |

Primary key — Combination of dbid, id, indid, partitionid, devid, opid, faultid, and type_code

# dbcc_operation_log

Description

The dbcc_operation_log table records the use of the dbcc checkstorage operations.

Columns

The columns for dbcc_operaiton_log are:

| Column Name | Datatype | Description |
|---|---|---|
| dbid | smallint | Identifies the target database. |
| opid | smallint | Identifies the sequence number of the dbcc checkstorage operation. opid is an automatically incrementing number, unique for each dbid and finish pair. |
| optype | smallint | The following value is valid for optype:<br>• 2 = checkstorage |
| suid | int | Identifies the user executing the command. |
| start | datetime | Identifies when the operation started. |
| finish | datetime null | Identifies when the operation ended. |
| seq | smallint null | The sequence number for a checkverify operation. |
| id | int null | The object ID, if used, for a checkverify operation. |
| maxseq | smallint null | The maximum sequence used by checkverify for a checkstorage oepration. |

Summary results are recorded in the dbcc_operation_results table.

Primary key

Combination of dbid, opid, and optype

# dbcc_operation_results

Description

The dbcc_operation_results table provides additional descriptive information for an operation recorded in the dbcc_operation_log table.

Columns

The columns for dbcc_operation_results are:

| Column Name | Datatype | Description |
|---|---|---|
| dbid | smallint | Identifies the target database. |
| opid | smallint | Identifies the dbcc operation ID. |
| optype | smallint | Identifies the dbcc operation type. |
| type_code | int | Defines the dbcc operation type. Valid values are 1000 – 1007. They are described in Table 2-1 on page 119. |
| intvalue | int null | Specifies the integer value. |
| realvalue | real null | Specifies the real value. |
| binaryvalue | varbinary(255) null | Specifies the binary value. |
| stringvalue | varchar(255) null | Specifies the string value. |
| datevalue | datetime null | Specifies the date value. |
| seq | smallint null | The sequence number for a checkverify operation. |

Each "value" column (intvalue, realvalue, binaryvalue, stringvalue, and datevalue) may contain a null value. At least one must be not null. If more than one of these columns contains a value other than null, the columns provide different representations of the same value.

Results of the dbcc checkstorage operations include the number of:

- Hard faults found
- Soft faults found
- Operations stopped due to a hard error

Primary key

Combination of dbid, opid, optype, and type_code

# dbcc_types

Description

The dbcc_types table provides the definitions of the datatypes used by dbcc checkstorage. This table is not actually used by the dbcc stored procedures. It is provided to facilitate the use of the other tables in dbccdb, and to document the semantics of the datatypes. Type codes for operation configuration, analysis data reported, fault classification, and fault report parameters are included. If you create your own stored procedures for generating reports, you can use the values listed in the type_name column as report headings.

Columns

The columns for dbcc_types are as follows.

**Note** To allow for future additions to dbcc_types, some type_code numbers are not used at this time.

***Table 2-1: dbcc types***

| *type_code* | *type_name* | Description |
|---|---|---|
| 1 | max worker processes | (Optional) Specifies the maximum number of worker processes that can be employed. This is also the maximum level of concurrent processing used. Minimum value is 1. |
| 2 | dbcc named cache | Specifies the size (in kilobytes) of the cache used by dbcc checkstorage and the name of that cache. |
| 3 | scan workspace | Specifies the ID and name of the workspace to be used by the database scan. |
| 4 | text workspace | Specifies the ID and name of the workspace to be used for text columns. |
| 5 | operation sequence number | Specifies the number that identifies the dbcc operation that was started most recently. |
| 6 | database name | Specifies the name of the database in sysdatabases. |
| 7 | OAM count threshold | Specifies the percentage by which the OAM counts must vary before they can be considered to be an error. |
| 8 | IO error abort | Specifies the number of I/O errors allowed on a disk before dbcc stops checking the pages on that disk. |
| 9 | linkage error abort | Specifies the number of linkage errors allowed before dbcc stops checking the page chains of an object. Some kinds of page chain corruptions might require a check to be stopped with fewer linkage errors than other kinds of page chain corruptions. |
| 10 | enable automatic workspace expansion | The flag that enables or disables automatic expansion of workspaces when estimated size exceeds the actual workspace size. |
| 1000 | hard fault count | Specifies the number of persistent inconsistencies (hard faults) found during the consistency check. |

| *type_code* | *type_name* | Description |
|---|---|---|
| 1001 | soft fault count | Specifies the number of suspect conditions (soft faults) found during the consistency check. |
| 1002 | checks aborted count | Specifies the number of linkage checks that were stopped during the consistency check. |
| 1007 | text column count | Specifies the number of non-null text/image column values found during the consistency check. |
| 5000 | bytes data | Specifies (in bytes) the amount of user data stored in the partition being checked. |
| 5001 | bytes used | Specifies (in bytes) the amount of storage used to record the data in the partition being checked. The difference between bytes used and bytes data shows the amount of overhead needed to store or index the data. |
| 5002 | pages used | Specifies the number of pages linked to the object being checked that are actually used to hold the object. |
| 5003 | pages reserved | Specifies the number of pages that are reserved for the object being checked, but that are not allocated for use by that object. The difference between (8 * extents used) and (pages used + pages reserved) shows the total uncommitted deallocations and pages incorrectly allocated. |
| 5004 | pages overhead | Specifies the number of pages used for the overhead functions such as OAM pages or index statistics. |
| 5005 | extents used | Specifies the number of extents allocated to the object in the partition being checked. For object 99 (allocation pages), this value is the number of extents that are not allocated to a valid object. Object 99 contains the storage that is not allocated to other objects. |
| 5006 | count | Specifies the number of component items (rows or keys) found on any page in the part of the object being checked. |
| 5007 | max count | Specifies the maximum number of component items found on any page in the part of the object being checked. |
| 5008 | max size | Specifies the maximum size of any component item found on any page in the part of the object being checked. |
| 5009 | max level | Specifies the maximum number of levels in an index. This datatype is not applicable to tables. |
| 5010 | pages misallocated | Specifies the number of pages that are allocated to the object, but are not initialized correctly. This is a fault counter. |
| 5011 | io errors | Specifies the number of I/O errors encountered. This datatype is a fault counter. |
| 5012 | page format errors | Specifies the number of page format errors reported. This datatype is a fault counter. |
| 5013 | pages not allocated | Specifies the number of pages linked to the object through its chain, but not allocated. This datatype is a fault counter. |
| 5014 | pages not referenced | Specifies the number of pages allocated to the object, but not reached through its chains. This datatype is a fault counter. |

| *type_code* | *type_name* | **Description** |
|---|---|---|
| 5015 | overflow pages | Specifies the number of overflow pages encountered. This datatype is applicable only to clustered indexes. |
| 5016 | page gaps | Specifies the number of pages not linked to the next page in ascending sequence. This number indicates the amount of table fragmentation. |
| 5017 | page extent crosses | Specifies the number of pages that are linked to pages outside of their own extent. As the number of page extent crosses increases relative to pages used or extents used, the effectiveness of large I/O buffers decreases. |
| 5018 | page extent gaps | Specifies the number of page extent crosses where the subsequent extent is not the next extent in ascending sequence. Maximal I/O performance on a full scan is achieved when the number of page extent gaps is minimized. A seek or full disk rotation is likely for each gap. |
| 5019 | ws buffer crosses | Specifies the number of pages that are linked outside of their workspace buffer cache during the dbcc checkstorage operation. This information can be used to size the cache, which provides high performance without wasting resources. |
| 5020 | deleted rows | Number of deleted rows in the object. |
| 5021 | forwarded rows | Number of forwarded rows in the object. |
| 5022 | empty pages | Number of pages allocated but not containing data. |
| 5023 | pages with garbage | Number of pages that could benefit from garbage collection. |
| 5024 | non-contiguous free space | Number of bytes of noncontiguous free space. |
| 10000 | page id | Specifies the location in the database of the page that was being checked when the fault was detected. All localized faults include this parameter. |
| 10001 | page header | Specifies the hexadecimal representation of the header of the page that was being checked when the fault was detected. This information is useful for evaluating soft faults and for determining if the page has been updated since it was checked. The server truncates trailing zeros. |
| 10002 | text column id | Specifies an 8-byte hexadecimal value that gives the page, row, and column of the reference to a text chain that had a fault. The server truncates trailing zeros. |
| 10003 | object id | Specifies a 9-byte hexadecimal value that provides the object id (table), the partition id (partition of the table) if applicable, and the index id (index) of the page or allocation being checked.<br>For example, if a page is expected to belong to table T1 because it is reached from T1's chain, but is actually allocated to table T2, the object id for T1 is recorded, and the object id expected for T2 is recorded. The server truncates trailing zeros. |

| type_code | type_name | Description |
|---|---|---|
| 10007 | page id expected | Specifies the page ID that is expected for the linked page when there is a discrepancy between the page ID that is expected and the page ID that is actually encountered.<br>For example, if you follow the chain from P1 to P2 when going forward, then, when going backward, P1 is expected to come after P2. The value of page id expected is P1, and the value of page id is P2. When the actual value of P3 is encountered, it is recorded as page id actual. |
| 10008 | page id actual | When there is a discrepancy between the page ID that is encountered and the expected page ID, this value specifies the actual page ID that is encountered. (See also, type_code 10007.)<br>For example, if you follow the chain from P1 to P2 when going forward, then, when going backward, P1 is expected to come after P2. The value of page id expected is P1, and the value of page id is P2. When the actual value of P3 is encountered, it is recorded as page id actual. |
| 10009 | object id expected | Specifies a 9-byte hexadecimal value that provides the expected object id (table), the partition id (partition of the table) if applicable, and the index id (index) of the page or allocation being checked.<br>For example, if a page is expected to belong to table T1 because it is reached from T1's chain, but is actually allocated to table T2, the object id for T1 is recorded, and the object id expected for T2 is recorded. The server truncates trailing zeros. |
| 10010 | data-only locked data page header | Indicates the 44-byte page header for the page where the fault is located. |
| 10011 | data-only locked b-tree leaf page header | Indicates the 44-byte page header for the page where the fault is located. |
| 10012 | data-only locked b-tree header | Indicates the 44-byte page header for the page where the fault is located. |
| 20001 | rerun checkstorage reco | Reruns checkstorage. |
| 20002 | indexalloc reco | Runs dbcc indexalloc with the fix option. |
| 20003 | tablealloc reco | Runs dbcc tablealloc with the fix option. |
| 20004 | checktable fix_spacebits reco | Runs dbcc tablealloc with the fix_spacebits option. |
| 20005 | checktable reco | Runs dbcc checktable. |
| 20006 | reorg reco | Runs the reorg command |
| 20007 | no action reco | This fault is harmless; no action is required. |
| 30000 | drop object reco | Drops the object and re-creates it. |
| 30001 | bulk copy reco | Bulk copies the data out and back in. |
| 40000 | check logs for hardware failure reco | Checks your operating system logs and corrects all reported hardware problems on disks containing a Sybase device. |
| 40001 | checkalloc reco | Runs dbcc checkalloc with the fix option. |

| *type_code* | *type_name* | **Description** |
|---|---|---|
| 40002 | reload db reco | Reloads the database from a clean backup. |
| 100000 | IO error | Indicates that part of the identified page could not be fetched from the device. This is usually caused by a failure of the operating system or the hardware. |
| 100001 | page id error | Indicates that the identifying ID (page number) recorded on the page is not valid. This might be the result of a page being written to or read from the wrong disk location, corruption of a page either before or as it is being written, or allocation of a page without subsequent initialization of that page. |
| 100002 | page free offset error | Indicates that the end of data on a page is not valid. This event affects insertions and updates on this page. It might affect some access to the data on this page. |
| 100003 | page object id error | Indicates that the page appears to be allocated to some other table than the one expected. If this is a persistent fault, it might be the consequence of either:<br>• An incorrect page allocation, which might only result in the effective loss of this page to subsequent allocation, or<br>• A corrupted page chain, which might prevent access to the data in the corrupted chain. |
| 100004 | timestamp error | Indicates that the page has a timestamp that is later than the database timestamp. This error can result in failure to recover when changes are made to this page. |
| 100005 | wrong dbid error | Indicates that the database ID dbid is stored on the database allocation pages. When this ID is incorrect, the allocation page is corrupt and all the indicated allocations are suspect. |
| 100006 | wrong object error | Indicates that the page allocation is inconsistent. The page appears to belong to one table or index, but it is recorded as being allocated to some other table or index in the allocation page. This error differs from page object id error in that the allocation is inconsistent, but the consequences are similar. |
| 100007 | extent id error | Indicates that an allocation was found for a table or index that is unknown to dbcc checkstorage. Typically, this results in the inability to use the allocated storage. |
| 100008 | fixed format error | Indicates that the page incorrectly indicates that it contains only rows of a single fixed length. dbcc checkstorage reports this error. dbcc checktable does not report it, but does repair it. |
| 100009 | row format error | Indicates that at least one row on the page is incorrectly formatted. This error might cause loss of access to some or all the data on this page. |
| 100010 | row offset error | Indicates that at least one row on the page is not located at the expected page offset. This error might cause loss of access to some or all of the data on this page. |

| *type_code* | *type_name* | Description |
|---|---|---|
| 100011 | text pointer error | Indicates that the location of the table row that points to the corrupted text or image data. This information might be useful for correcting the problem. |
| 100012 | wrong type error | Indicates that the page has the wrong format. For example, a data page was found in an index or a text/image column. |
| 100013 | non-OAM error | This error is a special case of wrong type error. It is not reported as a separate condition in the current release. |
| 100014 | reused page error | Indicates that a page is reached by more than one chain and that the chains belong to different objects. This error indicates illegal sharing of a page through corrupt page chain linkages. Access to data in either or both tables might be affected. |
| 100015 | page loop error | Indicates that a page is reached a second time while following the page chain for an object, which indicates a loop in the page chain. A loop can result in a session hanging indefinitely while accessing data in that object. |
| 100016 | OAM ring error | Indicates that a page is allocated but not reached by the page chains for the object.Typically, this results in the inability to use the allocated storage. |
| 100017 | OAM ring error | Indicates that the OAM page ring linkages are corrupted. This might not affect access to the data for this object, but it might affect insertions, deletions, and updates to that data. |
| 100018 | missing OAM error | Indicates that dbcc checkstorage found an allocation for the object that was not recorded in the OAM. This error indicates a corruption that might affect future allocations of storage, but probably does not affect access to the presently stored data. |
| 100019 | extra OAM error | Indicates that an allocation for this object was recorded in the OAM, but it was not verified in the allocation page. This error indicates a corruption that might affect future allocations of storage, but probably does not affect access to the presently stored data. |
| 100020 | check aborted error | Indicates that dbcc checkstorage stopped checking the table or index. To prevent multiple fault reports, the check operation on a single chain might be stopped without reporting this error. When an object contains several page chains, failure of the check operation for one chain does not prevent the continuation of the check operation on the other chains unless a fault threshold is exceeded. |
| 100021 | chain end error | Indicates that the end of the chain is corrupted. As a soft fault, it might indicate only that the chain was extended or truncated by more than a few pages during the dbcc checkstorage operation. |
| 100022 | chain start error | Indicates that the start of a chain is corrupted or is not at the expected location. If this is a persistent fault, access to data stored in the object is probably affected. |

| type_code | type_name | Description |
|---|---|---|
| 100023 | used count error | Indicates an inconsistency between the count of the pages used that is recorded in the OAM page and the count of the pages used that is determined by examining the allocation pages. |
| 100024 | unused count error | Indicates an inconsistency between the count of the pages reserved but unused that is recorded in the OAM page and the count of the pages reserved but unused that is determined by examining the allocation pages. |
| 100025 | row count error | Indicates an inconsistency between the row count recorded in the OAM page and the row count determined by dbcc checkstorage. |
| 100026 | serialloc error | Indicates a violation of the serial allocation rules applied to log allocations. |
| 100027 | text root error | Indicates a violation of the format of the root page of a text or image index. This check is similar to the root page checks performed by dbcc textalloc. |
| 100028 | page misplaced | Indicates that pages of this object were not found where they were expected to be from examination of the system tables. This usually indicates that sp_placeobject was used sometime in the past. In the dbcc_counters table, all misplaced pages are counted together, rather than being reported by device and partition. |
| 100029 | page header error | Indicates an internal inconsistency in the page's header other than the kind described by the other type codes. The severity of this error depends on the type of page and the inconsistency found. |
| 100030 | page format error | Indicates an internal inconsistency in the page's body other than the kind described by the other type codes. The severity of this error depends on the type of page and the inconsistency found. |
| 100031 | page not allocated | Indicates that dbcc checkstorage reached an unallocated page by following a page chain. This condition might affect access to data stored in this object. |
| 100032 | page linkage error | Indicates that dbcc checkstorage detected a fault with either the next or previous linkage of an interior page of a chain. If this is a persistent fault, access to data stored in the object is probably affected. |
| 100033 | non-contiguous free-space error | Indicates an invalid or inconsistent value for the noncontiguous free space on the page. |
| 100034 | insert free space error | Indicates an invalid or inconsistent value for the contiguous free space on the page. |
| 100035 | spacebits mismatch | Indicates an inconsistency in the page fullness indicator. |
| 100036 | deleted row count error | Indicates an invalid or inconsistent value for the deleted row count on the page. |
| 100037 | forwarded rows error | Indicates an inconsistency between the forwarded rows indicator and the number of forwarded rows on the page. |
| 100038 | page header type error | Indicates that a page header format indicator set incorrectly. |

| *type_code* | *type_name* | Description |
|---|---|---|
| 100039 | incorrect extent oampage | Extent OAM page reference is set incorrectly |
| 100040 | OAM page format error | Non-first OAM page has non-zero first OAM page-specific data. |

CHAPTER 3

# Monitoring Tables

This chapter describes the monitoring tables in alphabetical order.

The Attributes column provides information about how Adaptive Server manages the column. An Attribute value of:

- "Counter" indicates value in this column may wrap, or become zero and start incrementing again, because the value exceeds the maximum possible value of $2^{31}$. Adaptive Server resets the monitor counters when you run sp_sysmon without the noclear option. In Adaptive Server version 15.0.1 and later, the noclear option is, by default, included as a sp_sysmon parameter. In versions earlier than 15.0.1, you must specify noclear to prevent Adaptive Server from resetting the monitor counters.

  Resetting monitor counters may skew your results if you run sp_sysmon on the same Adaptive Server on which you are using the monitoring tables.
- "Null" indicates the column value may be null.
- "Reset" indicates the column is reset when you run sp_sysmon in a manner that causes it to clear the monitoring counters (see *Performance and Tuning Series: Monitoring Adaptive Server with sp_sysmon*).

# monCachedObject

Description

Stores statistics for all tables, partitions, and indexes with pages currently in a data cache.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monCacheObject are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| CacheID | int | | Unique identifier for the cache. |
| InstanceID | | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| DBID | int | | Unique identifier for the database. |
| IndexID | int | | Unique identifier for the index. |
| PartitionID | int | | Unique identifier for the partition. This is the same value as ObjectID for nonpartitioned objects. |
| CachedKB | int | | Number of kilobytes of the cache the object is occupying. |
| CacheName | varchar(30) | Null | Name of the cache. |
| ObjectID | int | Null | Unique identifier for the object. Null if the descriptor for the object has been removed from the server's metadata cache. In this situation, you can determine the object identifier by querying syspartitions in the specified database for the value of PartitionID. |
| DBName | varchar(30) | Null | Name of the database (NULL if the descriptor for the object was removed from the server's metadata cache). |
| OwnerUserID | int | Null | Unique identifier for the object owner. |
| OwnerName | varchar(30) | Null | Name of the object owner (null if the descriptor for the object was removed from the server's metadata cache). |
| ObjectName | varchar(30) | Null | Name of the object (null if the descriptor for the object was removed from the server's metadata cache). |
| PartitionName | varchar(30) | Null | Name of the object partition (null if the descriptor for the object was removed from the server's metadata cache). |
| ObjectType | varchar(30) | Null | Object type (null if the object is no longer open). |
| TotalSizeKB | int | Counter, null | Partition size, in kilobytes. |
| ProcessesAccessing | int | Counter, null | Number of processes currently accessing pages for this object in the data cache. |

# monCachePool

Description — Stores statistics for all pools allocated for all data caches.

Enable the enable monitoring configuration parameter for this monitoring table to collect data.

Columns — The columns for monCachePool are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| CacheID | int | | Unique identifier for the cache |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| IOBufferSize | int | | Size (in bytes) of the I/O buffer for the pool |
| AllocatedKB | int | | Number of bytes allocated for the pool |
| PhysicalReads | int | Counter | Number of buffers read from disk into the pool |
| Stalls | int | Counter | Number of times I/O operations were delayed because no clean buffers were available in the wash area for this data cache |
| PagesTouched | int | Counter | Number of pages that are currently being used within the pool |
| PagesRead | int | Counter | Number of pages read into the pool |
| BuffersToMRU | int | Counter | Number of buffers fetched and replaced in the most recently used portion of the pool |
| BuffersToLRU | int | Counter | Number of buffers fetched and replaced in the least recently used portion of the pool: fetch and discard |
| CacheName | varchar(30) | Null | Name of the cache |
| LogicalReads | int | Counter | Number of buffers read from the pool |
| PhysicalWrites | int | Counter | Number of write operations performed for data in this pool (one write operation may include multiple pages) |
| APFReads | int | Counter | Number of asynchronous prefetch (APF) read operations that loaded pages into this pool |
| APFPercentage | int | | The configured asynchronous prefetch limit for this pool |
| WashSize | int | | The wash size, in kilobytes, for a memory pool |

# monCachedProcedures

Description

Stores statistics for all stored procedures, triggers, and compiled plans currently stored in the procedure cache.

Enable the enable monitoring and statement statistics active configuration parameters for this monitoring table to collect data.

Columns

The columns for monCacheProcedures are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| ObjectID | int | | Unique identifier for the procedure |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| OwnerUID | int | | Unique identifier for the object's owner |
| DBID | int | | Unique identifier for the database in which the object exists |
| PlanID | int | | Unique identifier for the query plan for the object in the procedure cache |
| MemUsageKB | int | | Number of kilobytes of memory used by the procedure |
| CompileDate | datetime | | Date that the procedure was compiled |
| ObjectName | varchar(30) | Null | Name of the procedure |
| ObjectType | varchar(32) | Null | Type of procedure (for example, stored procedure or trigger) |
| OwnerName | varchar(30) | Null | Name of the object owner |
| DBName | varchar(30) | Null | Name of the database |
| RequestCnt | int4 | | Number of times this procedure was requested from cache |
| TempdbRemapCnt | int4 | | Number of times this procedure was remapped for the temporary database's ID. |
| AvgTempdbRemapTime | int4 | | Average time (in milliseconds) spent remapping the temporary databases's ID. |
| ExecutionCount | int | Counter | Number of times Adaptive Server executed the stored procedure plan or tree since it was cached |
| CPUTime | int | Counter | Total number of milliseconds of CPU time used |
| ExecutionTime | int | Counter | Total amount of elapsed time, in milliseconds, Adaptive Server spent executing the stored procedure plan or tree |
| PhysicalReads | int | Counter | Number of physical reads performed |
| LogicalReads | int | Counter | Number of pages read |
| PhysicalWrites | int | Counter | Number of physical writes performed |
| PagesWritten | int | Counter | Number of pages written |

# monCachedStatement

Description

Stores detailed monitoring information about the statement cache, including information about resources used during the previous executions of a statement, how frequently a statement is executed, the settings in effect for a particular plan, the number of concurrent uses of a statement, and so on. This information can be helpful when troubleshooting, and when deciding which statements to retain in the cache.

**Note** Machines that use multiple CPUs with different clock frequencies may report inaccurate elapsed time.

The columns in monCachedStatement allow two attributes: "counter" if the column has a counter value, and "reset" if you can reset the column using sp_sysmon.

Enable the enable monitoring, statement cache size, and enable stmt cache monitoring configuration parameters for this monitoring table to collect data.

Columns

The columns for monCacheStatement are:

| Names | Datatypes | Description |
|---|---|---|
| InstanceID | tinyint | (Cluster environments only) ID of an instance in ashared-disk cluster. |
| SSQLID | int | Unique identifier for each cached statement. This value is treated as a primary key for monCachedStatement, and is used in functions.<br>show_cached_text uses SSQLID to refer to individual statements in the cache. |
| Hashkey | int | Hash value of the SQL text of the cached statement. A hash key is generated based on a statement's text, and can be used as an approximate key for searching other monitoring tables. |
| StmtType | tinyint | |
| UserID | int | User ID of the user who initiated the statement that has been cached. |
| SUserID | int | Server ID of the user who initiated the cached statement. |
| DBID | smallint | Database ID of the database from which the statement was cached. |
| UseCount | int | Number of times the statement was accessed after it was cached. |
| StatementSize | int | Size of the cached statement, in bytes. |
| MinPlanSizeKB | int | Size of the plan when it is not in use, in kilobytes. |
| MaxPlanSizeKB | int | Size of the plan when it is in use, in kilobytes. |

| Names | Datatypes | Description |
|---|---|---|
| CurrentUsageCount | int | Number of concurrent users of the cached statement. Attribute is counter. |
| MaxUsageCount | int | Maximum number of times the cached statement's text was simultaneously accessed. Attribute is counter. |
| NumRecompilesSchemaChanges | int | Number of times the statement was recompiled due to schema changes. Running update statistics on a table may result in changes to the best plan. This change is treated as a minor schema change.<br>Recompiling a statement many times indicates that it is not effective to cache this particular statement, and that you may want to delete the statement from the statement cache to make space for some other, more stable, statement. Attribute is counter. |
| NumRecompilesPlanFlushes | int | Number of times the cached statement was recompiled because a plan was not found in the cache. Attribute is counter. |
| HasAutoParams | tinyint | "true" if the statement has any parameterized literals, "false" if it does not. |
| ParallelDegree | tinyint | Degree of parallelism used by the query that is stored for this statement |
| QuotedIdentifier | tinyint | Specifies whether the plan compiled with set quoted_identifier is enabled. |
| TransactionIsolationLevel | tinyint | Transaction isolation level for which the statement was compiled. |
| TransactionMode | tinyint | Specifies whether "chained transaction mode" is enabled for the statement. |
| SAAuthorization | tinyint | Specifies whether the plan was compiled with sa_role authorization. |
| SystemCatalogUpdate | tinyint | Specifies whether allow catalog updates was enabled when the plan was compiled. |
| MetricsCount | int | Number of times metrics were aggregated for this statement. |
| MinPIO | int | Maximum physical I/Os that occurred during any execution of this statement. |
| MaxPIO | int | Maximum physical I/Os that occurred during any execution of this statement. |
| AvgPIO | int | Average number of physical I/Os that occurred during execution of this statement. |
| MinLIO | int | Minimum logical I/Os that occurred during any execution of this statement. |
| MaxLIO | int | Maximum logical I/Os that occurred during any one execution of this statement. |

| Names | Datatypes | Description |
|---|---|---|
| AvgLIO | int | Average number of logical I/Os that occurred during execution of this statement. |
| MinCpuTime | int | The minimum amount of CPU time, in milliseconds, consumed by any execution of this statement. |
| MaxCpuTime | int | The maximum amount of CPU time, in milliseconds, consumed by any execution of this statement. |
| AvgCpuTime | int | The average amount of CPU time, in milliseconds, consumed by this statement. |
| MinElapsedTime | int | Minimum elapsed execution time for this statement. |
| MaxElapsedTime | int | Maximum elapsed execution time for this statement. |
| AvgElapsedTime | int | Average elapsed execution time for this statement. |
| AvgScanRows | int | Average number of scanned rows read per execution |
| MaxScanRows | int | Maximum number of scanned rows read per execution |
| AvgQualifyingReadRows | int | Average number of qualifying data rows per read command execution |
| MaxQualifyingReadRows | int | Maximum number of qualifying data rows per query execution |
| AvgQualifyingWriteRows | int | Average number of qualifying data rows per query execution |
| MaxQualifyingWriteRows | int | Maximum number of qualifying data rows per query execution |
| LockWaits | int | Total number of lock waits |
| LockWaitTime | int | Total amount of time, in milliseconds, spent waiting for locks |
| SortCount | int | Total number of sort operations |
| SortSpilledCount | int | Total number of sort operations spilled to disk |
| TotalSortTime | int | Total amount of time, in milliseconds, spent in sorts |
| MaxSortTime | int | Maximum amount of time, in milliseconds, spent in a sort |
| DBName | varchar(30) | Name of database from which the statement was cached. Attribute is null. |
| CachedDate | datetime | Timestamp of the date and time when the statement was first cached. |
| LastUsedDate | datetime | Tmestamp of the date and time when the cached statement was last used. Use this information with CachedDate to determine how frequently this statement is used, and whether it is helpful to have it cached. |
| LastRecompiledDate | datetime | Date when the statement was last recompiled, because of schema changes or because the statement was not found in the statement cache. |
| OptimizationGoal | varchar(30) | The optimization goal used to optimize this statement. |
| OptimizerLevel | varchar(30) | The optimizer level used to optimize this statement. |

| Names | Datatypes | Description |
|---|---|---|
| AdjustToParallel | int | Indicates if an insufficient number of worker threads were available to execute the query with the full degree of parallelism the query plan calls for, but the query did execute with some parallelism. |
| AdjustToSerial | int | Indicates if an insufficient number of worker threads were available to execute the query in parallel so the the query was executed serially. |
| ThreadDeficit | int | Indicates that the cumulative total number of worker threads were unavailable to execute this query since it was added to the statement cache. |

# monCIPC

Description

(Cluster environments only) Provides summary figures for total messaging within the cluster, as viewed from the current instance or all instances.

One row is returned in the monCIPC table for each instance in the cluster, if the system view is set to cluster; otherwise, a single row is returned for the instance on which the query is executed.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monCIPC are:

| Name | Datatype | Description |
|---|---|---|
| InstanceID | tinyint | ID of the instance within the cluster |
| ReceiveCount | int | Number of messages received by this instance |
| TransmitCount | int | Number of messages sent by this instance |
| Multicast | int | Number of messages sent that were addressed to all other instances in the cluster |
| Synchronous | int | Number of those messages sent synchronously |
| ReceiveSoftError | int | Number of recoverable errors received on this instance |
| ReceiveHardError | int | Number of unrecoverable errors received on this instance |
| TransmitsSoftError | int | Number of recoverable transmit errors on this instance |
| TransmitHardError | int | Number of unrecoverable transmit errors on this instance |
| Retransmits | int | Number of retransmissions performed by this instance |
| Switches | int | Number of switches between the primary interconnect network and the secondary interconnect network |
| FailedSwitches | int | Number of attempts to switch between primary and secondary interconnect networks that failed |
| RegularBuffersInUse | int4 | Number of buffers from the CIPC regular buffer pool currently allocated. |
| FreeRegularBuffers | int4 | Number of buffers available in the CIPC regular buffer pool. |
| MaxRegularBuffersInUse | int4 | Maximum number of buffers from the CIPC regular buffer pool allocated at any time since the server was started. |
| LargeBuffersInUse | int4 | Number of buffers from the CIPC large buffer pool currently allocated. |
| FreeLargeBuffers | int4 | Number of buffers available in the CIPC large buffer pool. |
| MaxLargeBuffersInUse | int4 | Maximum number of buffers from the CIPC large buffer pool allocated at any time since the server was started. |

# monCIPCEndpoints

Description

(Cluster environments only) Provides a detailed summary, giving traffic data for each subsystem within the cluster instance.

One row is returned for each logical endpoint in the instance. If the system view is set to cluster, a set of rows is returned for each node in the cluster.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monCIPCEndpoints are:

| Name | Datatype | Description |
|---|---|---|
| InstanceID | tinyint | ID of the instance within the cluster |
| ReceiveCount | int | Number of messages received by this logical endpoint within the cluster |
| TransmitCount | int | Number of messages sent by this logical endpoint within the instance |
| ReceiveBytes | int | Number of bytes received by this logical endpoint within the instance |
| TransmitBytes | int | Number of bytes sent by this logical endpoint within the instance |
| ReceiveQ | int | Current number of messages queued for this logical endpoint |
| MaxReceiveQ | int | Maximum number of messages ever observed queued for this logical endpoint |
| DoneQ | int | Current number of messages for this logical endpoint that were processed and await further action |
| MaxDoneQ | int | Maximum number of messages ever observed for this logical endpoint, which have been processed and await further action |
| MaxRecvQTime | real4 | Maximum time (in milliseconds) a message spends in the queues of the current logical end point. |
| AvgRecvQTime | real4 | Average time (in milliseconds) a message spends in the queues of the current logical end point. |
| EndPoint | varchar | Name of CIPC endpoint |

# monCIPCLinks

Description

(Cluster environments only) Monitors the state of the links between instances in the cluster.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monCIPCLinks are:

| Name | Datatype | Description |
|---|---|---|
| InstanceID | tinyint | ID of the instance within the cluster. |
| LocalInterface | varchar30 | Name of the link's local network endpoint. Same name that appears in the *hosts* file for a server name. |
| RemoteInterface | varchar30 | Name of the link's remote end point. Same name that appears in the *hosts* file for a server name. |
| PassiveState | varchar10 | Latest state listed in the traffic on the link. |
| PassiveStateAge | int | Time since the PassiveState column was updated, in milliseconds. |
| ActiveState | varchar10 | Latest state used, as determined by active monitoring (when no traffic was present on the link). |
| ActiveStateAge | int | Time since the ActiveState column was updated, in milliseconds. |

# monCIPCMesh

Description

(Cluster environments only) Gives summary figures for the mesh of connections, from the current instance to all other instances in the cluster, on a per-instance basis.

One row is returned for each of the four connections to each of the other nodes in the cluster, up to the maximum configured. If the system view is cluster, a set of rows for each instance active in the cluster is returned.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monCIPCMesh are:

| Name | Datatype | Description |
|---|---|---|
| InstanceID | tinyint | ID of the instance within the cluster. |
| FarInstanceID | tinyint | Instance number of the far-end instance in the cluster. |
| Received | int | Number of messages received by this instance from the FarInstanceID instance. |
| Dropped | int | Number of messages from the FarInstanceID instance that were dropped, due to a lack of resources. |
| Transmitted | int | Number of messages transmitted to the FarInstanceID instance. |
| Resent | int | Number of messages re-sent to the FarInstanceID instance. |
| Retry | int | Number of packets retried to the FarInstanceID instance. |
| ControlRx | int | Number of control messages received by the InstanceID instance. |
| ControlTx | int | Number of control messages sent by the InstanceID instance for this mesh. |
| SendQ | int | Current number of messages waiting to be sent to the FarInstanceID instance for this mesh. |
| MaxSendQ | int | Maximum number of packets in the send queue for this mesh since the InstanceID instance was started. |
| SentQ | int | Number of packets sent by the InstanceID instance to the FarInstanceID instance that have not yet been acknowledged by the FarInstanceID instance. |
| MaxSentQ | int | Maximum number of messages sent, but notification of sending is not yet processed. |
| MaxSendQTime | real | Maximum time that has been required to process a message in the send queue for this mesh. In milliseconds. |
| AvgSendQTime | real | Average amount of time required to process a message in the send queue for this mesh. In milliseconds. |

| Name | Datatype | Description |
|---|---|---|
| Mesh | varchar | The channel name for the connection. One of:<br>• Out of Band<br>• Message<br>• Large Message<br>• Direct memory access (DMA) |
| MinRTT | int | Minimum round-trip delay observed for messages (applies only to user datagram protocol (UDP) transport). |
| MaxRTT | int | Maximum round trip delay observed for messages (applies only to UDP transport). |
| AverageRTT | int | Average round trip delay observed for messages (applies only to UDP transport). |

# monCLMObjectActivity

Description

(Cluster environments only) Collects cluster lock information. monCLMObjectActivity:

- Tracks activity for objects only in the master and user databases.
- Tracks physical lock activity at the partition level.

Cluster object locks for a database have an Object-PartitionID of 0.

Enable the enable monitoring configuration parameter for this monitoring table to collect data.

Columns

monCLMObjectActivity contains these columns:

| Column name | Type | Description |
|---|---|---|
| InstanceID | int1 | Instance ID. |
| DBID | int4 | Database ID. |
| Object_PartitionID | int4 | Identity of the object making the lock request. |
| LockRequests | int4 | Number of cluster lock requests. |
| LocalMaster | int4 | Number of times a lock request finds the current instance to be the lock master.<br>One instance in the cluster becomes the "lock master." When an instance needs a cluster lock, it contacts the lock master for the lock. |
| Waited | int4 | Number of lock requests granted with contention at the remote instance. |
| Granted | int4 | Number of lock requests granted without contention at the remote instance. |
| RWConflictWaited | int4 | Number of lock requests that waited because of a read-write conflict lock that was granted to a task at a remote instance. |
| AvgRWConflictWaitTime | flt4 | Average amount of time spent performing the wait described by RWConflictWaited. |
| MaxRWConflictWaitTime | flt4 | Maximum amount of time spent performing the wait described by RWConflictWaited. |
| WWConflictWaited | int4 | Number of lock requests that waited because of a write-write conflict lock that was granted to a task at a remote instance. |
| AvgWWConflictWaitTime | flt4 | Average amount of time spent performing the wait described by WWConflictWaited. |
| MaxWWConflictWaitTime | flt4 | Maximum amount of time spent performing the wait described in WWConflictWaited. |
| ClusterMsgWaits | int4 | Number of waits due to cluster messaging. |
| AvgClusterMsgWaitTime | flt4 | Average wait time due to cluster messaging. |

| Column name | Type | Description |
|---|---|---|
| MaxClusterMsgWaitTime | flt4 | Maximum wait time due to cluster messaging. |
| DowngradeReqRecv | int4 | Number of downgrade service requests received at the cluster lock owner. |
| DowngradeReqRecvWithNoBlocker | int4 | Number of the downgrade service requests received without any blocking task ownership at cluster lock owner. |
| ClusterDeadlock | int4 | Number of deadlocks caused by multiple instances attempting to acquire the same cluster lock simultaneously. |
| Locktype | varchar(20) | Type of lock. |

# monClusterCacheManager

Description

(Cluster environments only) Stores diagnostic information about the cluster cache manager daemon running on each instance. monClusterCacheManager reports cluster-wide information on a per-instance basis.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monClusterCacheManager are:

| Name | Datatype | Description |
|---|---|---|
| InstanceID | tinyint | ID of the instance within the cluster |
| RequestsQueued | int | Number of requests queued to the cluster cache manager daemon |
| RequestsRequeued | int | Number of requests requeued to the cluster cache manager daemon |
| RequestsServiced | int | Number of requests serviced by the cluster cache manager daemon |
| DiskWrites | int | Number of disk writes initiated by the cluster cache manager daemon |
| SleepCount | int | Number of times the cluster cache manager daemon went to sleep |
| DaemonName | varchar | Name of the cluster cache manager daemon |
| TransfersInitiated | int | Number of transfers initiated by the cluster cache manager daemon |
| Downgrades | int | Number of downgrades performed by the cluster cache manager daemon |
| Releases | int | Number of releases performed by the cluster cache manager daemon |
| AvgServiceTime | int | Average time (in milliseconds) spent servicing a request |
| MaxQSize | int | Maximum number of requests queued to the cluster cache manager daemon at any time since the instance started |

# monCMSFailover

Description

(Cluster environments only) Tracks the time at which the cluster membership service (CMS) detects the failure, gets a new cluster view, resynchronizes the heartbeat, posts the failure event, and completes the failure event. There is a row for each instance.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

monCMSFailover contains these columns:

| Column name | Type | Description |
|---|---|---|
| InstanceID | tinyint1 | Instance performing the failover. |
| FailedInstanceID | varchar(96) | List of failed instance IDs, separated by commas. |
| FailDetectTime | datatime(8) | Time when cluster failure is detected. |
| InitViewTime | datatime(8) | Time when initial cluster view is obtained. |
| FinalViewTime | datatime(8) | Time when final cluster view is obtained. |
| ResynchHBTime | datatime(8) | Time when cluster-wide heartbeat is resynchronized. |
| NotifyFailTime | datatime(8) | Time when failure event is posted. |
| EventdoneTime | datatime(8) | Time when last failure event is finished. |

# monDataCache

Description — Stores statistics relating to Adaptive Server data caches.

Enable the enable monitoring configuration parameter for this monitoring table to collect data.

Columns — The columns for monDataCache are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| CacheID | int | | Unique identifier for the cache |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| RelaxedReplacement | int | | Specifies whether the cache is using relaxed cache replacement strategy |
| BufferPools | int | | Number of buffer pools within the cache |
| CacheSearches | int | Counter, reset | Cache searches directed to the cache |
| PhysicalReads | int | Counter, reset | Number of buffers read into the cache from disk |
| LogicalReads | int | Counter, reset | Number of buffers retrieved from the cache |
| PhysicalWrites | int | Counter, reset | Number of buffers written from the cache to disk |
| Stalls | int | Counter, reset | Number of times I/O operations were delayed because no clean buffers were available in the wash area |
| CachePartitions | smallint | | Number of partitions currently configured for the cache |
| CacheName | varchar(30) | Null | Name of cache |
| Status | varchar(30) | null | Status of cache. One of:<br>• Active<br>• Pending/Active<br>• Pending/Delete<br>• Update Cache<br>• Cache Create<br>• Cache Delete<br>• (Cluster Edition only) Cache Skip |
| Type | varchar(30) | null | Type of cache. One of:<br>• Default<br>• Mixed<br>• Mixed, HK Ignore<br>• Log Only<br>• In-Memory Storage |
| CacheSize | int | | Total size of cache, in kilobytes |
| ReplacementStrategy | varchar(30) | null | Cache replacement strategy |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| APFReads | int | Counter | Number of asynchronous prefetch (APF) reads for this data cache |
| Overhead | int | | Cache overhead |

# monDBRecovery

Description

(Cluster environments only) Contains rows from all instances in the cluster and contains rows for every database that contributes to recovery.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monDBRecovery are:

| Column name | Type | Description |
|---|---|---|
| DBID | int4 | Unique identifier for the database |
| InstanceID | int1 | Instance that performed the recovery (applicable only to the Cluster Edition) |
| MaxOpenXacts | int4 | Maximum number of open transactions seen durning recovery |
| MaxPFTSEntries | int4 | Maximum number of PFTS entries seen during recovery |
| Buckets | int4 | Number of buckets |
| LogBTotPages | int4 | Number of `log scan getpage` requests during the log boundary determination pass. |
| LogBTotAPFWaited | int4 | Number of `log scan getpage` requests that found the I/O in progress during the log boundary determination pass |
| LogBTotIO | int4 | Number of `log scan getpage` requests with physical I/O during the log boundary determination pass |
| AnlTotRec | int4 | Total number of log records to be scanned by the recovery process |
| AnlPhase1Recs | int4 | Number of log records in phase 1 recovery process |
| AnlPhase1RedoRecs | int4 | Number of log records to redo in phase 1 recovery |
| AnlPhase2Recs | int4 | Number of log records in phase 2 recovery process |
| AnlPhase2RedoRecs | int4 | Number of log records to redo in phase 2 recovery |
| AnlTotPages | int4 | Number of `log scan getpage` requests during the analysis process |
| AnlTotAPFWaited | int4 | Number of `log scan getpage` requests that found the I/O in progress during the analysis pass |
| AnlTotIO | int4 | Number of `log scan getpage` requests with physical I/O during the analysis pass |
| RedoOps | int4 | Total operations considered for redo |
| RedoOpsNotRedonePFTS | int4 | Operations that did not need redo (PFTS check) |
| RedoOpsRedonePFTS | int4 | Operations that might need redo (PFTS check) |
| RedoOpsRedoneTS | int4 | Operations that needed redo (timestamp check) |
| RedoOpsNotRedoneTS | int4 | Operations that did not need redo (timestamp check) |
| RedoLogTotPages | int4 | Number of `log scan getpage` requests during the redo pass |
| RedoLogTotAPFWaited | int4 | Number of `log scan getpage` requests that found the I/O in progress during the redo pass |

| Column name | Type | Description |
|---|---|---|
| RedoLogTotIO | int4 | Number of `log scan getpage` requests with physical I/O during the redo pass |
| RedoRecTotPage | int4 | Number of `recovery pages getpage` requests during the redo pass |
| RedoRecTotAPFWaited | int4 | Number of `recovery pages getpage` requests that found the I/O in progress during the redo pass |
| RedoRecTotIO | int4 | Number of `recovery pages getpage` requests with physical I/O in progress during the redo pass |
| UndoRecsUndone | int4 | Number of log records udone |
| UndoLogTotPages | int4 | Number of `log scan getpage` requests during the undo pass |
| UndoLogTotAPFWaited | int4 | Number of `log scan getpage` requests that found the I/O in progress during the undo pass |
| UndoLogTotIO | int4 | Number of `log scan getpage` requests with physical I/O during the undo pass |
| UndoRecTotPages | int4 | Number of `recovery pages getpage` requests during the undo pass |
| UndoRecTotAPFWaited | int4 | Number of `recovery pages getpage` requests that found the I/O in progress during the undo pass |
| UndoRedTotIO | int4 | Number of `recovery pages getpage` requests with physical I/O during the undo pass |
| DBName | varchar(30) | Name of the database |
| FailedInstanceID | int1 | ID of the failed instance (applicable only to the Cluster Edition) |
| Command | varchar(30) | One of load database, load transaction, online database, mount database, and start or failover commands executed by the process that is running recovery |
| RecType | varchar(30) | Type of recovery – one of server start, load database, load transaction, or node failover |
| LobBStartTime | datetime(8) | Start time for the log boundaries determination pass |
| LogBEndTime | datetime(8) | End time for the log boundaries determination pass |
| AnlStartTime | datetime(8) | Start time of analysis pass |
| AnlEndTime | datetime(8) | End time of the analysis pass |
| RedoStartTime | datetime(8) | Start time of the redo pass |
| RedoEndTime | datetime(8) | End time of the redo pass |
| UndoStartTime | datetime(8) | Start time of the undo pass |
| UndoEndTime | datetime(8) | End time of the undo pass |

# monDBRecoveryLRTypes

Description

(Cluster environments only) Tracks log records seen during recovery. Contains a row for each log record type for which at least one log record was seen by recovery.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

monDBRecoveryLRTypes contains these columns:

| Column name | Type | Description |
|---|---|---|
| DBID | int4 | Unique identifier for the database |
| InstanceID | int1 | (Cluster environments only) Instance that performed the recovery |
| NumRecs | int4 | Number of records seen during recovery, by type |
| LogRecType | varchar(30) | Log record type |

monDBRecoveryLRTypes requires no parameters.

# monDeadLock

Description

Provides information about deadlocks. Use deadlock pipe max messages to tune the maximum number of messages returned.

monDeadLock is an historical monitoring table. See *Performance and Tuning: Monitoring Tables*.

Use sp_monitor 'deadlock' to check current deadlock options. The deadlock parameter provides a number of reports based on monDeadLock, which are useful for analyzing the history of server deadlocks.

Enable the enable monitoring, deadlock pipe max messages, and deadlock pipe active configuration parameters for this monitoring table to collect data.

Columns

The columns for monDeadLock are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| DeadLockID | int | | Unique identifier for the deadlock |
| VictimKPID | int | | Kernel process ID (kpid) of the victim process for the deadlock |
| InstanceID | int | | ID of an instance in a shared-disk cluster. |
| ResolveTime | datetime | | Time when the deadlock was resolved |
| ObjectDBID | int | | Unique database identifier for database where the object resides |
| PageNumber | int | | Page number requested for the lock, if applicable |
| RowNumber | int | | Row number requested for the lock, if applicable |
| HeldFamilyId | smallint | | spid of the parent process holding the lock |
| HeldSPID | smallint | | spid of process holding the lock |
| HeldKPID | int | | kpid of process holding the lock |
| HeldProcDBID | int | | Unique identifier for the database where the stored procedure that caused the lock to be held resides, if applicable |
| HeldProcedureID | int | | Unique object identifier for the stored procedure that caused the lock to be held, if applicable |
| HeldBatchID | int | | Identifier of the SQL batch executed by the process holding the lock when the deadlock occurred |
| HeldContextID | int | | Unique context identifier for the process holding the lock when it was blocked by another process (not when it acquired the lock) |
| HeldLineNumber | int | | Line number within the batch of the statement being executed by the process holding the lock when it was blocked by another process (not when it acquired the lock) |
| WaitFamilyId | smallint | | spid of the parent process waiting for the lock |
| WaitSPID | smallint | | spid of the process waiting for the lock |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| WaitKPID | int | | kpid of the process waiting for the lock |
| WaitTime | int | | Amount of time, in milliseconds, that the waiting process was blocked before the deadlock was resolved |
| ObjectName | varchar(30) | Null | Name of the object |
| HeldUserName | varchar(30) | Null | Name of the user for whom the lock is being held |
| HeldApplName | varchar(30) | Null | Name of the application holding the lock |
| HeldTranName | varchar(255) | Null | Name of the transaction in which the lock was acquired |
| HeldLockType | varchar(20) | Null | Type of lock being held |
| HeldCommand | varchar(30) | | Category of process or command that the process was executing when it was blocked |
| WaitUserName | varchar(30) | Null | Name of the user for whom the lock is being requested |
| WaitLockType | varchar(20) | Null | Type of lock requested |
| HeldSourceCodeID | varchar(30) | | For internal use only. |
| WaitSourceCodeID | varchar(30) | | For internal use only. |
| HeldClientApplName | varchar(30) | Null | Value for the *clientapplname* property set by the application holding the lock |
| HeldClientName | varchar(30) | Null | Value of the *clientname* property set by the application holding the lock |
| HeldClientHostName | varchar(30) | Null | Value for the *clienthostname* property set by the application holding the lock |
| HeldHostName | varchar(30) | Null | Name of the host machine on which the application that executed the query holding the lock is running |
| HeldNumLocks | int | | Number of locks currently held by holding spid |
| HeldProcDBName | varchar(30) | Null | Name of the database in which the stored procedure was executing the blocking process at the time the deadlock occurred, if applicable |
| HeldProcedureName | varchar(30) | Null | Name of the stored procedure the blocking process was executing at the time the deadlock occurred, if applicable |
| HeldStmtNumber | int | | Statement number in the SQL batch of the SQL statement holding the lock |
| ObjectDBName | varchar(30) | Null | Name of the database |
| ObjectID | int | Null | Unique identifier for the object |
| WaitApplName | varchar(30) | Null | Name of the application waiting for the lock |
| WaitBatchID | int | | Identifier of the SQL batch executed by the process waiting for the lock when the lock timeout occurred |
| WaitClientApplName | varchar(30) | Null | Value of the *clientapplname* property set by the application waiting for the lock |
| WaitClientHostName | varchar(30) | Null | Value of the *clienthostname* property set by the application waiting for the lock |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| WaitClientName | varchar(30) | Null | Value of the *clientname* property set by the application waiting for the lock |
| WaitCommand | varchar(30) | Null | Category of process or command that the process was executing when it was blocked and then timed out |
| WaitContextID | int | | Unique context identifier for the process waiting for the lock when it was blocked by another process |
| WaitHostName | varchar(30) | Null | Name of the host running the process waiting for the lock. |
| WaitLineNumber | int | | Line number of the SQL statement in the SQL batch or stored procedure waiting for the lock |
| WaitProcDBID | int | | Unique identifier for the database in which the stored procedure waiting for the lock resides, if applicable |
| WaitProcDBName | varchar(30) | Null | Name for the database where the stored procedure that is waiting for the lock resides, if applicable |
| WaitProcDBName | varchar(30) | Null | Name for the database where the stored procedure that is waiting for the lock resides, if applicable |
| WaitProcedureID | int | | ID of the stored procedure waiting for the lock, if applicable |
| WaitProcedureName | varchar(30) | Null | Name for the stored procedure waiting for the lock, if applicable |
| WaitStmtNumber | int | | Line number in SQL batch waiting for the lock |
| WaitTranName | varchar(255) | Null | Name of the transaction in which the lock was requested |

# monDeviceIO

Description

Returns statistical information relating to activity on database devices.

Enable the enable monitoring configuration parameter for this monitoring table to collect data.

Columns

The columns for monDeviceIO are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| Reads | int | Counter, reset | Number of reads from the device |
| APFReads | int | Counter, reset | Number of asynchronous prefetch (APF) reads from the device |
| Writes | int | Counter, reset | Number of writes to the device |
| DevSemaphoreRequests | int | Counter, reset | Number of I/O requests to a mirrored device (if mirrored) |
| DevSemaphoreWaits | int | Counter, reset | Number of tasks forced to wait for synchronization of an I/O request to a mirrored device (if mirrored) |
| IOTime | int | Counter | Total amount of time (in milliseconds) spent waiting for I/O requests to be satisfied |
| LogicalName | varchar(30) | Null | Logical name of the device |
| PhysicalName | varchar(128) | Null | Full hierarchic file name of the device |

# monDeviceSpaceUsage

Description

Provides information about the file systems on which database devices are allocated. Space information is available only for file system devices. File system size and free space values are NULL for database devices allocated on raw devices.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns in monDeviceSpaceUsage are:

| Name | Datatype | Description |
|---|---|---|
| InstanceID | tinyint | (Cluster environments only) ID of the instance |
| VDevNo | int | Virtual number of the device |
| LogicalName | varchar(30) | Logical name of the device |
| PhysicalName | varchar(128) | Physical name of the device |
| DeviceSizeMB | int | Size of the device, in megabytes |
| FileSystemName | varchar(128) | Name of the file system |
| FileSystemSizeMB | int | Size of the file system, in megabytes |
| FileSystemFreeMB | int | Amount of available free space, in megabytes, on the file system |

# monEngine

Description — Provides statistics regarding Adaptive Server engines.

Enable the enable monitoring configuration parameter for this monitoring table to collect data.

Columns — The columns for monEngine are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| EngineNumber | smallint | | Number of the engine. |
| ThreadID | int | | ID of the thread associated with the engine. |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| CurrentKPID | smallint | | Kernel process identifier (kpid) for the currently executing process. |
| PreviousKPID | int | | kpid for the previously executing process. |
| CPUTime | int | Counter, reset | Total time, in seconds, the engine has been running. |
| SystemCPUTime | int | Counter, reset | Time, in seconds, the engine has been executing system database services. |
| UserCPUTime | int | Counter, reset | Time, in seconds, the engine has been executing user commands. |
| IOCPUTime | int4 | | The amount of time, in seconds, the engine has been waiting for issued IOs to complete. |
| IdleCPUTime | int | Counter, reset | Time, in seconds, the engine has been in idle spin mode. |
| Yields | int | Counter, reset | Number of times this engine yielded to the operating system. If you are running Adaptive Server in process mode, modify the rate of yielding during idle periods using runnable process search count. If you are running Adaptive Server in threaded mode, modify the rate of yielding during idle periods with alter thread pool .. idle timeout. |
| Connections | int | Counter | Number of connections this engine handles. |
| DiskIOChecks | int | Counter, reset | (Process mode only) Number of times the engine checked for asynchronous disk I/O. Modify the frequency of these checks with i/o polling process count. Does not apply when a thread is used for I/O completion polling. |
| DiskIOPolled | int | Counter, reset | (Process mode only) Number of times the engine polled for completion of outstanding asynchronous disk I/O. The polling occurs whenever disk I/O checks indicate that asynchronous I/O has been posted and is not yet complete. |
| DiskIOCompleted | int | Counter, reset | (Process mode only) Number of asynchronous disk I/Os completed when the engine polled for outstanding asynchronous disk I/O. |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| MaxOutstandingIOs | int | | Current number of I/O requests initiated by this engine that are not completed. |
| ProcessesAffinitied | int | | Number of processes associated with this engine. |
| ContextSwitches | int | Counter, reset | Number of context switches. |
| HkgcMaxQSize | int | | Maximum number of items Adaptive Server can queue for housekeeper garbage collection in this engine. |
| HkgcPendingItems | int | | Number of items yet to be collected by housekeeper garbage collector on this engine. |
| HkgcHWMItems | int | | Maximum number of pending items queued for housekeeper garbage collector at any instant since server started. |
| HkgcOverflows | int | | Number of items that could not be queued to housekeeper garbage collector due to queue overflows. |
| Status | varchar(20) | Null | Status of the engine (online, offline, and so on). |
| Starttime | datetime | Null | Date that the engine came online. |
| StopTime | datetime | | Date that the engine went offline. |
| AffinitiedToCPU | int | Null | Number of the CPU to which the engine is affinitied. |
| OSPID | int | Null | Identifier for the operating system process executing the engine. |

# monErrorLog

Description

Returns the most recent error messages from the Adaptive Server error log. Use errorlog pipe max messages to tune the maximum number of messages returned. See *Performance and Tuning: Monitoring Tables*.

Enable the enable monitoring, errorlog pipe max messages, and errorlog pipe active configuration parameters for this monitoring table to collect data.

Columns

The columns for monErrorLog are:

| Name | Datatype | Description |
|---|---|---|
| SPID | smallint | Session process identifier (spid) |
| InstanceID | int | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| KPID | int | Kernel process identifier (kpid) |
| FamilyID | smallint | spid of the parent process |
| EngineNumber | smallint | Engine on which the process was running |
| ErrorNumber | int | Error message number |
| Severity | int | Severity of error. Adaptive Server versions 15.7 and later use a value of 99 to indicate stack traces; versions earlier than 15.7 use a value of 0. |
| State | int | State of error |
| Time | datetime | Timestamp when error occurred |
| ErrorMessage | varchar(512) | Text of the error message. Attribute is null. |

# monFailoverRecovery

Description

(Cluster environments only) Contains aggregated failover recovery diagnostic information for the cluster lock manager (CLM), database recovery, and cluster membership service (CMS) modules.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

monFailoverRecovery contains these columns:

| Column name | Type | Description |
|---|---|---|
| InstanceID | tinyint1 | Instance performing the recovery. |
| ModuleName | varchar(30) | Name of the module. One of CML, CMS, or Database |
| FailedInstanceID | tinyint1 | ID of the failed instance. |
| StartTime | datatime(8) | Start time for the module's recovery. |
| EndTime | datatime(8) | End time for the module's recovery. |

# monInmemoryStorage

Description

Provides information about inmemory devices configured to store the contents of inmemory databases.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

monInmemoryStorage contains these columns:

| Column name | Type | Description |
|---|---|---|
| InstanceID | tinyint | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| ID | int | ID of the data cache to which this device is bound. |
| DeviceNum | int | Device number. Always -1 for inmemory devices. |
| StartPage | int | Page ID for the first page in this device. |
| NumPage | int | Number of pages in this device. |
| SizeKB | int | Device size, in kilobytes. |
| Name | varchar(30) | Name of the data cache for this device. |
| DeviceName | varchar(30) | Name of the inmemory storage device. |
| Type | varchar(30) | The type of storage. Always set to “cache”. |
| Status | varchar(30) | Status of the device. |

# monIOController

Description — Provides information about I/O controllers.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns — The columns for monIOController are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| ControllerID | int | | ID of the I/O controller |
| KTID | int | | ID of the kernel task |
| EngineNumber | int | | Engine that owns this controller |
| BlockingPolls | bigint | Counter | Number of blocking polls |
| NonBlockingPolls | bigint | Counter | Number of nonblocking polls |
| EventPolls | bigint | Counter | Number of polls returning an event |
| NonBlockingEventPolls | bigint | Counter | Number of nonblocking polls returning an event |
| FullPolls | bigint | Counter | Number of polls returning the maximum number of events |
| Events | bigint | Counter | Number of events polled |
| EventHWM | bigint | Counter | Highest number of events returned in a single poll |
| Pending | int | Counter | Number of pending I/O operations |
| Completed | bigint | Counter | Number of completed I/O operations |
| Reads | bigint | Counter | Number of read or receive operations |
| Writes | bigint | Counter | Number of write or send operations |
| Deferred | bigint | Counter | Number of I/O operations deferred or delayed |
| Type | varchar(30) | | I/O controller type |

# monIOQueue

Description

Provides device I/O statistics displayed as data and log I/O for normal and temporary databases on each device.

Enable the enable monitoring configuration parameter for this monitoring table to collect data.

Columns

The columns for monIOQueue are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| IOs | int | Counter | Total number of I/O operations |
| IOTime | int | Counter | Amount of time (in milliseconds) spent waiting for I/O requests to be satisfied |
| LogicalName | varchar(30) | Null | Logical name of the device |
| IOType | varchar(12) | Null | Category for grouping I/O. One of UserData, UserLog, TempdbData, TempdbLog, or System. |

# monLicense

Description

Provides a list of all licences currently checked out by the Adaptive Server.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monLicense are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| Quantity | int | | Quantity of licenses used for this feature. |
| Name | varchar(30) | Null | Name of the feature license. |
| Edition | varchar(30) | Null | Edition of Adaptive Server for which this feature is licensed. |
| Type | varchar(64) | Null | License type. |
| Version | varchar(16) | Null | Version of the feature license in use |
| Status | varchar(30) | Null | Status of this feature license (that is, whether the license is withing a grace period expired). |
| LicenseExpiry | datetime | Null | Date that the license expires, if this is an expiring license. |
| GraceExpiry | datetime | Null | Date this license expires, if this license was awarded on grace. Refer to the Status column to determine whether this license was awarded a grace period. |
| LicenseID | varchar(150) | Null | License identifier. This may not be available if the license has been awarded a grace period. |
| Filter | varchar(14) | Null | Filter used when selecting this feature license. Use sp_lmconfig to change the filter. |
| Attributes | varchar(64) | Null | License attributes. These attributes are "*name=value*" pairs which, if specified, limit certain characteristics of Adaptive Server. Possible limiters are:<br>• ME = maximum number of engines<br>• MC = maximum number of connections<br>• MS = maximum number of disk space<br>• MM = maximum number of memory<br>• CP = maximum number of CPUs |

**Note** monLicense does not require mon_role permission; any user can use it.

# monLocks

Description — Returns a list of granted locks and pending lock requests.

Enable the enable monitoring configuration parameter for this monitoring table to collect data.

Columns — The columns for monLocks are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| SPID | smallint | | Session process identifier of process holding or requesting the lock. |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| KPID | int | | Kernel process identifier |
| DBID | int | | Unique identifier for this database object. |
| ParentSPID | smallint | | Parent process ID. |
| LockID | int | | Lock object ID. |
| Context | int | | Lock context (bit field). These values are the same as for those of the context column in syslocks. See the *Reference Manual: Tables* for information about syslocks. |
| DBName | varchar(30) | | Name of the database for the locked object. This column is NULL if the database is not open when monLocks is queried. |
| ObjectID | int | Null | Unique identifier for the object |
| LockState | varchar(20) | Null | Indicates if the lock is granted. Values are:<br>• Granted<br>• Requested |
| LockType | varchar(20) | Null | Type of lock. Values are:<br>• Exclusive<br>• Shared<br>• Update |
| LockLevel | varchar(30) | Null | The type of object for which the lock was requested. Values are:<br>• Row<br>• Page<br>• Table<br>• Address |
| WaitTIme | int | Null | The time (in seconds) for which the lock request was not granted. |
| PageNumber | int | Null | Page that is locked when `LockLevel = 'PAGE'` |
| RowNumber | int | Null | Row that is locked when `LockLevel = 'ROW'` |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| BlockedBy | int | | If the lock request is blocked, the BlockedBy column is the session process identifier for the process holding the lock that is blocking this lock request. Null if request is not blocked. |
| BlockedState | varchar(64) | | Lock state if the lock being held is blocking other lock requests or if the lock request is blocked. Values are:<br>• Blocked<br>• Blocking<br>• Demand<br>• Detached<br>• Null (if there is no blocking condition) |
| SourceCodeID | varchar(30) | | For internal use only. |

# monLockTimeout

Description

Provides information about lock timeouts. Each row identifies the object on which a blocked lock request occurred, and identities of the blocked and blocking processes.

You must enable the enable monitoring, lock timeout pipe active, and lock timeout pipe max messages configuration parameters for monLockTimeout monitoring table to collect data.

Columns

The columns in monLockTimeout are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| InstanceID | tinyint | | (Cluster environments only) ID of an instance in a cluster. |
| LockWaitPeriod | int | | Configured amount of time processes wait before a timeout occurs. |
| LockTimeoutLevel | varchar (20) | Null | Timeout level. One of:<br>• DTM_SERVER<br>• SERVER<br>• SESSION<br>• COMMAND<br>• INVALID |
| ObjectDBID | int | | Unique database identifier for database in which the object resides. |
| ObjectDBName | varchar(30) | Null | Name of database in which the object resides. |
| ObjectID | int | | Unique identifier for the object. |
| ObjectName | varchar(255) | Null | Name of the object. |
| PageNumber | int | | Page number requested for the lock, if applicable. |
| RowNumber | int | | Row number requested for the lock, if applicable. |
| ExpiredAtTime | datetime | | Time when lock expires. |
| HeldSPID | int | | Server process ID (spid) of process holding the lock. |
| HeldKPID | int | | Kernel process ID (kpid) of process holding the lock. |
| HeldUserName | varchar(30) | Null | Name of the user for whom the lock is held. |
| HeldApplName | varchar(30) | Null | Name of the application holding the lock. |
| HeldHostName | varchar(30) | Null | Name of the host machine on which the application that executed the query holding the lock is running. |
| HeldClientName | varchar(30) | Null | Value of the clientname property set by the application holding the lock. |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| HeldClientApplName | varchar(30) | Null | Value for the *clientapplname* property set by the application holding the lock. |
| HeldClientHostName | varchar(30) | Null | Value for the *clienthostname* property set by the application holding the lock. |
| HeldTranName | varchar(255) | Null | Name of the transaction that acquired the lock. |
| HeldCommand | varchar(30) | Null | Category of process or command the process was executing when the process was blocked. |
| HeldFamilyID | int | | spid of the parent process holding the lock. |
| HeldProcDBID | int | | Unique identifier for the database where the stored procedure that caused the lock to be held resides, if applicable. |
| HeldProcDBName | varchar(30) | Null | Name for the database where the stored procedure that caused the lock to be held resides, if applicable. |
| HeldProcedureName | varchar(255) | Null | Name for the stored procedure that caused the lock to be held, if applicable. |
| HeldBatchID | int | | Identifier of the SQL batch executed by the process holding the lock when the lock timeout occurred. |
| HeldContextID | int | | Unique context identifier for the process holding the lock when it was blocked by another process (not when it acquired the lock). |
| HeldLineNumber | int | | Line number in the SQL batch of the SQL statement holding the lock. |
| HeldStmtNumber | int | | Statement number in the SQL batch of the SQL statement holding the lock. |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| HeldLockType | varchar(20) | Null | Type of lock. One of:<br>• Exclusive table<br>• Shared table<br>• Exclusive intent<br>• Shared intent<br>• Exclusive page<br>• Shared page<br>• Update page<br>• Exclusive row<br>• Shared row<br>• Update row<br>• Next key<br>• Exclusive address<br>• Shared address<br>• Semaphore |
| HeldNumLocks | int | | Number of locks currently held by holding spid. |
| HeldNumTimeoutsCausedByTran | int | | Number of timeouts caused by this holding transaction. |
| HeldNumTimeoutsCausedByLock | int | | Number of timeouts caused by this lock resource. |
| HeldSourceCodeID | varchar(30) | Null | Location of the source code where the lock being held was acquired (internal use only). |
| WaitSPID | int | | spid of the process waiting for the lock. |
| WaitKPID | int | | kpid of the process waiting for the lock. |
| WaitUserName | varchar(30) | Null | Name of the user for whom the lock is being requested. |
| WaitApplName | varchar(30) | Null | Name of the application waiting for the lock. |
| WaitHostName | varchar(30) | Null | Name of the host running the process waiting for the lock. |
| WaitClientName | varchar(30) | Null | Value of the *clientname* property set by the application waiting for the lock. |
| WaitClientApplName | varchar(30) | Null | Value of the *clientapplname* property set by the application waiting for the lock. |
| WaitClientHostName | varchar(30) | Null | Value of the *clienthostname* property set by the application waiting for the lock. |
| WaitTranName | varchar(255) | Null | Name of the transaction in which the lock was requested. |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| WaitCommand | varchar(30) | Null | Category of process or command that the process was executing when it was blocked and then timed out. |
| WaitFamilyID | int | | spid of the parent process waiting for the lock. |
| WaitProcDBID | int | | Unique identifier for the database in which the stored procedure waiting for the lock resides, if applicable. |
| WaitProcDBName | varchar(255) | Null | Name for the database where the stored procedure that is waiting for the lock resides, if applicable. |
| WaitProcedureName | varchar(255) | Null | Name for the stored procedure waiting for the lock, if applicable. |
| WaitBatchID | int | | Identifier of the SQL batch executed by the process waiting for the lock when the lock timeout occurred. |
| WaitContextID | int | | Unique context identifier for the process waiting for the lock when it was blocked by another process. |
| WaitLineNumber | int | | Line number of the SQL statement in the SQL batch waiting for the lock. |
| WaitStmtNumber | int | | Line number in SQL batch waiting for the lock. |
| WaitLockType | varchar(30) | Null | Type of lock. One of:<br>• Exclusive table<br>• Shared table<br>• Exclusive intent<br>• Shared intent<br>• Exclusive page<br>• Shared page<br>• Update page<br>• Exclusive row<br>• Shared row<br>• Update row<br>• Next key<br>• Exclusive address<br>• Shared address<br>• Semaphore |
| WaitNumTimeoutsCausedByTran | int | | Number of timeouts caused by a waiting transaction. |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| WaitSourceCodeID | int | | Location in the source code when the timeout occured and the waiting lock request was made (for internal use only). |
| HeldProcedureID | int | | Unique object identifier for the stored procedure that the blocking process was executing when the timeout occurred |
| WaitProcedureID | int | | Unique object identifier for the stored procedure that is waiting for the lock, if applicable |

# monLogicalCluster

Description

(Cluster environments only) Displays information about the logical clusters currently configured on the system.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monLogicalCluster are:

| Name | Datatype | Description |
|---|---|---|
| LCID | int | Logical cluster ID. |
| Attributes | int | Bitmask of logical cluster attributes. |
| ActiveConnections | int | Number of active connections using this logical cluster. |
| BaseInstances | tinyint | Number of instances configured as base instances for this logical cluster. |
| ActiveBaseInstances | tinyint | Number of base instances on which this logical cluster is currently active. |
| FailoverInstances | tinyint | Number of instances configured as failover instances for this logical cluster. |
| ActiveFailoverInstances | tinyint | Number of failover instances on which this logical cluster is currently active. |
| Name | varchar(30) | Logical cluster name. |
| State | varchar(20) | Current state. One of:<br>• Online<br>• Offline<br>• Failed<br>• Inactive<br>• Time_wait |
| DownRoutingMode | varchar(20) | Down routing-mode setting. One of:<br>• System<br>• Open<br>• Disconnect |
| FailoverMode | varchar(20) | Failover mode setting, `instance` or `cluster`. |
| StartupMode | varchar(20) | Start-up mode setting, `automatic` or `manual`. |
| SystemView | varchar(20) | System view setting, `instance` or `cluster`. |
| Roles | varchar(20) | Comma-delimited list of special roles for this logical cluster. The "system" logical cluster always has the system role. The open logical cluster has the "open" role. If the system logical cluster also has the open role, the value for this column is `system, open`. Logical clusters without any special roles return a null value. |

| Name | Datatype | Description |
|---|---|---|
| LoadProfile | varchar(30) | Load profile associated with this logical cluster. |
| ActionnRelease | varchar(20) | The current action release mode for this logical cluster. Values are:<br>• Manual<br>• Automatic<br>Manual indicates that the user must execute the action release command to release the actions for this cluster. |
| Gather | varchar(30) | Indicates whether this logical cluster is configured to automatically gather routable connections to this logical cluster. Values are:<br>• Manual<br>• Automatic |

# monLogicalClusterAction

Description

(Cluster environments only) Shows all administrative actions against logical clusters from start-up until these actions are released.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monLogicalClusterAction are:

| Name | Datatype | Description |
|---|---|---|
| Handle | int | Unique handle used to cancel this action. |
| State | varchar(20) | State of the action: `active`, `complete`, `releasing`, or `canceled`. |
| LCID | int | Logical cluster ID to which this action applies. |
| LogicalClusterName | varchar(30) | Logical cluster name of this logical cluster (denormalized to reduce joins). |
| Action | varchar(15) | Action being performed. A combination of the command running and its scope. For example, offline instance or failover cluster. |
| FromInstances | varchar(96) | A comma-separated list of `from instances` for this command and action (instance being brought offline). |
| ToInstances | varchar(96) | A comma-separated list of `to instances` for this command and action (instances being brought online). |
| InstancesWaiting | int | Number of instances waiting to go offline (this is a count of FromInstances that are in the `time_wait` state). |
| WaitType | varchar(20) | Current wait state for this action. One of: `wait`, `until`, or `nowait`. |
| StartTime | datetime | Date and time the command was issued. |
| Deadline | datetime | Date and time the command must be finished (based on the time value supplied to the `wait` or `until` options). |
| CompleteTime | datetime | Date and time the command and action completed (when InstancesWaiting is zero and the action went from `active` to the `complete` state). Returns NULL for incomplete actions. |
| ConnectionsRemaining | int | Number of connections remaining to move as a result of this command. |
| NonMigConnections | int | Number of connections to be terminated because they do not support the migration protocol. |
| NonHAConnections | int | Number of connections that do not support the high availability failover protocol. These connections are disconnected and cannot fail over when the command finishes. |

# monLogicalClusterInstance

Description

(Cluster environments only) Displays information about the many-to-many relationship between instances and logical clusters.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monLogicalClusterInstance are:

| Name | Datatype | Description |
|---|---|---|
| LCID | int | Logical cluster ID |
| LogicalClusterName | varchar(30) | Logical cluster name |
| InstanceID | tinyint | ID of the instance within the cluster |
| InstanceName | varchar(30) | Instance name |
| Type | varchar(20) | Instance type |
| FailoverGroup | tinyint | Failover group to which this instance is a member (failover instances only) |
| State | varchar(20) | State of this instance with respect to the logical cluster |
| ActiveConnections | int | Number of active connections for this logical cluster on this instance |
| NonMigConnections | int | Number of active connections that do not support the connection migration protocol |
| NonHAConnections | int | Number of active connections that do not support the high availability failover protocol |
| LoadScore | real | Workload score for this instance using the load profile associated with its logical cluster |

# monLogicalClusterRoute

Description

(Cluster environments only) Displays information about the configured routes (application, login, and alias bindings). You need not have the mon_role role to query this monitor table.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monLogicalClusterRoute are:

| Name | Datatype | Description |
|---|---|---|
| LCID | int | Logical cluster ID |
| LogicalClusterName | varchar(30) | Logical cluster name |
| RouteType | varchar(20) | Route type. One of: `application`, `login`, or `alias` |
| RouteKey | varchar(30) | Application, login, or alias name associated with this route. |

# monNetworkIO

Description

Returns network I/O statistics for all communication between Adaptive Server and client connections.

Enable the enable monitoring configuration parameter for this monitoring table to collect data.

Columns

The columns for monNetworkIO are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster |
| PacketsSent | int | Counter, reset | Number of packets sent |
| PacketsReceived | int | Counter, reset | Number of packets received |
| BytesSent | int | Counter, reset | Number of bytes sent |
| BytesReceived | int | Counter, reset | Number of bytes received |
| PacketsSentMln | int | | Number of packets, in millions, sent by the server |
| PacketsReceivedMln | int | | Number of packets, in millions, received by the server |
| BytesSentMB | int | | Number of bytes, in megabytes, sent by the server |
| BytesReceivedMB | int | | Number of bytes, in megabytes, received by the server |

# monOpenDatabases

Description

Provides state and statistical information pertaining to databases that are currently in the server's metadata cache.

If the value of number of open databases is too low, Adaptive Server may flush database descriptors from the metadata cache. If this occurs, Adaptive Server loses the database statistics, but the statistics are reinitialized the next time the database descriptor is installed in the metadata cache.

Enable the enable monitoring configuration parameter for this monitoring table to collect data.

Columns

The columns for monOpenDatabases are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| DBID | int | | Unique identifier for the database |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| BackupInProgress | int | | Specifies whether a backup is currently in progress for the database |
| LastBackupFailed | int | | Specifies whether the last backup of the database failed |
| TransactionLogFull | int | | Specifies whether the database transaction log is full |
| AppendLogRequests | int | Counter | Number of semaphore requests when attempting to append to the database transaction log |
| AppendLogWaits | int | Counter | Number of times a task had to wait for the append log semaphore to be granted |
| DBName | varchar(30) | Null | Name of the database |
| BackupStartTime | datetime | Null | Date the last full database backup started |
| SuspendedProcesses | int | Null | Number of processes currently suspended due to the database transaction log being full |
| QuiesceTag | varchar(30) | Null | Tag used in the quiesce database command for this database if the database is in a quiesced state |
| LastCheckpointTime | datetime | Null | Date and time checkpoint last ran for this database |
| LastTranLogDumpTime | datetime | Null | Date and time of this database's most recently successful transaction log dump.<br>The time is not updated if the transaction is dumped using the truncate_only or no_log. |
| PRSUpdateCount | int | Counter | Number of updates to precomputed result sets caused by insert, update, or deletes to the base table. |
| PRSSelectCount | int | Counter | The number of times the optimizer selected precomputed result sets in this database when generating a query plan. |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| PRSRewriteCount | int | Counter | The number of times the optimizer determined the precomputed result sets in this database were valid when generating the query plan. |

# monOpenObjectActivity

Description — Provides statistics for all open tables and indexes.

Enable the enable monitoring, per object statistics active, and object lockwait timing configuration parameters for this monitoring table to collect data.

Columns — The columns for monOpenObjectActivity are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| DBID | int | | Unique identifier for the database. |
| ObjectID | int | | Unique identifier for the object. |
| IndexID | int | | Unique identifier for the index.. |
| InstanceID | int | | (Cluster environments only) Unique identifier for an instance. |
| DBName | varchar(30) | Null | Name of the database in which the object resides |
| ObjectName | varchar(30) | Null | Name of the object. |
| LogicalReads | int | Counter, null | Total number of times a buffer for this object has been retrieved from a buffer cache without requiring a read from disk. |
| PhysicalReads | int | Counter, null | Number of buffers read from disk. |
| APFReads | int | Counter, null | Number of APF buffers read from disk. |
| PagesRead | int | Counter, null | Total number of pages read. |
| PhysicalWrites | int | Counter, null | Total number of buffers written to disk. |
| PagesWritten | int | Counter, null | Total number of pages written to disk. |
| RowsInserted | int | Counter, null | Number of rows inserted. |
| RowsDeleted | int | Counter, null | Number of rows deleted. |
| RowsUpdated | int | Counter, null | Number of updates. |
| Operations | int | Counter, null | Number of times the object was accessed. |
| LockRequests | int | Counter, null | Number of requests for a lock on the object. |
| LockWaits | int | Counter, null | Number of times a task waited for an object lock. |
| OptSelectCount | int | Counter, null | Number of times the optimizer selected this index to be used in a query plan. |
| LastOptSelectDate | datetime | Null | Last date the index was selected for a plan during compilation. |
| UsedCount | int | Counter, null | Number of times the object was used in a plan during execution. |
| LastUsedDate | datetime | Null | Last date the index was used in a plan during execution. |
| HkgcRequests | int | | Total number of events queued for an object. A large value implies the system is generating large amounts of garbage for the specified object. |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| HkgcPending | int | | The number of pending events for an object. A large value implies that a lot of garbage is yet to be collected, although the housekeeper will clean it up. If you reboot Adaptive Server, all entries in the housekeeper queue are lost, and the garbage from those pages is not collected when you restart Adaptive Server. |
| HkgcOverflows | int | | The number of overflow object events. A large value implies the housekeeper queues are filling up. Generated garbage will not then be cleaned up because the housekeeper cannot schedule the job. |
| PhysicalLocks | int | | (Cluster environments only) Number of physical locks requested per object. |
| PhsycialLocksRetained | int | | (Cluster environments only) Number of physical locks retained. Use to identify the lock hit ratio for each object. Good hit ratios imply balanced partitioning for this object. |
| PhysicalLocksRetainWaited | int4 | | (Cluster environments only) Number of physical lock requests waiting before a lock is retained. |
| PhysicalLocksDeadlocks | int | | (Cluster environments only) Number of times a requested physical lock returned a deadlock. The Cluster Physical Locks subsection of sp_sysmon uses this counter to report deadlocks while acquiring physical locks for each object. |
| PhysicalLocksWaited | int | | (Cluster environments only) Number of times an instance waited for a physical lock request. |
| PhysicalLocksPageTransfer | int | | (Cluster environments only) Number of page transfers that occurred when an instance requested a physical lock. The Cluster Physical Locks subsection of sp_sysmon uses this counter to report the node-to-node transfer and physical-lock acquisition as a node affinity ratio for this object |
| TransferReqWaited | int4 | | (Cluster environments only) Number of times physical lock requests waiting before receiving page transfers. |
| AvgPhysicalLocksWaitTime | int4 | | (Cluster environments only) Average amount of time clients spend before the physical lock is granted. |
| MaxPhysicalLockWaitTime | real | | (Cluster environments only) Maximum amount of time this object waited for before a physical lock was granted. |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| AvgTransferReqWaitTime | int4 | | (Cluster environments only) Average amount of time physical lock requests wait before receiving page transfers. |
| MaxTransferReqWaitTime | real | | (Cluster environments only) Maximum amount of time physical lock requests waited to receive page transfers. |
| TotalServiceRequests | int4 | | (Cluster environments only) Number of physical lock requests serviced by the cluster cache manager of an instance. |
| PhysicalLocksDowngraded | int4 | | (Cluster environments only) Number of physical lock downgrade requests serviced by the cluster cache manager of an instance. |
| PagesTransferred | int4 | | (Cluster environments only) Number of pages transferred at an instance by the cluster cache manager. |
| ClusterPageWrites | int4 | | (Cluster environments only) Number of pages written to disk by the cluster cache manager of an instance. |
| AvgServiceTime | int4 | | (Cluster environments only) Average amount of service time spent by the cluster cache manager of an instance. |
| MaxServiceTime | real | | (Cluster environments only) Maximum amount of service time spent by the cluster cache manager of an instance. |
| AvgQueueWaitTime | real | | (Cluster environment only) Average amount of time, in milliseconds, spent waiting for Adpative Server to complete buffer transfers for this object. |
| MaxQueueWaitTime | real | | (Cluster environment only) Maximum amount of time, in milliseconds, spent waiting for Adaptive Server to complete a buffer transfer for this object . |
| AvgTimeWaitedOnLocalUsers | int4 | | (Cluster environments only) Average amount of time, in milliseconds, an instance's cluster cache manager waited because of page use by users on this instance. |
| MaxTimeWaitedOnLocalUsers | real | | (Cluster environments only) Maximum amount of time, in milliseconds, an instance's cluster cache manager waited because of page use by users on this instance. |
| AvgTransferSendWaitTime | int4 | | (Cluster environments only) Average amount of time an instance's cluster cache manager spends for page transfer. |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| MaxTransferSendWaitTime | real | | (Cluster environments only) Maximum amount of time an instance's cluster cache manager waited for a page transfer to complete. |
| AvgIOServiceTime | int4 | | (Cluster environments only) Average amount of time used by an instance's cluster cache manager for page transfer. |
| MaxIOServiceTime | real | | (Cluster environments only) Maximum amount of time the Cluster Cache Manager took to write pages to disk. |
| AvgDowngradeServiceTime | int4 | | (Cluster environments only) Average amount of time the cluster cache manager uses to downgrade physical locks. |
| MaxDowngradeServiceTime | real | | (Cluster environments only) Maximum time a task spent waiting for the physical lock to be downgraded on a page. |
| SharedLockWaitTime | int | Counter, reset, null | The total amount of time, in milliseconds, that all tasks spent waiting for a shared lock. |
| ExclusiveLockWaitTime | int | Counter, reset, null | The total amount of time, in milliseconds, that all tasks spent waiting for an exclusive lock. |
| UpdateLockWaitTime | int | Counter, reset, null | The total amount of time, in milliseconds, that all tasks spent waiting for an update lock. |
| ObjectCacheDate | datetime | Counter, reset, null | Indicates the date and time when the object was added to the cache. |
| PRSSelectCount | int | Counter, null | The number of times the precomputed result set was used in a query. |
| LastPRSSelectDate | datetime | null | Date for the last time the precomputed result set was used in a query. |
| PRSRewriteCount | int | Counter, null | Number of times the optimizer determined that the precomputed result set was valid for use in a query. the optimizer may not have used the precomputed result set because it found a better choice. |
| LastPRSRewriteDate | datetime | null | Date for the last time the optimizer determined that the precomputed result set was valid for use in a query. |

**Note** The value of OptSelectCount may be less than that of UsedCount since you can use the plan for a stored procedure or trigger multiple times. Also, because Adaptive Server may decide not to execute certain portions of a query plan during execution, UsedCount may be less than OptSelectCount.

# monOpenPartitionActivity

Description — Provides information about the use of each open partition on the server.

Enable the enable monitoring and per object statistics active configuration parameters for this monitoring table to collect data.

Columns — The columns for monOpenPartitionActivity are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| DBID | int | | Unique identifier for the database. |
| ObjectID | int | | Unique identifier for the object. |
| IndexID | int | | Unique identifier for the index. |
| PartitionID | int | | Unique identifier for the partition. |
| InstanceID | int | | ID of an instance in a shared-disk cluster. |
| DBName | varchar(30) | Null | Name of the database in which the object resides. |
| ObjectName | varchar(30) | Null | Name of the object. |
| PartitionName | varchar(30) | Null | Name of the partition. |
| LogicalReads | int | Counter, null | Total number of buffers read. |
| PhysicalReads | int | Counter, null | Number of buffers read from disk. |
| APFReads | int | Counter, null | Number of asynchronous prefetch (APF) buffers read. |
| PagesRead | int | Counter, null | Total number of pages read. |
| PhysicalWrites | int | Counter, null | Total number of buffers written to disk. |
| PagesWritten | int | Counter, null | Total number of pages written to disk. |
| RowsInserted | int | Counter, null | Number of rows inserted. |
| RowsDeleted | int | Counter, null | Number of rows deleted. |
| RowsUpdated | int | Counter, null | Number of updates. |
| OptSelectCount | int | Counter, null | Number of times object was selected for plan during compilation. |
| LastOptSelectDate | datetime | Null | Last date the index was selected for plan during compilation. |
| UsedCount | int | Counter, null | Number of times the object was used in a plan during execution. |
| LastUsedDate | datetime | Null | Last date the index was used in a plan during execution. |
| HkgcRequests | int | | Total number of events queued for a partition. A large value implies the system is generating large amounts of garbage for the specified partition. |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| HkgcPending | int | | The number of pending events for a partition. A large value implies that a lot of garbage is yet to be collected, although the housekeeper will clean it up. If you reboot Adaptive Server, all entries in the housekeeper queue are lost, and the garbage from those pages is not collected when you restart Adaptive Server. |
| HkgcOverflows | int | | The number of overflow partition events. A large value implies the housekeeper queues are filling up. Generated garbage will not then be cleaned up because the housekeeper cannot schedule the job. |
| PhysicalLocks | int | | (Cluster environments only) Number of physical locks requested per object. |
| PhsycialLocksRetained | int | | Number of physical locks retained. Use to identify the lock hit ratio for each object. Good hit ratios imply balanced partitioning for this object. |
| PhysicalLocksRetainWaited | int4 | | (Cluster environments only) Number of physical lock requests waiting before a lock is retained. |
| PhysicalLocksDeadlocks | int | | (Cluster environments only) Number of times a physical lock requested returned a deadlock. The Cluster Physical Locks subsection of sp_sysmon uses this counter to report deadlocks while acquiring physical locks for each object. |
| PhysicalLocksWaited | int | | (Cluster environments only) Number of times an instance waited for a physical lock request. |
| PhysicalLocksPageTransfer | int | | (Cluster environments only) Number of page transfers that occurred when an instance requested a physical lock. The Cluster Physical Locks subsection of sp_sysmon uses this counter to report the node-to-node transfer and physical-lock acquisition as a node affinity ratio for this object. |
| TransferReqWaited | int4 | | (Cluster environments only) Number of times physical lock requests waiting before receiving page transfers. |
| MaxPhysicalLockWaitTime | real | | (Cluster environments only) Maximum amount of time this object waited for before a physical lock was granted. |
| AvgPhysicalLockWaitTime | int4 | | (Cluster environments only) Average amount of time clients spend before the physical lock is granted. |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| HkgcPending | int | | The number of pending events for a partition. A large value implies that a lot of garbage is yet to be collected, although the housekeeper will clean it up. If you reboot Adaptive Server, all entries in the housekeeper queue are lost, and the garbage from those pages is not collected when you restart Adaptive Server. |
| HkgcOverflows | int | | The number of overflow partition events. A large value implies the housekeeper queues are filling up. Generated garbage will not then be cleaned up because the housekeeper cannot schedule the job. |
| PhysicalLocks | int | | (Cluster environments only) Number of physical locks requested per object. |
| PhsycialLocksRetained | int | | Number of physical locks retained. Use to identify the lock hit ratio for each object. Good hit ratios imply balanced partitioning for this object. |
| PhysicalLocksRetainWaited | int4 | | (Cluster environments only) Number of physical lock requests waiting before a lock is retained. |
| PhysicalLocksDeadlocks | int | | (Cluster environments only) Number of times a physical lock requested returned a deadlock. The Cluster Physical Locks subsection of sp_sysmon uses this counter to report deadlocks while acquiring physical locks for each object. |
| PhysicalLocksWaited | int | | (Cluster environments only) Number of times an instance waited for a physical lock request. |
| PhysicalLocksPageTransfer | int | | (Cluster environments only) Number of page transfers that occurred when an instance requested a physical lock. The Cluster Physical Locks subsection of sp_sysmon uses this counter to report the node-to-node transfer and physical-lock acquisition as a node affinity ratio for this object. |
| TransferReqWaited | int4 | | (Cluster environments only) Number of times physical lock requests waiting before receiving page transfers. |
| MaxPhysicalLockWaitTime | real | | (Cluster environments only) Maximum amount of time this object waited for before a physical lock was granted. |
| AvgPhysicalLockWaitTime | int4 | | (Cluster environments only) Average amount of time clients spend before the physical lock is granted. |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| MaxTransferReqWaitTime | real | | (Cluster environments only) Maximum amount of time physical lock requests waited to receive page transfers. |
| AvgTransferReqWaitTime | int4 | | (Cluster environments only) Average amount of time physical lock requests wait before receiving page transfers. |
| TotalServiceRequests | int4 | | (Cluster environments only) Number of physical lock requests serviced by the cluster cache manager of an instance. |
| PhysicalLocksDowngraded | int4 | | (Cluster environments only) Number of physical lock downgrade requests serviced by the cluster cache manager of an instance. |
| PagesTransferred | int4 | | (Cluster environments only) Number of pages transferred at an instance by the cluster cache manager. |
| ClusterPageWrites | int4 | | (Cluster environments only) Number of pages written to disk by the cluster cache manager of an instance. |
| AvgServiceTime | int4 | | (Cluster environments only) Average amount of time spent by the cluster cache manager of an instance. |
| MaxServiceTime | real | | (Cluster environments only) Maximum amount of time spent by the cluster cache manager of an instance. |
| AvgQueueWaitTime | int | | (Cluster environment only) Average amount of time, in milliseconds, spent waiting for Adaptive Server to complete buffer transfers for this object. |
| MaxQueueWaitTime | int | | (Cluster environments only) Maximum amount of time, in milliseconds, spent waiting for Adaptive Server to complete a buffer transfer for this object . |
| AvgTimeWaitedOnLocalUsers | int4 | | (Cluster environments only) Average amount of service time an instance's cluster cache manager waits because of page use by users on this instance. |
| MaxTimeWaitedOnLocalUsers | real | | (Cluster environments only) Maximum amount of time, in milliseconds, an instance's cluster cache manager waited for a physical lock because the object in question was in use by another process. |
| AvgTransferSendWaitTime | int4 | | (Cluster environments only) Average amount of service time an instance's cluster cache manager spends for page transfer. |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| MaxTransferSendWaitTime | real | | (Cluster environments only) Maximum amount of time the Cluster Cache Manager for an instance waited for page transfer to complete |
| AvgIOServiceTime | int4 | | (Cluster environments only) Average amount of service time used by an instance's cluster cache manager for page transfer. |
| MaxIOServiceTime | real | | (Cluster environments only) Maximum amount of time the Cluster Cache Manager took to write pages to disk. |
| AvgDowngradeServiceTime | int4 | | (Cluster environments only) Average amount of time the cluster cache manager uses to downgrade physical locks. |
| MaxDowngradeServiceTime | real | | (Cluster environments only) Maximum time a task spent waiting for the physical lock to be downgraded on a page. |
| ObjectCacheDate | datetime | Counter, reset, null | Indicates the date and time when the object was added to the cache. |
| HkgcRequestsDcomp | int | | Total number of data pages of the partition that were queued for page compression |
| HkgcPendingDcomp | int | | Number of data pages of the partition that are still pending for page compression |
| HkgcOverflowsDcomp | int | | Total number of pages that could not be compressed because the housekeeper queue was full. |
| IOSize1Page | int | | Number of IO operations performed for each IO one page in size |
| IOSize2Pages | int | | Number of IO operations performed for each IO that is 2 pages in size |
| IOSize4Pages | int | | Number of IO operations performed for each IO that is 4 pages is size |
| IOSize8Pages | int | | Number of IO operations performed for each IO that is 8 pages in size |

**Note** Because you can use the plan for a stored procedure or trigger multiple times, the value of the OptSelectCount column may be less than the value of UsedCount. In addition, because the Adaptive Server may decide not to execute certain portions of a query plan during execution, the UsedCount may be less than the OptSelectCount.

# monPCIBridge

Description

Contains information about the Java PCI Bridge. This table provides information about the Java environment.

You do not need to enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monPCIBridge are:

| Name | Datatype | Description |
|---|---|---|
| InstanceID | tinyint | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| Status | char(10) | Current status of the PCI Bridge. Values are:<br>• ACTIVE<br>• DOWN |
| ConfiguredSlots | int | Number of configured slots. Set using max pci slots configuration parameter. |
| ActiveSlots | int | Number of currently active slots. |
| ConfiguredPCIMemoryKB | int | Total memory configured for the PCI Bridge using the pci memory configuration parameter. |
| UsedPCIMemoryKB | int | Total memory currently used by the PCI bridge and its components. |

# monPCIEngine

Description

Displays engine information for the PCI Bridge and its plug-ins. This table provides information about the Java environment.

You do not need to enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monPCIEngine are:

| Name | Datatype | Description |
|---|---|---|
| InstanceID | tinyint | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| Engine | int | Engine number |
| Status | char(10) | Status of the plug-in on the engine. Values are:<br>• ACTIVE<br>• INIT |
| PLBStatus | char(10) | Status of the PCI Launcher Boss. Values are:<br>• ACTIVE<br>• DOWN |
| NumberofActiveThreads | int | Number of active threads currently under control of the PCI Launcher Boss. |
| PLBRequests | int | Number of requests for the PCI Launcher Boss to execute a function for a native thread. |
| PLBwakeupRequests | int | Number of times the PCI Launcher Boss received a wake-up to execute work for a native thread. |

# monPCISlots

Description

Contains information about the plug-in bound to each slot in the PCI Bridge. This table provides information about the Java environment.

You do not need to enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monPCISlots are:

| Name | Datatype | Description |
|---|---|---|
| InstanceID | tinyint | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| Slot | int | Number of active slot: Values are 1 – 31. |
| Status | char(10) | Status of the slot. Values are:<br>• INIT<br>• IN USE<br>• STOPPED |
| Modulename | varchar(30) | Logical module name bound to the current slot. |
| engine | int | Engine associated with the slot. |

# monPCM

Description

(Cluster environments only) Tracks the peer coordination module (PCM) client activities in the cluster (for example, the number of fragment that were sent and received), and contains a row for each PCM client.

You do not need to enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monPCM are:

| Column name | Type | Description |
|---|---|---|
| InstanceID | int1 | Instance ID for which the information is collected |
| Sent | int4 | Number of messages sent per module |
| Fragments_sent | int4 | Number of fragments sent per module |
| Fragments_received | int4 | Number of fragments received per module |
| Received | int4 | Number of messages received per module |
| Reply | int4 | Number of replies received per module |
| Unicast | int4 | Number of unicast messages sent per module |
| Mulicat | int4 | Number of multicast messages sent per module |
| Sync | int4 | Number of synchronous messages sent per module |
| Async | int4 | Number of asynchronous messages sent per module |
| MinBytes | int4 | Minimum number of bytes transferred per message |
| AvgBytes | int4 | Average number of bytes transferred per message |
| MaxBytes | int4 | Maximum number of bytes transferred per message |
| MinDialog | int4 | Minimum length of the dialogues |
| AvgDialog | int4 | Average length of the dialogues |
| MaxDialog | int4 | Maximum length of the dialogues |
| Dialog | int4 | Number of the dialogues |
| MinTimeSyncApi | flt4 | Minimum time spent in PCM API in synchronous mode in the PCM layer per module |
| AvgTimeSyncApi | flt4 | Average time spent in PCM API in synchronous mode in the PCM layer per module |
| MaxTimeSyncApi | flt4 | Maximum time spent in PCM API in synchronous mode in the PCM layer per module |
| MinTimeAsyncApi | flt4 | Minimum time spent in PCM API in asynchonous mode in the PCM layer per module |
| AvgTimeAsyncApi | flt4 | Average time spent in PCM API in asynchronous mode in the PCM layer per module |
| MaxTimeAsyncApi | flt4 | Maximum time spent in PCM API in asynchronous mode in the PCM layer per module |

| Column name | Type | Description |
|---|---|---|
| MinTimeCIPCMsgAlloc | flt4 | Minimum time spent in cipcmsg allocations in the PCM layer per module |
| AvgTimeCIPCMsgAlloc | flt4 | Average time spent in cipcmsg allocations in the PCM layer per module |
| MaxTimeCIPCMsgAlloc | flt4 | Maximum time spent in cipcmsg allocations in the PCM layer per module |
| MinTimeCIPCSendCB | flt4 | Minimum time spent in cipc_sendcb per module |
| AvgTimeCIPCSendCB | flt4 | Average time spent in cipc_sendcb per module |
| MaxTimeCIPCSendCB | flt4 | Maximum time spent in cipc_sendcb per module |
| MinTimeCIPCUnicastsmsg | flt4 | Minimum time spent in CIPC while sending the unicasts message per module |
| AvgTimeCIPCUnicastsmsg | flt4 | Average time spent in CIPC while sending the unicasts message per module |
| MaxTimeCIPCUnicastsmsg | flt4 | Maximum time spent in CIPC while sending the unicasts message per module |
| MinTimeCIPCMulticastsmsg | flt4 | Minimum time spent in CIPC while sending the multicasts message per module |
| AvgTimeCIPCMulticastsmsg | flt4 | Average time spent in CIPC while sending the multicasts message per module |
| MaxTimeCIPCMulticastsmsg | flt4 | Maximum time spent in CIPC while sending the multicasts message per module |
| MinTimeClientRecvCB | flt4 | Minimum time spent in client receive callback in the PCM layer per module |
| AvgTimeClientRecvCB | flt4 | Average time spent in client receive callback in the PCM layer per module |
| MaxTimeClientRecvCB | flt4 | Maximum time spent in client receive callback in the PCM layer per module |
| ModuleName | int4 | Name of the PCM client |

# monProcedureCache

Description — Returns statistics relating to Adaptive Server procedure cache.

Enable the enable monitoring configuration parameter for this monitoring table to collect data.

Columns — The columns for monProcedureCache are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| Requests | int | Counter, reset | Number of stored procedures requested |
| Loads | int | Counter, reset | Number of stored procedures loaded into cache |
| Writes | int | Counter, reset | Number of times a procedure was normalized and the tree written back to sysprocedures |
| Stalls | int | Counter, reset | Number of times a process had to wait for a free procedure cache buffer when installing a stored procedure into cache |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |

# monProcedureCacheMemoryUsage

Description

Includes one row for each procedure cache allocator. An allocator is identified by an allocator ID, which is internal to Adaptive Server.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monProcedureCacheMemoryUsage are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| InstanceID | tinyint | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| AllocatorID | int | | Allocator ID |
| ModuleID | int | | Module ID (internal to Adaptive Server) |
| Active | int | | Number of memory pages (2KB) currently allocated to this allocator |
| HWM | int | | Maximum number of memory pages allocated since the server was started |
| ChunkHWM | int | | Largest number of contiguous memory pages allocated since the server was started |
| AllocatorName | varchar(30) | | Name of the allocator |
| NumReuseCaused | int | Null | Number of times this allocator has caused replacement |

# monProcedureCacheModuleUsage

Description

Includes one row for each module that allocates memory from procedure cache. A module, which is identified with a module ID, is a functional area classification internal to Adaptive Server procedure cache management.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monProcedureCacheModuleUsage are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| InstanceID | tinyint | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| ModuleID | int | | A module ID |
| Active | int | | Number of memory pages (2KB) currently allocated to this module |
| HWM | int | | The maximum number of memory pages allocated since the server was started |
| NumPagesReused | int | Null | Number of pages allocated to this module |
| ModuleName | varchar(30) | | Name of the module |

# monProcess

Description

Provides detailed statistics about processes that are currently executing or waiting.

Enable the enable monitoring and wait event timing configuration parameters for this monitoring table to collect data.

Columns

The columns for monProcess are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| SPID | smallint | | Session process identifier |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| KPID | int | | Kernel process identifier |
| ServerUserID | int | | Server user ID (SUID) of the user associated with this process. |
| BatchID | int | | Unique identifier for the SQL batch containing the executing statement |
| ContextID | int | | A unique identifier generated each time an executing query causes a stored procedure, trigger, execute immediate, deferred compilation, or other compiled object execution to occur |
| LineNumber | int | | Line number of the current statement within the SQL batch |
| SecondsConnected | int | | Number of seconds since this connection was established |
| DBID | int | | Unique identifier for the database used by the process |
| EngineNumber | smallint | | Unique identifier of the engine on which the process is executing |
| Priority | int | | Priority at which the process is executing |
| FamilyID | int | Null | spid of the parent process, if this is a worker process |
| Login | varchar(30) | Null | Login user name |
| Application | varchar(30) | Null | Application name. May be blank if the application did not set a name in its login structure. |
| Command | varchar(30) | Null | Category of process or command the process is currently executing |
| NumChildren | int | Null | Number of child processes, if executing a parallel query |
| SecondsWaiting | int | Null | Amount of time, in seconds, the process has been waiting, if the process is currently blocked by a lock held by another process. |
| WaitEventID | int | Null | Unique identifier for the event for which the process is waiting, if the process is currently in a wait state. |
| BlockingSPID | int | Null | Session process identifier of the process holding the lock this process requested, if waiting for a lock |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| BlockingXLOID | int | Null | Unique lock identifier for the lock that this process has requested, if waiting for a lock |
| DBName | varchar(30) | Null | Name of the database the process is currently using |
| EngineGroupName | varchar(30) | Null | Engine group for the process |
| ExecutionClass | varchar(30) | Null | Execution class for the process |
| MasterTransactionID | varchar(255) | Null | Name of the transaction the process has open |
| HostName | varchar(30) | Null | Name of the host machine on which the application that started the process is running. |
| ClientName | varchar(30) | Null | Value of the *clientname* property set by the application. |
| ClientHostName | varchar(30) | Null | Value of the *clienthostname* property set by the application. |
| ClientApplName | varchar(30) | Null | Value of the *clientapplname* property set by the application. |

# monProcessActivity

Description — Provides detailed statistics about process activity.

Enable the enable monitoring and wait event timing configuration parameters for this monitoring table to collect data.

Columns — The columns for monProcessActivity are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| SPID | smallint | | Session process identifier. |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| KPID | int | | Kernel process identifier. |
| ServerUserID | int | | Server user identifier (SUID) of the user running this process. The value in ServerUserID matches the syslogins.suid column. Use the suser_name function to obtain the corresponding name. |
| CPUTime | int | Counter | CPU time (in milliseconds) used by the process. |
| WaitTime | int | Counter | Time (in milliseconds) the process spent waiting. |
| PhysicalReads | int | Counter | Number of buffers read from disk. |
| LogicalReads | int | Counter | Number of buffers read from cache. |
| PagesRead | int | Counter | Number of pages read. |
| PhysicalWrites | int | Counter | Number of buffers written to disk. |
| PagesWritten | int | Counter | Number of pages written. |
| MemUsageKB | int | | Amount of memory (in bytes) allocated to the process. |
| LocksHeld | int | | Number of locks process currently holds. |
| TableAccesses | int | Counter | Number of pages read that Adaptive Server retrieved without using an index. |
| IndexAccesses | int | Counter | Number of pages read that Adaptive Server retrieved using an index. |
| WorkTables | int | Counter | Total number of work tables the process created. |
| TempDbObjects | int | Counter | Total number of temporary tables the process created. |
| ULCBytesWritten | int | Counter | Number of bytes written to the user log cache for the process. |
| ULCFlushes | int | Counter | Total number of times the user log cache was flushed. The value is a sum of regular and tempdb user log cache. |
| ULCFlushFull | int | Counter | Number of times the user log cache was flushed because it was full. The value is a sum of regular and tempdb user log cache. |
| ULCMaxUsage | int | | The maximum usage (in bytes) of the user log cache by the process. The value is a sum of regular and tempdb user log cache. |
| ULCCurrentUsage | int | | The current usage (in bytes) of the user log cache by the process. The value is a sum of regular and tempdb user log cache. |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| Transactions | int | Counter | Number of transactions started by the process. |
| Commits | int | Counter | Number of transactions committed by the process. |
| Rollbacks | int | Counter | Number of transactions rolled back by the process. |
| HostName | varchar(30) | Null | Name of the host machine on which the application that executed the query is running. |
| Application | varchar(30) | Null | Name of the application. |
| ClientName | varchar(30) | Null | Value of the *clientname* property set by the application. |
| ClientHostName | varchar(30) | Null | Value of the *clienthostname* property set by the application. |
| ClientApplName | varchar(30) | Null | Value of the *clientapplname* property set by the application. |

# monProcessLookup

Description

Provides identifying information about each process on the server. See “monProcessActivity” on page 196 for statistics about the activity of each process.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monProcessLookup are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| SPID | smallint | | Session process identifier |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| KPID | int | | Kernel process identifier |
| Login | varchar(30) | Null | Login user name |
| Application | varchar(30) | Null | Application name |
| ClientHost | varchar(30) | Null | Host name of client |
| ClientIP | varchar(24) | Null | IP address of client |
| ClientOSPID | varchar(30) | Null | Client application’s operating system process identifier |
| ClientName | varchar(30) | Null | Value of the *clientname* property set by the application |
| ClientHostName | varchar(30) | Null | Value of the *clienthostname* property set by the application |
| ClientApplName | varchar(30) | Null | Value of the *clientapplname* property set by the application |

Use the set command to configure *clientname*, *clienthostname*, *clientapplname*. See the *Reference Manual: Commands*.

# monProcessMigration

Description — (Cluster environments only) Displays information about the connection currently migrating.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns — The columns for monProcessMigration are:

| Name | Datatype | Description |
|---|---|---|
| SPID | int4 | Pending migration session process ID |
| KPID | int4 | Kernel process ID |
| LogicalCluster | varchar(30) | Current logical cluster |
| Instance | varchar(30) | Current instance. |
| MigrationLogicalCluster | varchar(30) | Migration logical cluster. |
| MigrationInstance | varchar(30) | Migration instance. |
| Command | varchar(30) | Migration trigger. |

# monProcessNetIO

Description — Provides the network I/O activity information for each process.

Enable the enable monitoring configuration parameter for this monitoring table to collect data.

Columns — The columns for monProcessNetIO are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| SPID | smallint | | Session process identifier |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| KPID | int | | Kernel process identifier |
| NetworkPacketSize | int | | Network packet size the session is currently using. |
| PacketSent | int | Counter | Number of packets sent |
| PacketsReceived | int | Counter | Number of packets received |
| BytesSent | int | Counter | Number of bytes sent |
| BytesRecieved | int | Counter | Number of bytes received |
| NetworkEngineNumber | smallint | | Number of the engine that this process is using as its network engine. |

# monProcessObject

Description

Provides statistical information regarding objects currently being accessed by processes.

Enable the enable monitoring and per object statistics active configuration parameters for this monitoring table to collect data.

Columns

The columns for monProcessObject are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| SPID | smallint | | Session process identifier |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| KPID | int | | Kernel process identifier |
| DBID | int | | Unique identifier for the database in which the object resides |
| ObjectID | int | | Unique identifier for the object |
| PartitionID | int | | Unique identifier for the partition |
| IndexID | int | | Unique identifier for the index |
| OwnerUserID | int | | User identifier for the object owner |
| LogicalReads | int | Counter | Number of buffers read from cache |
| PhysicalReads | int | Counter | Number of buffers read from disk |
| PhysicalAPFReads | int | Counter | Number of asynchronous prefetch buffers read from disk |
| DBName | varchar(30) | Null | Name of database |
| ObjectName | varchar(30) | Null | Name of the object |
| PartitionName | varchar(30) | Null | Name of the partition |
| ObjectType | varchar(30) | Null | Type of object |
| PartitionSize | int | Counter, null | Partition size in kilobytes |

# monProcessProcedures

Description — Returns a list of all procedures being executed by processes.

Enable the enable monitoring and statement statistics active configuration parameters for this monitoring table to collect data.

Columns — The columns for monProcessProcedures are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| SPID | smallint | | Session process identifier |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| KPID | int | | Kernel process identifier |
| DBID | int | | Unique identifier for object's database |
| OwnerUID | int | | Unique identifier for the object owner |
| ObjectID | int | | Unique identifier for the procedure |
| PlanID | int | | Unique identifier for the query plan |
| MemUsageKB | int | | Amount of memory, in KB, used by the procedure |
| CompileDate | datetime | | Date that the procedure was compiled |
| ContextID | int | | A unique identifier generated each time an executing query causes a stored procedure, trigger, execute immediate, deferred compilation, or other compiled object execution to occur |
| LineNumber | int | | The line in the procedure currently being executed |
| StmtNumber | int | | The currently executing statement |
| DBName | varchar(30) | Null | Name of the database that contains the procedure |
| OwnerName | varchar(30) | Null | Name of the owner of the object |
| ObjectName | varchar(30) | Null | Name of the procedure |
| ObjectType | varchar(32) | Null | The type of procedure (for example, stored procedure or trigger) |
| ExecutionCount | int | Counter | Number of times Adaptive Server executed this instance of the stored procedure held in the procedure cache |
| CPUTime | int | Counter | Amount of CPU time, in milliseconds, Adaptive Server spent executing the instance of this stored procedure held in the procedure cache |
| ExecutionTime | int | Counter | Total amount of time, in milliseconds, Adaptive Server spent executing the instance of this stored procedure held in the procedure cache |
| PhysicalReads | int | Counter | Number of physical reads performed by the instance of this stored procedure held in the procedure cache |
| LogicalReads | int | Counter | Number of logical reads performed by the instance of this stored procedure held in the procedure cache |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| PhysicalWrites | int | Counter | Number of physical writes performed by the instance of this stored procedure held in the procedure cache |
| PagesWritten | int | Counter | Number of pages read by the instance of this stored procedure held in the procedure cache |

# monProcessSQLText

Description

Provides the SQL text currently being executed by the process. Use max SQL text monitored to tune the maximum size of the SQL text.

monProcessSQLText returns a row for each row of the SQL text batch a process executes (specified by SPID). That is, if a batch contains three rows, monProcessSQLText returns three rows in its result set. The value for LineNumber indicates the number of the line in the batch. If the length of a single row exceeds 255 bytes, monProcessSQLText returns multiple rows and the value for LineNumber is the same for all rows, but the value for SequenceInLine is different for each row.

Enable the enable monitoring, max SQL text monitored, SQL batch capture configuration parameter for this monitoring table to collect data.

Columns

The columns for monProcessSQLText are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| SPID | smallint | | Session process identifier. |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| KPID | int | | Kernel process identifier. |
| ServerUserID | int | | Server user identifier (SUID) of the user executing this SQL. The ServerUserID matches the value for the syslogins.suid column. Use the suser_name function to obtain the corresponding name. |
| BatchID | int | | Unique identifier for the SQL batch containing the SQL text. |
| LineNumber | int | | SQL batch line number for the row's SQL text. |
| SequenceInLine | int | | Each row has a unique, and increasing, SequenceInLine value. If the length of the SQL text exceeds 255 bytes, the text is split over multiple rows. |
| SQLText | varchar(255) | Null | The text being executed. |

# monProcessStatement

Description

Provides information about the statement currently executing.

Enable the enable monitoring, statement statistics active, per object statistics active, and wait event timing configuration parameters for this monitoring table to collect data.

Columns

The columns for monProcessStatement are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| SPID | smallint | | Session process identifier. |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| KPID | int | | Kernel process identifier. |
| DBID | int | | Unique identifier for the database currently being used by the process. |
| ProcedureID | int | | Unique identifier for the stored procedure. |
| PlanID | int | | Unique identifier for the plan the process is executing. |
| BatchID | int | | The batch number for the process in which the statement is executed. |
| ContextID | int | | The stack frame of the procedure, if a procedure. |
| LineNumber | int | | Line number of the statement within the SQL batch. |
| CPUTime | int | Counter | CPU time, in milliseconds, used by the statement. |
| WaitTime | int | Counter | Amount of time, in milliseconds, the task has waited while the statement executes. |
| MemUsageKB | int | | Number of kilobytes of memory used for execution of the statement. |
| PhysicalReads | int | Counter | Number of buffers read from disk. |
| LogicalReads | int | Counter | Number of buffers read from cache. |
| PagesModified | int | Counter | Number of pages modified by the statement. |
| PacketsSent | int | Counter | Number of network packets sent by Adaptive Server. |
| PacketsReceived | int | Counter | Number of network packets received by Adaptive Server. |
| NetworkPacketSize | int | | Size, in bytes, of the network packet currently configured for the session. |
| PlansAltered | int | Counter | Number of plans altered at execution time. |
| RowsAffected | int | | Number of rows affected by the current statement. Queries using an inefficient query plan likely show a high number of logical I/Os per returned row. |
| DBName | varchar(30) | | Name of the database in which this process is executing. If the process is executing a stored procedure or other compiled object, the database name is the name of the database for that object. |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| StartTime | datetime | Null | Date when the statement began executing. |

# monProcessWaits

Description

Provides a list of all wait events for which current processes on the server are waiting. Returns only wait events whose Waits value is greater than zero.

Enable the enable monitoring, wait event timing, and process wait events, configuration parameters for this monitoring table to collect data.

Columns

The columns for monProcessWaits are:

| Name | Datatype | Attribute | Description |
|---|---|---|---|
| SPID | smallint | | Session process identifier |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| KPID | int | | Kernel process identifier |
| ServerUserID | int | | Server user ID (SUID) of the user associated with this process. |
| WaitEventID | smallint | | Unique identifier for the wait event |
| Waits | int | Counter | Number of times the process has waited for the event |
| WaitTime | int | Counter | Amount of time, in milliseconds, that the process has waited for the event |

WaitEventInfo contains descriptions of each wait event. Join the WaitEventID column from each monitor table to view this data.

See *Performance and Tuning: Monitoring Tables* for a descriptions of select wait events.

# monProcessWorkerThread

Description

Provides statistics for the activity of each currently configured worker process.

Enable the enable monitoring configuration parameter for this monitoring table to collect data.

Columns

The columns for monProcessWorkerThread are:

| Name | Datatype | Attribute | Description |
|---|---|---|---|
| SPID | smallint | | Session process identifier |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| KPID | int | | Kernel process identifier |
| ThreadsActive | int | | Number of worker threads currently in use by the process |
| MaxParallelDegree | smallint | | The maximum degree of parallelism this task can use, which is set with the set parallel_degree option for the session, or the current Run Value for max parallel degree. |
| MaxScanParallelDegree | smallint | | The maximum degree of parallelism for scans this task can use, which is set with set scan_parallel_degree for the session, or if this is not set, the current Run Value for max scan parallel degree. |
| ParallelQueries | int | Counter | Total number of parallel queries performed by this process |
| PlansAltered | int | Counter | Number of plans altered from “optimal” for the process. Plans are altered if Adaptive Server has an insufficient number of worker threads available to execute the query with an optimal degree of parallelism. |
| FamilyID | int | Null | The spid of the parent process, if this is a worker process |

# monRepLogActivity

Description — Collects information from monitor counters updated by Replication Agent

You need not enable any configuration parameters for this monitoring table to collect data.

Columns — The columns for monRepLogActivity are:

| Name | Datatype | Description |
|---|---|---|
| DBID | int | Unique identifier for the database currently being used by the process |
| SPID | int | Session process identifier |
| InstanceID | tinyint | (Cluster environments only) ID of an instance in a shared-disk cluster |
| LogRecordsScanned | int | Total number of log records scanned |
| LogRecordsProcessed | int | Total number of log records processed |
| NumberOfScans | int | Total number of scans performed |
| TotalTimeForLogScans | bigint | Total amount of time the scanner thread used to scan the log |
| LongestTimeForLogScans | bigint | Longest time spent on a single scan |
| AvgTimeForLogScans | bigint | Average amount of time spent on the log scan |
| Updates | int | Total number of updates processed |
| Inserts | int | Total number of inserts processed |
| Deletes | int | Total number of deletes processed |
| StoredProcedures | int | Total number of stored procedures processed |
| SQLStatements | int | Total number of SQL statements processed |
| DDL | int | Total number of DDL log records processed |
| Writetext | int | Total number of Log records processed by writetext commands |
| LobColumns | int | Total number of DML log records processed for a table with off- ow, large object columns |
| CLRs | int | Total number of CLRs processed |
| Checkpoints | int | Total number of checkpoints processed |
| BeginTransaction | int | Total number of begin transactions processed |
| CommitTransaction | int | Total number of commit transactions processed |
| AbortedTransaction | int | Total number of aborted transactions processed |
| PreparedTransaction | int | Total number of transactions found in the prepare state |
| DelayedCommit | int | Total number of delayed commits processed |
| MaintenanceUserTransaction | int | Total number of transactions opened by the maintenance user |
| NumberOfLogExtentions | int | Total number of times the RepAgent waited for extensions to transactions |

| Name | Datatype | Description |
|---|---|---|
| TotalTimeOfLogExtentions | bigint | Total amount of time, in milliseconds, the RepAgent waited for log extensions |
| LongestTimeOfLogExtentions | bigint | Longest amount of time, in milliseconds, the RepAgent waited for log extensions |
| AvgTimeOfLogExtentions | bigint | Average amount of time, in milliseconds, the RepAgent waited for log extensions |
| MaxHashSchemaSize | int | Maximum size of the hash schema cache |
| NumberOfSchemasReused | int | Total number of schemas reused |
| NumberOfSchemaFwdLookup | int | Total number of schema forward lookups |
| TotalTimeOfSchemaFwdLookup | bigint | Total amount of time, in milliseconds, spent on forward scans |
| LongestTimeOfSchemaFwdLookup | bigint | Longest amount of time, in milliseconds, spent on a forward scan |
| AvgTimeOfSchemaFwdLookup | bigint | Average amount of time, in milliseconds, spent on forward scans |
| NumberOfSchemaBckwLookup | int | Total number of schema backward lookups |
| TotalTimeOfSchemaBckwLookup | bigint | Total amount of time spent on schema backward lookups |
| LongestTimeOfSchemaBckwLookup | bigint | The longest amount of time, in milliseconds, spent on a backward scan. |
| AvgTimeOfSchemaBckwLookup | bigint | Average amount of time, in milliseconds, spent on backward scans |
| NumberOfMempoolAllocates | int | Total number of RepAgent pool allocates |
| NumberOfMempoolFrees | int | Total number of RepAgent memory pool frees |
| MempoolCurrentSize | int | Current size of the RepAgent memory pool |
| MempoolHighUsage | int | RepAgent memory pool high usage |
| DBName | varchar(30) | Name of the database in which the task scans |

# monRepScanners

Description — Provides information on where the Rep Agent Scanner task is spending its time

You need not enable any configuration parameters for this monitoring table to collect data.

Columns — The columns for monRepScanners are:

| Description | Datatype | Description |
|---|---|---|
| DBID | int | Unique identifier for the database currently being used by the process. |
| SPID | int | Session process identifier |
| InstanceID | tinyint | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| EngineBinding | int | Number of the engine with which this task is bound (not applicable to threaded mode) |
| LogRecordsScanned | int | Total number of log records scanned |
| LogrecordsProcessed | int | Total number of log records processed |
| NumberOfTruncPointRequested | int | Total number of times RepAgent asked Replication Server for a new truncation point |
| NumberOfTruncPointMoved | int | Total number of times RepAgent moved the secondary truncation point |
| DBName | varchar(30) | Name of the database in which this task scans |
| Status | varchar(30) | Current task status |
| SleepStatus | varchar(30) | Current sleep status, if sleeping |
| StartMarker | varchar(30) | Start marker in the log for this scanner |
| EndMarker | varchar(30) | End marker in the log for this scanner |
| CurrentMarker | varchar(30) | Current marker in the log for this scanner |
| OldestTransaction | varchar(30) | Oldest open transaction |

# monRepScannersTotalTime

Description — Provides information on where the Rep Agent Scanner task is spending its time

You need not enable any configuration parameters for this monitoring table to collect data.

Columns — The tables for monRepScannersTotalTime are:

| Name | Datatype | Description |
|---|---|---|
| DBID | int | Unique identifier for the database currently being used by the process. |
| SPID | int | Session process identifier |
| InstanceID | tinyint | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| LogRecProcessed | bigint | Total number of log records processed by the scanner thread |
| BytesPacked | bigint | Amount of bytes packed by the scanner thread |
| TotalTime | bigint | Total amount of time used by the scanner thread |
| MRPBootstrapTime | bigint | Total amount of time required, in microseconds, to complete the multipath replication bootstrap cycle |
| ScanTime | bigint | Total amount of time spent scanning |
| ProcessTime | bigint | Total amount of time spent processing log records |
| SchemaLookupsTime | bigint | Total amount of time spent looking for an object's schema in RepAgent cache |
| PackTime | bigint | Total amount of time spent packing the LTL |
| QueueingTime | bigint | Total amount of time spent queueing LTL packets |
| HashBindingSize | bigint | Total number of buckets in the hash binding table holding an object's binding information |
| HashBindingEntries | bigint | Total number of objects bound to a path when RepAgent was boot strapped |
| HashBindingCollisions | bigint | The length of the longest collision chain used in the hash binding table |
| YieldsOnFullQueue | bigint | Total number of scanner yields on a full queue |
| WaitsOnSenderThread | bigint | Total number of waits on a sender thread |
| WaitTimeOnSenderThread | bigint | Total amount of time, in milliseconds, spent waiting on the sender thread |
| LongestWaitOnSenderThread | bigint | Longest amount of time, in milliseconds, spent waiting on the sender thread |

# monRepSenders

Description

Provides processing information about Rep Agent Sender tasks.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monRepSenders are:

| Name | Datatype | Description |
|---|---|---|
| DBID | int | Unique identifier for the database currently being used by the process. |
| SPID | int | Session process identifier |
| InstanceID | tinyint | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| EngineBinding | int | Number of the engine with which this task is bound (not applicable to threaded mode) |
| MessageQueueSize | int | Maximum size of the message queue |
| MessagesInQueue | int | Total number of messages in the message queue |
| NumberOfScannerYields | int | Total number of times the scanner yielded on a full queue |
| NumberOfScannerSleeps | int | Total number of times the scanner slept on a full queue |
| NumberOfBytesSent | int | Total number of bytes sent |
| LastRepServerError | int | Last error from Replication Server |
| NumberOfRetries | int | Total number of connection retries |
| SleepsOnEmptyQueue | int | Total number of sleeps spent on an empty message queue |
| NumberOfQueueFlushes | int | Total number of times a sender flushed its queue |
| SleepTimeOnEmptyQueue | int | Total amount of time, in milliseconds, spent sleeping in an empty queue |
| LongestSleepTimeOnEmptyQueue | int | Longest amount of time, in milliseconds, spent sleeping on an empty queue |
| MaxQueueSize | int | Maximum queue size ever reached |
| DBName | varchar(30) | Name of the database in which the task scans |
| Dataserver | varchar(30) | Dataserver name used to connect to Replication Server |
| ReplicationServer | varchar(30) | Replication Server name used to connect to Replication Server |
| Username | varchar(30) | User name used to connect to Replication Server |
| Status | varchar(30) | Current status of this task |
| SleepStatus | varchar(30) | Current sleep status, if sleeping |

# monSpinlockActivity

Description — Provides statistics about spinlock activity.

Enable the enable spinlock monitoring configuration parameter for this monitoring table to collect data.

Columns — The columns for monSpinlockActivity are:

| Name | Datatype | Description |
|---|---|---|
| SpinlockName | varchar(255) | Name of spinlock |
| Grabs | bigint | Number of grabs for this spinlock |
| Spins | bigint | Number of spins on this spinlock |
| Waits | bigint | Number of waits for this spinlock |
| OwnerPID | int | Current owner Process Identifier |
| LastOwnerPID | int | Previous owner Process Identifier |
| Contention | real | Spinlock contention, as percentage |
| InstanceID | tinyint | (Cluster environments only) ID of an instance in a shared-disk cluster. |

# monSQLRepActivity

Description

Provides statistics for SQL statements that were successfully replicated on all open objects.

Enable the enable monitoring and per object statistics active configuration parameters for this monitoring table to collect data.

Columns

The columns for monSQLRepActivity are:

| Name | Datatype | Description |
|---|---|---|
| DBID | int | Unique identifier of the database the process is currently using |
| ObjectID | int | ID of the object being monitored |
| InstanceID | tinyint | (Cluster environments only) ID of an instance in a shared-disk cluster |
| DBName | varchar(30) | Name of database containing the object being monitored for activity |
| ObjectName | varchar(30) | Name of the object being monitored for activity |
| UpdateStmts | int | Number of update statements replicated as SQL |
| InsertSelectStmts | int | Number of insert and select statements replicated as SQL |
| DeleteStmts | int | Number of delete statements replicated as SQL |
| SelectIntoStmts | int | Number of select into statements replicated as SQL |
| RowsThreshold | int | Low boundary range for the number of rows affected by the statements |

# monSQLRepMisses

Description — Provides statistics for SQL statements that were not successfully replicated for all open objects.

Enable the enable monitoring and per object statistics active configuration parameter for this monitoring table to collect data.

Columns — The columns for monSQLRepMisses are:

| Name | Datatype | Description |
|---|---|---|
| DBID | int | Unique identifier of the database the process is currently using |
| ObjectID | int | ID of the object being monitored |
| InstanceID | tinyint | (Cluster environments only) ID of an instance in a shared-disk cluster |
| DBName | varchar(30) | Name of database containing the object being monitored for activity |
| ObjectName | varchar(30) | Name of the object being monitored for activity |
| Threshold | int | Number of statements that could not be replicated as SQL because the number of affected rows was below the defined threshold |
| QueryLimitation | int | Number of statements that could not be replicated as SQL because of a query limitation |
| Configuration | int | Number of statements that could not be replicated as SQL because of the configuration |

# monState

Description — Provides information regarding the overall state of Adaptive Server.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns — The columns for monState are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| LockWaitThreshold | int | | Time (in seconds) that a process must wait for a lock before it is counted as blocked and reported in the LockWaits column. The default value for LockWaitThreshold is 5 seconds. The default is used if you do not specify a value in the where clause of the query (for example `LockWaitThreshold=30`). |
| LockWaits | int | | Number of process that have waiting for a lock longer than the value of LockWaitThreshold. |
| DaysRunning | int | | Number of days Adaptive Server has been running. |
| CheckPoints | int | | Specifies if any checkpoint is currently running. |
| NumDeadlocks | int | Counter | Total number of deadlocks that have occurred. |
| Diagnostic Dumps | int | | Specifies if a shared memory dump is currently in progress for this server. |
| Connections | int | | Number of active inbound connections. |
| MaxRecovery | int | | The maximum time (in minutes), per database, that Adaptive Server uses to complete its recovery procedures in case of a system failure; also, the current Run Value for the recovery interval in minutes configuration option. |
| Transactions | int4 | | Number of transactions run, server-wide. |
| StartDate | datetime | | Date and time Adaptive Server was started. |
| CountersCleared | datetime | | Date and time the monitor counters were last cleared. |

# monStatementCache

Description

Provides statistical information about the statement cache. You must enable the statement cache before monStatementCache table can collect data.

Enable the enable monitoring, enable stmt cache monitoring, and statement cache size configuration parameters for this monitoring table to collect data.

Columns

The columns for monStatementCache are:

| Name | Type | Attributes | Description |
|---|---|---|---|
| InstanceID | tinyint | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| TotalSizeKB | int | | Configured size, in KB, of the statement cache. |
| UsedSizeKB | int | | Amount of the statement cache, in KB, currently in use. |
| NumStatements | int | | Number of statements in the statement cache. |
| NumSearches | int | Counter, reset | Number of times the statement cache was searched. |
| HitCount | int | Counter, reset | Number of times the statement cache was searched and a match was found. |
| NumInserts | int | Counter, reset | Number of statements that were inserted into the statement cache. |
| NumRemovals | int | Counter, reset | Number of times statements were removed from the statement cache. This value includes statements that were removed with explicit purges or from a replacement strategy. |
| NumRecompilesSchemaChanges | int | Counter, reset | Number of recompiles due to schema changes in the tables referred to in the cached statements. |
| NumRecompilesPlanFlushes | int | Counter, reset | Number of recompiles due to the plan flushes from the cache. |

# monSysLoad

Description

(Cluster environments only) Provides trended statistics on a per-engine basis. You need not have the mon_role role to query this monitor table.

There is one row per engine per statistic, with the exception of kernel run queue length, which is reported only for engine number 0.

Averages are computed using an algorithm that eliminates momentary peaks and valleys and provides a an indication of overall trends.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monSysLoad are:

| Name | Datatype | Description |
|---|---|---|
| InstanceID | tinyint | ID of the instance within the cluster. |
| EngineNumber | smallint | Engine to which this row belongs. |
| SteadyState | real | Average value for this statistic since Adaptive Server started. |
| Avg_1min | real | One-minute moving average for this statistic. |
| Avg_5min | real | Five-minute moving average for this statistic. |
| Avg_15min | real | Fifteen-minute moving average for this statistic. |
| Max_1min | real | Maximum 1-minute average since start-up. |
| Max_5min | real | Maximum 5-minute average since start-up. |
| Max_15min | real | Maximum 15-minute average since start-up. |
| Max_1min_Time | datetime | *datetime* at which Max_1min occurred. |
| Max_5min_Time | datetime | *datetime* at which Max_5min occurred. |
| Max_15min_Time | datetime | *datetime* at which Max_15min occurred. |
| Statistic | | Name of the statistic this row represents:<br>• Percent CPU busy<br>• Percent I/O busy<br>• Run queue length<br>• Kernel run queue length<br>• Outstanding disk I/Os<br>• Disk I/Os per second<br>• Network I/Os per second |
| Sample | float | Value of the metric at the last sample interval (that is, the current value of the metric). |
| Peak | float | The highest Sample value since the instance started (that is, the peak Sample value). |
| Peak_time | datetime | The date and time the Peak value was achieved. |
| StatisticID | int | A fixed identifier for this statistic. You may want to write applications to the fixed StatisticID instead of the localized Statistic name. |

# monSysPlanText

Description

Provides the history of the query plans for recently executed queries. monSysPlanText returns one row of text from each line of the running query plans (similar to what is returned sp_showplan or by set showplan on). To make sure monSysPlanText reads the query plan text in the correct sequence, order the query result by SequenceNumber. For queries returning data for multiple queries or processes, order the query result by SPID, KPID, BatchID, SequenceNumber.

Enable the enable monitoring, plan text pipe max messages, and plan text pipe active configuration parameters for this monitoring table to collect data.

Columns

The columns for monSysPlanText are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| PlanID | int | | Unique identifier for the plan. |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| SPID | smallint | | Session process identifier. |
| KPID | int | | Kernel process identifier. |
| BatchID | int | | Unique identifier for the SQL batch for which the plan was created. |
| ContextID | int | | The stack frame of the procedure, if a procedure. |
| SequenceNumber | int | | A monotonically increasing number indicating the position of the PlanText column within the entire plan text. |
| DBID | int | | Unique identifier for the database where the procedure is stored, if the plan is for a stored procedure. |
| ProcedureID | int | | Unique identifier for the procedure, if the plan is for a stored procedure. |
| DBName | varchar(30) | Null | Name of the database in which the statement represented by this plan is executed. This column is NULL if this database is not open when monSysPlanText is queried. If the process is executing a stored procedure or other compiled object, the database name is the name of the database for that object. |
| PlanText | varchar(160) | Null | Plan text output. |

Typically, there are multiple rows in this table for each query plan. Arrange the rows by sorting on the SequenceNumber column in ascending order.

monSysPlanText is a historical monitoring table. See “Stateful historical monitoring table” in Chapter 1, “Introduction to Monitoring Tables” in the *Performance and Tuning Guide*.

# monSysSQLText

Description

Provides the most recently executed SQL text, or the SQL text currently executing. The maximum number of rows returned can be tuned with sql text pipe max messages.

Enable the enable monitoring, SQL batch capture, sql text pipe max messages, sql text pipe active configuration parameters for this monitoring table to collect data.

monSysSQLText is a historical monitoring table. See *Performance and Tuning: Monitoring Tables*.

Columns

The columns for monSysSQLText are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| SPID | smallint | | Session process identifier. |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| KPID | int | | Kernel process identifier. |
| ServerUserID | int | | Server user identifier (SUID) of the user who executed this SQL text. The ServerUserID matches the value in syslogins.suid. Use the suser_name function to obtain the corresponding name. |
| BatchID | int | | Unique identifier for the SQL batch containing the SQL text. |
| SequenceInBatch | int | | Indicates the position of this portion of SQL text within a batch (the SQL text for a batch may span multiple rows). |
| SQLText | varchar(255) | Null | SQL text. |

**Note** In many cases the text for a query spans multiple rows in this table. Arrange rows in proper order by sorting on the SequenceInBatch column in ascending order.

# monSysStatement

Description

Provides a history of the most recently executed statements on the server. Use statement pipe max messages to tune the maximum number of statement statistics returned.

Enable the enable monitoring, statement statistics active, per object statistics active, statement pipe max messages, and statement pipe active configuration parameters for this monitoring table to collect data.

monSysStatement is a historical monitoring table. See *Performance and Tuning: Monitoring Tables*.

Columns

The columns for monSysStatements are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| SPID | smallint | | Session process identifier. |
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| KPID | int | | Kernel process identifier. |
| DBID | int | | Unique identifier for the database. |
| ProcedureID | int | | Unique identifier for the procedure. |
| PlanID | int | | Unique identifier for the stored plan for the procedure. |
| BatchID | int | | Unique identifier for the SQL batch containing the statement. |
| ContextID | int | | The stack frame of the procedure, if a procedure. |
| LineNumber | int | | Line number of the statement within the SQL batch. |
| CpuTime | int | Counter | Number of milliseconds of CPU used by the statement. |
| WaitTime | int | Counter | Number of milliseconds the task has waited during execution of the statement. |
| MemUsageKB | int | | Number of kilobytes of memory used for execution of the statement. |
| PhysicalReads | int | Counter | Number of buffers read from disk. |
| LogicalReads | int | Counter | Number of buffers read from cache. |
| PagesModified | int | Counter | Number of pages modified by the statement. |
| PacketsSent | int | Counter | Number of network packets sent by Adaptive Server. |
| PacketsReceived | int | Counter | Number of network packets received by Adaptive Server. |
| NetworkPacketSize | int | | Size (in bytes) of the network packet currently configured for the session. |
| PlansAltered | int | Counter | The number of plans altered at execution time. |
| RowsAffected | int | | Number of rows affected by the current statement. Queries using an inefficient query plan likely show a high number of logical I/Os per returned row. |

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| ErrorStatus | int | | The error return status of the statement. |
| HashKey | int | | Hash value for the text of the statement; this is not a unique identifier. This column is zero (0) if the statement is not executed from the statement cache. |
| SsqlId | int | | ID of the query plan for this statement within the statement cache. This column is zero (0) if the statement is not executed from the statement cache. |
| ProcNestLevel | int | | Nesting level of the statement. This column is zero (0) if the statement is an ad hoc query. If the statement is within a stored procedure, this column indicates the nesting level of that stored procedure. |
| StatementNumber | int | | Number indicating the order in which this statement was executed within the SQL batch for the process. |
| DBName | varchar(30) | | Name of the database inwhich the statement is executed. This column is NULL if the database is no longer open when monSysStatement is queried. If the process is executing a stored procedure or other compiled object, the database name is the name of the database for that object. |
| StartTime | datetime | Null | Date the statement began execution. |
| EndTime | datetime | Null | Date the statement finished execution. |

# monSysWaits

Description

Provides a server-wide view of the statistics for events on which processes have waited.

Enable the enable monitoring and wait event timing configuration parameters for this monitoring table to collect data.

Columns

The columns for monSysWaits are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| WaitEventID | smallint | | Unique identifier for the wait event |
| WaitTime | int | Counter | Amount of time (in seconds) tasks spent waiting for the event |
| Waits | int | | Number of times tasks waited for the event |

See *Performance and Tuning: Monitoring Tables* for more information

You can join the monSysWaits table with monWaitEventInfo using the WaitEventID columns as the join column to obtain the wait event descriptions. For example:

```
select w.Waits, w.WaitTime, w.WaitEventID, i.Description
from master..monSysWaits w, master..monWaitEventInfo i
where w.WaitEventID = i.WaitEventID
```

# monSysWorkerThread

Description

Returns server-wide statistics related to worker thread configuration and execution.

Enable the enable monitoring configuration parameter for this monitoring table to collect data.

Columns

The columns for monSysWorkerThread are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| InstanceID | int | | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| ThreadsActive | int | | Number of worker processes currently active |
| TotalWorkerThreads | int | | Maximum number of worker processes (configured by setting number of worker processes) |
| HighWater | int | reset | The maximum number of worker processes that have ever been in use |
| ParallelQueries | int | Counter, reset | Number of parallel queries attempted |
| PlansAltered | int | Counter, reset | Number of plans altered due to unavailable worker processes |
| WorkerMemory | int | | The amount of memory currently in use by worker processes |
| TotalWorkerMemory | int | | The amount of memory configured for use by worker processes |
| WorkerMemoryHWM | int | reset | The maximum amount of memory ever used by worker processes |
| MaxParallelDegree | int | | The maximum degree of parallelism that can be used: the current Run Value for max parallel degree configuration option |
| MaxScanParallelDegree | int | | The maximum degree of parallelism that can be used for a scan: the current Run Value for max scan parallel degree configuration option |

# monTableColumns

Description

Describes all the columns for each monitoring table. monTableColumns helps determine what columns are in the monitoring tables. You can join monTableColumns with monTables to report columns and column attributes for the monitoring tables.

The metadata view for this table is identical for all instances in a shared-disk cluster.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monTableColumns are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| TableID | int | | Unique identifier for the view |
| ColumnID | int | | Position of the column |
| TypeID | int | | Identifier for the datatype of the column |
| Precision | tinyint | | Precision of the column, if numeric |
| Scale | tinyint | | Scale of the column, if numeric |
| Length | smallint | | Maximum length of the column (in bytes) |
| Indicators | int | | Indicators for specific column properties (for example, if the column is prone to wrapping and should be sampled)[1] |
| TableName | varchar(30) | Null | Name of the table. |
| ColumnName | varchar(30) | Null | Name of the column. |
| TypeName | varchar(20) | Null | Name of the datatype of the column. |
| Description | varchar(255) | Null | Description of the column (includes the column's unit of measurement). |
| Language | varchar(30) | | Allows you to specify the language in which Adaptive Server returns the values of the Description column and the Label column.<br>By default, Adaptive Server returns US English. Queries must use the the IS0-639 and ISO-3166 naming conventions. |
| Label | varchar(50) | | Description of the data presented in the column. You can use these values in application user interfaces instead of the actual column names. |

The Indicators column is a bitmap. Use a bit mask to to determine which bits are turned on. Possible values are:

- 1 – the value for Indicators may increase rapidly and lead to counter wrapping if values reach $2^{32}$, which can occur in columns that have the number 1 bit in the Indicators column value turned on. To determine whether the 1 bit is turned on, use:

  ```
  select TableName, ColumnName
  from Master..monTableColumns
  where Indicators & 1 != 0
  ```

- 2 – the counter is shared with sp_sysmon and is reset if you execute sp_sysmon. . .clear.

  To display all columns sp_sysmon clears with the clear parameter, use:

  ```
  Select TableName, ColumnName
  from master..monTableColumns
  where Indicators & 2 != 0
  ```

# monTableCompression

Description — Contains the table's compression history. Enable the enable monitoring and per object statistics active configuration parameters for this monitoring table to collect data.

Columns — The columns for monTableCompression are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| InstanceID | tinyint | | (Cluster Edition only) Server instance ID |
| DBID | int | | ID of the database to which this table was transferred |
| ObjectID | int | | ID of the compressed object |
| PartitionID | int | | ID of the compressed partition |
| CompRowInserted | bigint | Counter | Number of compressed rows inserted |
| CompRowUpdated | bigint | Counter | Number of updated compressed rows |
| CompRowForward | bigint | Counter | Number of compressed rows forwarded from the update |
| CompRowScan | bigint | Counter | Number of compressed rows accessed |
| RowDecompressed | bigint | Counter | Number of rows decompressed |
| RowPageDecompressed | bigint | Counter | Number of page-compressed rows decompressed to be row-compressed |
| ColDecompressed | bigint | Counter | Number of columns decompressed |
| RowCompNoneed | int | Counter | Number of rows not compressed because their compressed row length exceeded their normal row length |
| PageCompNoneed | bigint | Counter | Number of pages that are not suitable for page-level compression because Adaptive Server cannot generate a dictionary or index |
| PagesCompressed | bigint | Counter | Number of pages compressed at the page-level |
| AvgBytesSavedPageLevel | bigint | Counter | Number of bytes page level compression saved |
| TableName | varchar | NULL | Name of the compressed table |

# monTableParameters

Description

Provides a description for all columns in a monitoring table used to optimize query performance for the monitoring tables.

The metadata view for this table is identical for all instances in a shared-disk cluster.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monTableParameters are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| TableID | int | | Unique identifier for the table |
| ParameterID | int | | Position of the parameter |
| TypeID | int | | Identifier of the datatype of the parameter |
| Precision | tiny_int | | Precision of the parameter, if numeric |
| Scale | tiny_int | | Scale of the parameter, if numeric |
| Length | small_int | | Maximum length of the parameter (in bytes) |
| TableName | varchar(30) | Null | Name of the table |
| ParameterName | varchar(30) | Null | Name of the parameter |
| TypeName | varchar(20) | Null | Name of the datatype of the parameter |
| Description | varchar(255) | Null | Description of the parameter |

# monTables

Description

Provides a description of all monitoring tables. You can join monTables with monTableColumns for a description of each monitoring table and the columns it contains.

The metadata view for this table is identical for all instances in a shared-disk cluster.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monTables are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| TableID | int | | Unique identifier for the table |
| Columns | tinyint | | Total number of columns in the table |
| Parameters | tinyint | | Total number of optional parameters you can specify |
| Indicators | int | | Indicators for specific table properties (for example, if the table retains session context)<br>The Indicators column is a bit map. Use a bitmask to to determine which bits are turned on. A value of 1 indicates the table is a historical table.<br>To display all tables that are historical:<br>`Select TableName`<br>`from master..monTables`<br>`where Indicators & 1 != 0` |
| Size | int | | Maximum row size (in bytes) |
| TableName | varchar(30) | Null | Table name |
| Description | varchar(368) | Null | Table description |
| Language | varchar(30) | | Allows you to specify the language in which Adaptive Server returns the values of the Description column.<br>By default, Adaptive Server returns US English. Queries must use the the IS0-639 and ISO-3166 naming conventions. |

# monTableTransfer

Description

MonTableTransfer provides historical transfer information for tables in Adaptive Server's active memory. It does not store information for completed transfers. MonTableTransfer provides transfer information on currently ongoing transfers of all tables, whether they are marked for incremental transfer or not, and on previous transfers on tables marked for incremental transfer.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monTableTransfer are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| InstanceID | tinyint | NOT NULL | (Cluster environments only) Holds the instance ID of the server in which the command is running. In non-clustered servers, always holds zero. |
| DBID | smallint | | Database ID of table |
| TableID | int | | Unique identifier of table |
| TableName | varchar(255) | NULL | Name of table |
| SequenceID | int | | Adaptive Server-generated internal tracking ID |
| TrackingID | int | NULL | User-specified tracking ID |
| PercentDone | smallint | | Percentage of transfer work done, expressed as an integer between 0 – 100 (all completed transfers show 100) |
| BeginTime | datetime | | Date and time at which transfer begins |
| EndTime | datetime | NULL | Date and time at which transfer ends. Ongoing transfers show NULL. |
| EndCode | smallint | NULL | Ending status of transfer.<br>• 0 – successful transfer.<br>• NULL – ongoing transfer.<br>• Error code – failed transfer. |
| TransferFloor | bigint | | Timestamp at which data can be sent |
| TransferCeiling | bigint | | Timestamp at which data is uncommitted and cannot be sent |
| RowsSent | bigint | | Number of rows sent |
| BytesSent | bigint | | Number of bytes sent |
| Format | varchar(8) | NOT NULL | Contains the name of the destination format: one of ase, bcp, csv, or iq. |

# monTask

Description — Specific to Adaptive Server in threaded mode, contains one row for each task.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns — The columns for monTask are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| InstanceID | tinyint | NOT NULL | ID of the instance within the cluster |
| KTID | int | | ID of the kernel task |
| ThreadPoolID | int | NULL | ID of the thread pool this task is associated with |
| ThreadID | int | | ID of the thread running this task |
| KPID | int | NULL | Adaptive Server kernel process ID (KPID) |
| SPID | int | | Session process identifier (spid) |
| Name | varchar(30) | | Name of the task |
| ThreadPoolName | varchar(30) | NULL | Name of the thread pool this task is associated with |

# monTempdbActivity

Description

(Cluster environments only) Provides statistics for all open local temporary databases, including global system tempdb when the instance is started in tempdb configuration mode.

monTempdbActivity requires the enable monitoring, per object statistics active, and object lockwait timing configuration parameters to collect data.

Columns

The columns for monTempdbActivity are:

| Name | Datatype | Description |
|---|---|---|
| DBID | int | Unique identifier for the database |
| InstanceID | tinyint | ID of the instance within the cluster |
| DBName | varchar(30) | Name of the database |
| AppendLogRequest | int | Number of semaphore requests from an instance attempting to append to the database transaction log |
| AppendLogWaits | int | Number of times a task waits for the append log semaphore to be granted |
| LogicalReads | int | Total number of buffers read |
| PhysicalReads | int | Number of buffers read from disk |
| APFReads | int | Number of asynchronous prefetch (APF) buffers read |
| PagesRead | int | Total number of pages read |
| PhysicalWrites | int | Total number of buffers written to disk |
| PagesWritten | int | Total number of pages written to disk |
| LockRequests | int | Number of requests for a object lock in this temporary database |
| LockWaits | int | Number of times a task waited for an object lock in this temporary database |
| CatLockRequests | int | Number of requests for a lock on the system catalog |
| CatLockWaits | int | Number of times a task waited for a lock for system table |
| AssignedCnt | int | Number of times this temporary database was assigned to a user task |
| SharableTabCnt | int | Number of sharable tables created |

# monThread

Description

Specific to Adaptive Server in threaded mode: Contains one row for each thread.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monThread are:

| Name | Datatype | Description |
|---|---|---|
| InstanceID | tinyint | ID of the instance within the cluster |
| ThreadID | int | ID of the thread pool |
| KTID | int | Internal kernel thread ID |
| OSThreadID | int | ID of the operating system thread |
| AltOSThreadID | int | Alternate operating system thread ID (on some platforms this may be a lightweight process (LWP) ID) |
| ThreadPoolID | int | ID of the thread pool |
| State | varchar(30) | Current state of the thread |
| ThreadAffinity | int | CPU number to which the thread has affinity |
| ThreadPoolName | varchar(30) | Name of the thread pool |
| TaskRuns | bigint | Number of tasks this thread has run |
| TotalTicks | bigint | Total number of ticks for this thread |
| IdleTicks | bigint | Total number of ticks during which this thread was idle |
| SleepTicks | bigint | Total number of ticks during which this thread was sleeping |
| BusyTicks | bigint | Total number of ticks during which this thread was busy |
| UserTime | bigint | Total amount of thread user CPU time, in miliseconds |
| SystemTime | bigint | Total amount of thread system CPU time, in miliseconds |
| MinorFaults | bigint | Total number of minor page faults. Value is 0 on Windows |
| MajorFaults | bigint | Total number of major page faults. Value is 0 on Windows |
| VoluntaryCtxtSwitches | bigint | Total number of voluntary operating system context switches. Value is 0 on Windows |
| NonVoluntaryCtxtSwitches | bigint | Total number of nonvoluntary operating system context switches. Value is 0 on Windows |

# monThreadPool

Description

Specific to Adaptive Server in threaded mode: Contains one row for each thread pool.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monThreadPool are:

| Name | Datatype | Description |
|---|---|---|
| ThreadPoolID | int | ID of the thread pool |
| Size | int | Number of threads in the thread pool |
| TargetSize | int | Requested size (differs from Size only when you change pool sizes) |
| Tasks | int | Number of tasks associated with the thread pool |
| ThreadPoolName | varchar(30) | Name of the thread pool |
| ThreadPoolDescription | varchar(255) | (Optional) description of the thread pool |
| Type | varchar(30) | Thread pool type, Engine (multiplexed) or Run to Completion (RTC) |
| IdleTimeout | int | Amount of time, in microseconds, that threads in this pool search for runnable tasks before idling |

# monWaitClassInfo

Description

Provides a textual description for all of the wait classes (for example, waiting for a disk read to complete). All wait events (see the description for monWaitEventInfo) have been grouped into wait classes that classify the type of event for which a process is waiting.

This table displays the same information for all instances in a shared-disk cluster

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monWaitClassInfo are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| WaitClassID | smallint | | Unique identifier for the wait event class |
| Description | varchar(50) | Null | Description of the wait event class |
| Language | varchar(30) | | Allows you to specify the language in which Adaptive Server returns the values of the Description column.<br>By default, Adaptive Server returns US English. Queries must use the the IS0-639 and ISO-3166 naming conventions. |

# monWaitEventInfo

Description

Provides a textual description of wait conditions reported in the monSysWaits and monProcessWaits tables.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monWaitEventInfo are:

| Name | Datatype | Attributes | Description |
|---|---|---|---|
| WaitEventID | smallint | | Unique identifier for the wait event type |
| WaitClassID | smallint | | Unique identifier for the wait event class |
| Description | varchar(50) | Null | Description of the wait event type |
| Language | varchar(30) | | Allows you to specify the language in which Adaptive Server returns the values of the Description column.<br>By default, Adaptive Server returns US English. Queries must use the the IS0-639 and ISO-3166 naming conventions. |

Join monWaitEventInfo with monProcessWaits or monSysWaits on the WaitEventID column to obtain the wait event descriptions listed in those tables.

# monWorkload

Description

(Cluster environments only) Displays the workload score for each logical cluster on each instance according to its load profile.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monWorkload are:

| Name | Datatype | Description |
|---|---|---|
| LCID | tinyint | Logical cluster ID |
| InstanceID | tinyint | ID of the instance within the cluster |
| LoadProfileID | tinyint | ID of the load profile used to generate the load score |
| LoadScore | int | Load score for this instance or logical cluster |
| ConnectionsScore | float | Weighted value for the `user connections` metric |
| CpuScore | float | Weighted value for the `cpu utilization` metric |
| RunQueueScore | float | Weighted value for the `run queue` metric |
| IoLoadScore | float | Weighted value for the `io load` metric |
| EngineScore | float | Weighted value for the `engine deficit` metric |
| UserScore | float | Weighted value for the `user` metric |
| LogicalClusterName | varchar(30) | Logical cluster name |
| InstanceName | varchar(30) | Instance name |
| LoadProfileName | tinyint | Name of the load profile used to generate the load score |

# monWorkloadPreview

Description

(Cluster environments only) Provides an estimate of how a load profile impacts the workload score without enabling the profile. monWorkload includes one row for each logical cluster and instance on which this logical cluster is running. The load score and components are based on the current profile for that logical cluster. The monWorkloadPreview table has one row for each combination of instance and load profile configured on the system, allowing the administrator to see how workload scoring would be done for each profile. You need not have the mon_role role to query this monitor table.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monWorkloadPreview are:

| Name | Datatype | Description |
|---|---|---|
| InstanceID | tinyint | ID of the instance within the cluster |
| LoadProfileID | smallint | Load profile ID |
| LoadScore | int | Load score for this instance or logical cluster |
| ConnectionScore | float | Weighted value for the `user connections` metric |
| CpuScore | float | Weighted value for the `cpu utilization` metric |
| RunQueueScore | float | Weighted value for the `run queue` metric |
| IoLoadScore | float | Weighted value for the `io load` metric |
| EngineScore | float | Weighted value for the `engine deficit` metric |
| UserScore | float | Weighted value for the `user` metric |
| InstanceName | varchar(30) | Instance name |
| LoadProfileName | varchar(30) | Name of load profile used to generate the load score |

# monWorkloadProfile

Description

(Cluster environments only) Displays currently configured workload profiles. You need not have the mon_role role to query this monitor table.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monWorkloadProfile are:

| Name | Datatype | Description |
|---|---|---|
| ProfileID | smallint | Workload profile ID |
| ConnectionsWeight | tinyint | Weight associated with the `active connections` metric |
| CpuWeight | tinyint | Weight associated with the `cpu utilization` metric |
| RunQueueWeight | tinyint | Weight associated with the `run queue` metric |
| IoLoadWeight | tinyint | Weight associated with the `io load` metric |
| EngineWeight | tinyint | Weight associated with the `engine deficit` metric |
| UserWeight | tinyint | Weight associated with the `user metric` metric |
| LoginThreshold | | Threshold for the login load distribution. |
| DynamicThreshold | smallint | Threshold for dynamic load distribution (that is, post-login migration for load purposes) |
| Hysteresis | tinyint | Minimum load score that enables redirection. |
| Name | varchar(30) | Workload profile name |
| Type | varchar(30) | Type of workload profile. Indicates whether the profile is defined by a user or the system. Values are:<br>• User<br>• System |

# monWorkloadRaw

Description

(Cluster environments only) Provides the raw workload statistics for each instance. You need not have the mon_role role to query this monitor table.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns

The columns for monWorkloadRaw are:

| Name | Datatype | Description |
|---|---|---|
| InstanceID | tinyint | ID of the instance within the cluster |
| ConnectionsRaw | float | Raw value for the `user connections` metric |
| CpuRaw | float | Raw value for the `cpu utilization` metric |
| RunQueueRaw | float | Raw value for the `run queue` metric |
| IoLoadRaw | float | Raw value for the `io load` metric |
| EngineRaw | float | Raw value for the `engine deficit` metric |
| UserRaw | float | Raw value for the `user` metric |
| InstanceName | varchar(30) | Instance name |

# monWorkQueue

Description — Provides information on work queues.

You need not enable any configuration parameters for this monitoring table to collect data.

Columns — The columns for monWorkloadQueue are:

| Name | Datatype | Description |
|---|---|---|
| InstanceID | tinyint | (Cluster environments only) ID of an instance in a shared-disk cluster. |
| CurrentLength | int | Current number of queued items |
| MaxLength | int | Maximum number of queued items |
| TotalRequests | int | Total number of requests |
| QueuedRequests | int | Total number of requests that waited for another request to finish |
| WaitTime | int | Amount of time, in millisecons, requests waited |
| Name | varchar(30) | Name of the work queue |

CHAPTER 4

# *sybpcidb* Tables

The sybpcidb database stores configuration information for the Java PCI Bridge and the PCA/JVM plug-in. This chapter describes the sybpcidb tables in alphabetical order.

You create sybpcidb, install its tables, and create its system stored procedures when you configure the server for Java. See the installation guide for your platform. See also *Java in Adaptive Server Enterprise* for information about how to use the sp_jreconfig and sp_pciconfig stored procedures that let you configure and display information in sybpcidb.

# pca_jre_arguments

Description — Stores information about the arguments used to configure the PCA/JVM plug-in.

Columns — Located in sybpcidb. The columns for pca_jre_arguments are:

| Name | Datatype | Description |
|---|---|---|
| jre_args_directive_index | int | The index of the directive to which the argument belongs. |
| jre_args_name | varchar(255) | The name of the argument. |
| jre_args_units | varchar(255) | The argument type. Values are:<br>• switch<br>• string<br>• number<br>• array |
| jre_args_number_value | int | If units=number, holds the number associated with the argument. |
| jre_args_string_value | varchar(255) | If units=string or units=array, holds the string value associated with the argument. |
| jre_args_description | varchar(255) | A brief text description of the argument. |
| jre_args_enabled | int | Values are:<br>• 0 – not enabled<br>• 1 – enabled (default) |
| jre_args_status | int | Reserved for future use. |

Indexes

- Unique clustered index on jre_args_directive_index, jre_args_name, jre_args_string_value

# pca_jre_directives

Description — Stores information about the directives used to configure the PCA/JVM.

Columns — Located in sybpcidb. The columns for pca_jre_directives are:

| Name | Datatype | Description |
|---|---|---|
| jre_directives_index | int | The index of the directive. |
| jre_directives_name | varchar(255) | The name of the directive. |
| jre_directives_description | varchar(255 | A text description of the directive. |
| jre_directives_enabled | int | Values are:<br>• 0 – not enabled<br>• 1 – enabled (default) |
| jre_directives_status | int | Reserved for future use. |

Indexes
- Unique clustered index on jre_directives_name.
- Unique nonclustered index on jre_directives_index.

# pci_arguments

Description — Stores information that defines each of the arguments used to configure the PCI Bridge.

Columns — Located in sybpcidb. The columns for pci_arguments are:

| Name | Datatype | Description |
|---|---|---|
| pci_args_directive_index | int | The index of the directive to which the argument belongs. |
| pci_args_name | varchar(255) | The name of the argument. |
| pci_args_units | varchar(255) | The units type. Values are:<br>• switch<br>• number |
| pci_args_number_value | int | When units=number, the value of number. If units=switch, the value is zero (0). |
| pci_args_string_value | varchar(255) | Reserved for future use. |
| pci_args_description | varchar(255) | Brief text description of the argument and its purpose. |
| pci_args_enabled | int | Values are:<br>• 0 – not enabled<br>• 1 – enabled (default) |
| pci_args_status | int | Reserved for future use. |

Indexes
- Unique clustered index on pci_args_directive_index, pci_args_name

# pci_directives

Description Stores the directives that configure the PCI Bridge.

Columns Located in sybpcidb. The columns for pci_directives are:

| Name | Datatype | Description |
|---|---|---|
| pci_directives_index | int | The index of the directive. |
| pci_directives_name | varchar(255) | The name of the directive. |
| pci_directives_description | varchar(255) | A description of the directive. |
| pci_directives_enabled | int | Values are:<br>• 0 – not enabled<br>• 1 – enabled (default) |
| pci_directives_status | int | Reserved for future use. |

Indexes
- Unique clustered index on pci_directives_name
- Unique nonclustered index on pci_directives_index

# pci_slotinfo

Description — Contains information describing each slot, including table names for the slot's directives and arguments.

Columns — Located in sybpcidb. The columns for pci_slotinfo are:

| Name | Datatype | Description |
|---|---|---|
| slot_number | int | The number of the slot. |
| slot_name | varchar(255) | The name of the slot, such as JVM. |
| slot_pca_directives_table_name | varchar(255 | The name of the PCA directives table, such as pca_jre_directives. |
| slot_pca_arguments_table_name | varchar(255 | The name of the PCA arguments table, such as pca_jre_arguments. |
| slot_status | varchar(255) | Reserved for future use. |

Indexes

- Unique clustered index on slot_name
- Unique nonclustered index on slot_number

# pci_slot_syscalls

Description — Contains the runtime system call configuration information for the runtime dispatching model used by the PCI Bridge.

Columns — Located in sybpcidb. The columns for pci_slot_syscalls are:

| Name | Datatype | Description |
|---|---|---|
| syscall_slot_number | int | The slot number associated with the system call. |
| syscall_system_call | varchar(255) | The name of the system call. |
| syscall_dispatch_name | varchar(255) | The name of the dispatch function for the system call. |
| syscall_enabled | int | Values are:<br>• 0 – not enabled<br>• 1 – enabled (default) |
| syscall_status | int | Reserved for future use. |

Indexes

- Unique clustered index on syscall_slot_number, syscall_system_call

# Index

## E



## F



## G



## H



## I



## J



## K



## L

## M

## N



## O



## P



## R

## S

## T

## U



## V

## W