SYBASE®

New Feature Summary

Adaptive Server® Enterprise 15.5

DOCUMENT ID: DC01165-01-1550-03
LAST REVISED: May 2010

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

Contents

New Features in Adaptive Server Version 15.5 Cluster Edition

Adaptive Server® 15.5 Cluster Edition introduces multiple simultaneous failover, distributed transaction management, the **mount** and **unmount** commands, and the ability to use **alter database** to add space to an archived database.

Note: The Cluster Edition does not currently support in-memory databases, relaxed-durability databases, template databases, or minimally-logged DML.

Adaptive Server 15.5 Cluster Edition Feature and Platform Matrix

The feature and platform matrix shows feature availability for supported operating systems in Adaptive Server 15.5 Cluster Edition. A “Y” indicates the feature is supported for that platform.

Adaptive Server Cluster Edition options	HP-UX Itanium 64-bit	IBM AIX 64-bit	Linux Opteron 64-bit	Solaris 64-bit
Security and directory services	Y	Y	Y	Y
Cybersafe Kerberos				Y
Pluggable Authentication Module	Y	Y	Y	Y
Fine-grained access control	Y	Y	Y	Y
LDAP server directory	Y	Y	Y	Y
LDAP user authentication	Y	Y	Y	Y
Platform Native Kerberos				Y
Secure Sockets Layer	Y	Y	Y	Y
MIT Kerberos	Y	Y	Y	Y

Adaptive Server Cluster Edition options	HP-UX Itanium 64-bit	IBM AIX 64-bit	Linux Opteron 64-bit	Solaris 64-bit
Encrypted columns, including fine-grained access control (FGAC)	Y	Y	Y	Y
High availability				
Partitions	Y	Y	Y	Y
In-memory databases				
Tivoli Storage Manager for Backup Server	Y	Y	Y	Y
Active messaging	Y	Y	Y	Y
Enhanced Full-Text Search (EFTS)				
Features included in base Adaptive Server				
Cross-platform dump and load	Y	Y	Y	Y
Job Scheduler	Y	Y	Y	Y
Native XML	Y	Y	Y	Y
IPv6	Y	Y	Y	Y
Java option	Y	Y	Y	Y
Web Services	Y	Y	Y	Y
Distributed transaction management	Y	Y	Y	Y
Content management (external file support)	Y	Y	Y	Y
Archived database access	Y	Y	Y	Y

Multiple simultaneous failover

Adaptive Server Cluster Edition version 15.5 and later support multiple simultaneous instance failures.

Note: Adaptive Server version 15.5 Cluster Edition does not support multiple simultaneous failure on Veritas VCS.

Multiple simultaneous failure support occurs when more than one instance fails within a single cluster view, but the cluster remains online and provides the same failover recovery as it does when a single instance fails.

The number of failures cannot be greater than the value for **cluster redundancy level**, a configuration parameter that allows a database administrator to set the maximum number of recoverable simultaneous instance failures for the cluster.

Adding space to an archive database

Adaptive Server Cluster Edition version 15.5 and later supports archive databases. In general, access to an archive database is the same in both a clustered and a nonclustered Adaptive Server. In either scenario, when an archive database runs out of space, use the **alter database** command to add space to the archive database.

In a clustered Adaptive Server, run **alter database** from the same node that is updating the archive database. If you run **alter database** from a different node, Adaptive Server prints an error message with the number of the node that is actually updating the archive database.

Distributed transaction management in the shared-disk cluster

In version 15.5 and later, Adaptive Server supports distributed transaction management (DTM) on its clustered architecture.

The clustered Adaptive Server:

- Is fully compliant with the X/Open XA protocol when it acts as the resource manager (RM), without additional services, such as XA-Server.
- Ensures consistent commit or rollback for transactions that update Adaptive Server data via remote procedure calls (RPCs) and Component Integration Services (CIS).
- Can be part of distributed transactions coordinated by other Adaptive Server installations using the Adaptive Server Transaction Coordination (ASTC) mechanism.

- Can coordinate the distributed transactions across multiple Adaptive Server installations using the ASTC mechanism.

Note: The Cluster Edition does not support the Microsoft Distributed Transaction Coordinator (MSDTC) proprietary protocol.

In general, the user interface for distributed transactions is the same in the Adaptive Server clustered environment as in the nonclustered environment. Applications using DTM on a nonclustered Adaptive Server can use the same applications on the clustered Adaptive Server. See *Using Adaptive Server Distributed Transaction Management Features*.

Although the user interface for distributed transactions on the Cluster Edition is the same as that for nonclustered Adaptive Server, support for distributed transactions on the cluster must take into account the cluster-specific issues described in the *Clusters Users Guide*. For example:

- Using the cluster as the resource manager (RM)
- Requests to nonowner instances
- Handling instance failures
- Using transaction coordination with ASTC
- Impact of connection migration
- System configuration

System Changes Adaptive Server 15.5 Cluster Edition

Adaptive Server 15.5 Cluster Edition supports changes to commands, monitoring tables and configuration parameters.

Changed commands

In Adaptive Server Cluster Edition version 15.5 and later, you can use **mount database** and **unmount database** in a shared-disk cluster.

If an instance fails while **mount database** or **unmount database** is in progress, the command may abort. In this case, you must reissue **mount database** or **unmount database** when the instance failover recovery is complete.

Monitoring Tables

Monitoring tables added for Adaptive Server version 15.5.

Adaptive Server Cluster Edition version 15.5 adds these monitoring tables:

- monTableTransfer – provides historical transfer information for tables in Adaptive Server active memory.
- monInmemoryStorage – used for in-memory databases. For internal purposes only.

The Cluster Edition version 15.0.1 and later include monitoring tables to collect table statistics. These are the monitoring tables included with version 15.5:

- monCIPC – provides summary figures for total messaging within the cluster, as viewed from the current instance or all instances.
- monCIPCEndpoints – provides a detailed summary, giving traffic data for each subsystem within the cluster instance.
- monCIPCLinks – monitors the state of the links between instances in the cluster.
- monCIPCMesh – gives summary figures for the mesh of connections, from the current instance to all other instances in the cluster, on a per-instance basis.
- monCLMObjectActivity – collects cluster lock information.
- monClusterCacheManager – stores diagnostic information about the cluster cache manager daemon running on each instance. monClusterCacheManager reports cluster-wide information on a per-instance basis.

- monCMSFailover – 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.
- monDBRecovery – contains rows from all instances in the cluster and contains rows for every database that contributes to recovery.
- monDBRecoveryLRTypes – 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.
- monFailoverRecovery – contains aggregated failover recovery diagnostic information for the cluster lock manager (CLM), database recovery, and CMS modules.
- monLogicalCluster – displays information about the logical clusters currently configured on the system.
- monLogicalClusterAction – shows all administrative actions against local clusters from start-up until these actions are released.
- monLogicalClusterInstance – displays information about the many-to-many relationship between instances and logical clusters.
- monLogicalClusterRoute – displays information about the configured routes (application, login, and alias bindings). You need not have the mon_role role to query this monitor table.
- monPCM – 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.
- monProcessMigration – displays information about the connection currently migrating.
- monSysLoad – provides trended statistics on a per-engine basis. You need not have the mon_role role to query this monitor table.
- monTempdbActivity – provides statistics for all open local temporary databases, including global system tempdb when the instance is started in tempdb configuration mode.
- monWorkload – displays the workload score for each logical cluster on each instance according to its load profile.
- monWorkloadPreview – 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.
- monWorkloadProfile – displays currently configured workload profiles. You need not have the mon_role role to query this monitor table.
- monWorkloadRaw – provides the raw workload statistics for each instance. You need not have the mon_role to query this monitor table.

Configuration Parameters

New configuration parameter for Adaptive Server Cluster Edition version 15.5 and later.

cluster redundancy level – The maximum number of instances that can fail simultaneously while allowing recovery to proceed concurrently with other activity. The cluster shuts down if the failed number of instances exceeds the maximum.

Functions

New functions for Adaptive Server Cluster Edition versions 15.5 and later.

- **xact_owner_instance** – returns the instance on which an external transaction is running, or 0.
- **xact_conmigrate_check** – determines whether a connection can process an external transaction.

New Features in Adaptive Server® Version 15.5

Adaptive Server 15.5 introduces in-memory and relaxed-durability databases, Backup Server support for IBM Tivoli Storage Manager, faster compression for backups, deferred name resolution for stored procedures, incremental data transfer, support for FIPS 140-2 password encryption, and new datatypes.

Adaptive Server 15.5 Feature and Platform Matrix

The feature and platform matrix shows feature availability for supported operating systems in Adaptive Server 15.5. A “Y” indicates the feature is supported for that platform.

Adaptive Server options	HP-UX Itanium 64-bit	HP-UX PA Risc 64-bit	IBM AIX 64-bit	Linux on Power 64-bit	Linux Opteron 64-bit	Linux x86 32-bit	Solaris 32-bit	Solaris 64-bit	Solaris Opteron 64-bit	Windows Opteron X64	Windows x86 32-bit
Security and directory services	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Cybersafe Kerberos							Y	Y			Y
Pluggable Authentication Module	Y		Y	Y	Y	Y	Y	Y	Y		
Fine-grained access control	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
LDAP server directory	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
LDAP user authentication	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Platform Native Kerberos							Y	Y			
Secure Sockets Layer	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
MIT Kerberos	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y
Encrypted columns, including fine-grained access control (FGAC)	Y	Y	Y		Y	Y	Y	Y	Y	Y	Y

Adaptive Server options	HP-UX Itanium 64-bit	HP-UX PA Risc 64-bit	IBM AIX 64-bit	Linux on Power 64-bit	Linux Opteron 64-bit	Linux x86 32-bit	Solaris 32-bit	Solaris 64-bit	Solaris Opteron 64-bit	Windows Opteron X64	Windows x86 32-bit
High availability	Y	Y	Y		Y	Y	Y	Y			Y
Partitions	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
In-memory databases	Y	Y	Y	Y	Y			Y	Y	Y	
Tivoli Storage Manager for Backup Server	Y		Y		Y			Y	Y	Y	
Active messaging	Y	Y	Y		Y	Y	Y	Y	Y	Y	Y
Enhanced Full-Text Search (EFTS)		Y	Y		Y	Y	Y	Y			Y
Features included in base Adaptive Server											
Cross-platform dump and load	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Job Scheduler	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Native XML	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
IPv6	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Java option	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Web Services	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Distributed transaction management	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Content management (external file support)	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Archived database access	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

In-Memory and Relaxed-Durability Databases

In-memory and relaxed-durability databases provide enhanced performance.

In-memory databases run entirely in a named cache (that is, in the Adaptive Server memory space), without using disk storage for data or logs. Because an in-memory database does not require I/O, its performance can be much better than a traditional, disk-resident database. In-memory databases are not designed for recovery: their transaction logs are written to the cache and not to disk, and any data changes are lost if the server is restarted. In-memory databases perform transactional logging for runtime rollback, and for other operations, such as firing triggers, deferred mode updates, replication, and so on.

Disk-resident databases perform writes to disk, and ensure that the transactional properties of atomicity, consistency, integrity, and durability (known as the ACID properties) are maintained. Durability refers to the persistence of transactions after they have committed. A traditional Adaptive Server database operates at full durability by writing its transaction log to disk when a transaction commits. This, along with data pages being written periodically to disk, ensures that all committed transactions are durable.

In-memory databases do not write data or log to disk, and trade the guarantee of transaction durability for performance improvements. In the event of a database failure, in-memory databases cannot be recovered. If your applications require data recoverability following a server failure or a normal shutdown, consider using a traditional Adaptive Server database.

With support for relaxed durability, Sybase extends the performance benefits of an in-memory database to disk-resident databases. A traditional disk-resident database guarantees transactional recovery from a server failure. Relaxed-durability databases trade the full durability of committed transactions for enhanced runtime performance for transactional workloads.

The performance benefits of in-memory and relaxed-durability databases include:

- An in-memory database operates entirely in-memory, so it does not wait for I/O.
- Improved buffer and user log cache management, so you need not create an overhead of user log cache flushes and buffer management when Adaptive Server performs concurrent updates to the same data.
- Runtime strategies that may avoid flushing task changes registered in the user-log cache to the transaction log when the transaction commits or aborts. These strategies reduce the contention on in-memory log pages.
- Support for minimally logged DML operations that use in-memory logging techniques improves the performance of large-volume DML operations.

Adaptive Server version 15.5 allows you to create these types of databases:

- Disk-resident databases with durability set to full (this is the default Adaptive Server database)

- User-defined disk-resident temporary databases
- In-memory user databases with durability set to no_recovery
- User-defined in-memory temporary databases with durability set to no_recovery
- Disk-resident relaxed-durability databases with durability set to no_recovery or at_shutdown

Adaptive Server supports **dump database** and **load database** for both in-memory and relaxed-durability databases. In addition, you can dump an in-memory database and load it into a disk-resident database, and vice versa. Adaptive Server supports template databases as a way to reinitialize the contents of nonrecoverable databases when the server restarts.

See the *In-Memory Database Users Guide*.

Faster Compression for Backups

New compression levels for dumping databases and transactions provide faster, more complete, and less CPU-intensive compression.

Two new compression levels have been added to the **dump database...compression=** and**dump transaction...compression=** commands: 100 and 101. Compression level 100 provides faster compression; level 101 provides better compression. Both new compression levels are less CPU-intensive than levels 0 – 9.

See the *System Administration Guide* and the *Reference Manual: Commands.*

Backup Server Support for the IBM® Tivoli® Storage Manager

Adaptive Server 15.5 supports IBM Tivoli Storage Manager (TSM) for storage management services in partnership with Backup Server. Support for TSM is a licensed Adaptive Server feature.

The TSM is a third-party client/server program that provides storage management services to licensed users. The Adaptive Server Backup Server supports tape drives and disk files as backup media. TSM works with Backup Server to enable a greater variety of backup media. You can store Adaptive Server backups on any backup media supported by TSM.

Adaptive Server handles the dump and load of databases and transactions to and from TSM; TSM handles storage and retrieval from the storage media. TSM provides storage management services, but you can administer all your Adaptive Server backup and restore operations, including queries for backup objects and deletion of backup objects, from Backup Server.

When you execute the **dump** or **load** commands, Backup Server invokes the Sybase interface with the TSM API, which provides communication with the TSM. When you use the **dump** command, you specify an object name that is uniquely associated with the backup object. This

object name is the same as the TSM object name, and should later be used to specify the same database or transaction dump for the load operation. In general, you can use the same options with the **dump** and **load** commands with TSM as you use with Backup Server when TSM is not configured.

Adaptive Server introduces these stored procedures to support the query and delete of backup objects from TSM:

- **sp_deletesmobj**– deletes some or all of the current server's backup objects from TSM.
- **sp_querysmobj** – retrieves a list of a server's backup objects.

See *Using Backup Server with IBM Tivoli Storage Manager*.

Deferred Name Resolution for User-Defined Stored Procedures

Deferred name resolution lets you create a stored procedure before referenced objects exist.

In versions of Adaptive Server earlier than 15.5, referenced objects were required to already exist before the stored procedure using them could be created. The deferred name resolution feature now allows objects, except for user-defined datatype objects, to be resolved when the stored procedure is executed for the first time.

This feature uses the **deferred name resolution** configuration parameter, which operates at the server level, or a new **set** command, **set deferred_name_resolution**, which operates at the connection level.

See the *Transact-SQL Guide*, the *System Administration Guide: Volume 1*, and the *Reference Manual: Commands*.

FIPS 140-2 Login Password Encryption

Adaptive Server 15.5 supports the FIPS 140-2 validated cryptographic module.

FIPS 140-2 certifed Certicom Security Builder GSE encrypts login passwords in a transmitted login packet, in memory and on disk, if you use the configuration parameter FIPS login password encryption:

```
sp_configure 'FIPS login password encryption', 1
```

Note: A Security and Directory Services license is required to enable this parameter. If the parameter is not enabled, the OpenSSL security provider performs login password encryption.

See the *Users Guide for Encrypted Columns*.

Incremental Data Transfer

Incremental data transfer lets you transfer data to Adaptive Server or other products.

The **transfer table** command allows you to transfer data incrementally, and, if required, to a different product. In versions of Adaptive Server earlier than 15.5, you could transfer only whole tables from one Adaptive Server to another.

Note: Adaptive Server enables the data transfer feature when you purchase, install, and register the in-memory database license.

Incremental data transfer:

- Lets you export data, including only the data that has changed since a prior transmission, from Adaptive Server tables that are marked for incremental transfer.
- Allows table data to be read without obtaining the usual locks, without guaranteeing any row retrieval order, and without interfering with other ongoing reads or updates.
- Lets you write selected rows to an output file (which can be a named pipe) formatted for a defined receiver: IQ (Sybase IQ), ASE (Adaptive Server Enterprise), bulk copy (**bcp**), or character-coded output. All selected rows are transmitted without encryption, and, by default, any encrypted columns within the row are decrypted before transmittal. The file to which you are writing must be visible to the machine on which Adaptive Server is running (the file can be an NFS file that Adaptive Server can open as a local file).
- Maintains a history of transmissions for eligible tables, and lets you remove transmission history when it is no longer wanted. Exports data from tables declared ineligible for incremental transfer, subject to certain restrictions.
- Transfers entire rows from indicated tables. You cannot currently select certain columns, select a partition within a table, or transfer results from SQL queries.

See Chapter 8, "Adding, Changing, Transferring, and Deleting Data," in the *Transact-SQL Users Guide.*

bigdatetime and bigtime Datatypes

bigdatetime and *bigtime* provide microsecond precision.

Two new datatypes, *bigdatetime* and *bigtime*, provide a timestamp with microsecond precision that contains the year, month, day, hour, minute, second, and fraction of a second to six decimal places. A *bigdatetime* value requires 8 bytes of storage. A *bigtime* value contains the time of day, containing hour, minute, second, and fraction of a second. The fraction is stored to six decimal places. A *bigtime* value requires 8 bytes of storage.

See the *Adaptive Server Transact-SQL Users Guide*, the *Reference Manual: Building Blocks*, and *Java in Adaptive Server Enterprise*.

Creating and Managing tempdb Groups

Updates for **tempdb** groups for version 15.5

Adaptive Server allows you to create and manage user-created **tempdb** groups in addition to managing the default **tempdb** group. User-created **tempdb** groups can include other user-created temporary databases, and support application and login binding.

You cannot remove the system **tempdb** from the default temporary database group. You cannot add system **tempdb** to any other user-created **tempdb** group.

You can designate and administer user-created **tempdb** groups to contain only disk-resident or in-memory temporary databases. The server does not explicitly impose any such restriction, but by controlling the membership you can assign disk-only or in-memory-only **tempdb** groups to specific logins or applications.

System Changes in Adaptive Server 15.5

Adaptive Server 15.5 supports new and changed datatypes, functions, system procedures, commands, configuration parameters, monitoring tables, system tables, and utilities. New auditing options have also been added.

Datatypes

New *bigtime* and *bigdatetime* datatypes provide precision timestamp information.

Datatypes	Description
bigtime	A *bigtime* value includes the hour, minute, second, and fraction of a second. The fraction is stored to 6 decimal places.
bigdatetime	A *bigdatetime* value includes the year, month, day, hour, minute, second, and fraction of a second to 6 decimal places.

Two new functions return *bigtime* and *bigdatetime* values:

- **current_bigtime**
- **current_bigdatetime**

bigtime and *bigdatetime* can be used in these existing functions:

- **datepart**
- **datename**
- **datediff**
- **dateadd**

See the *Reference Manual: Blocks*.

Functions

New and changed functions in Adaptive Server 15.5

Table 1. New functions

Function	Description
db_attr	Returns the durability, dml_logging, and template settings for the specified database.
object_attr	Reports the table's current logging mode, depending on the session-specific, table-wide, and database-wide settings.
cache_usage	Returns cache usage for the cache to which the specified object is bound, as the percentage of the cache which is currently in use by all objects bound to that cache.
current_bigdatetime	Returns a *bigdatetime* value representing the current date and time with microsecond precision.
current_bigtime	Returns a *bigtime* value representing the current time with microsecond precision.

Table 2. Changed functions

Function	Description
datepart	Produces the specified *datepart* argument of the specified date as an integer.
datename	Produces the specified *datepart* as a character string.
datediff	Calculates the number of date parts between two specified dates or times.
dateadd	Adds an interval to a specified date or time.

See the *Reference Manual: Commands*.

System Stored Procedures

New and changed system stored procedures support Tivoli Storage Manager, and in-memory and relaxed-durability databases.

Table 3. New system stored procedures

System stored procedures	Description
sp_deletesmobj	Deletes backup objects from the TSM.
sp_querysmobj	Retrieves a list of backup objects from the TSM.

Table 4. Changed system stored procedures

System stored procedures	Description
sp_addsegment	Updated to manage space in in-memory databases.
sp_addthreshold	Updated to manage space in in-memory databases.
sp_bindcache	You cannot bind objects or databases to in-memory storage caches, and you cannot bind an in-memory database or objects in an in-memory database to any cache.
sp_cacheconfig	Creates, extends the size of, or drops, an in-memory storage cache.
sp_cachestrategy	The prefetch and MRU parameters do not apply to tables and indexes in in-memory databases.
sp_dbextend	Automatic database expansion is currently not supported for in-memory databases.
sp_deviceattr	The directio and dsync device attributes do not apply to in-memory devices.
sp_downgrade	Supports downgrading an Adaptive Server containing in-memory or relaxed-durability databases, or databases using templates or minimal logging.
sp_diskdefault	You cannot use **sp_diskdefault** to specify in-memory devices as a default devices.

System stored procedures	Description
sp_dropdevice	Drops an in-memory device created from an in-memory storage cache.
sp_dropsegment	Updated to manage space in in-memory databases.
sp_dropthreshold	Updated to manage space in in-memory databases.
sp_extendsegment	Updated to manage space in in-memory databases.
sp_help	Reports on properties, such as minimal logging attribute, for a table.
sp_helpcache	Displays properties of the in-memory storage cache, the in-memory database created on it, and details of free space on this cache.
sp_helpdb	Reports on database properties, such as durability, DML logging level, in-memory or not, use, if any, of a template database, or as a template database.
sp_helpdevice	Reports the in-memory device properties created from an in-memory storage cache.
sp_modifythreshold	Updated to manage space in in-memory databases.
sp_plan_dbccdb	Sets up **dbccdb** for checkstorage execution in an in-memory database.
sp_poolconfig	Large I/O buffer pools are not supported in an in-memory database.
sp_post_xpload	Supports cross-platform operations for in-memory databases.
sp_tempdb	Supports user-created temporary database groups, login or application bindings to temporary database groups and for in-memory databases.
sp_unbindcache, sp_unbindcache_all	You cannot unbind objects in or the in-memory database itself from the host in-memory storage cache.

See the *Reference Manual: Procedures*.

Commands

New and changed commands for Adaptive Server 15.5

Table 5. New commands

Command	Description
transfer table	Initiates an incremental table transfer. A new grant with grant option supports **transfer table**. It grants a specified user permission to transfer a specified table. `grant transfer table on table_name to user with grant option`

Table 6. Changed commands

Command	Change
alter database	Syntax added to support changing the durability of a database, the level of DML logging, a database's template.
alter table	Syntax added to support changing a table's logging mode for insert, update, and delete (DML) operations. Syntax added to support **transfer table**: `set transfer table [on | off]`
create database	Syntax added to create in-memory and relaxed-durability databases with durability set to full, no_recovery, or at_shutdown. Syntax added to specify DML logging level and template database, if any.
create table	Syntax added to specify DML logging level for tables in in-memory databases. Syntax added to support **transfer table**: `with transfer table [on | off]`
disk init	Syntax added to create in-memory data devices for in-memory databases.
dump database	Syntax added to support the Tivoli Storage Manager (TSM). The keyword **syb_tsm** invokes the Sybase interface with the TSM API (libsyb_tsm). `database_name to "syb_tsm::object_name"`

Command	Change
dump database ... compression=	Syntax added to support faster, less CPU-intensive compression levels 100 and 101.
dump transaction	Syntax added to support TSM. The keyword **syb_tsm** invokes the Sybase interface with the TSM API (libsyb_tsm). `database_name to "syb_tsm::object_name"`
dump transaction ... compression=	Syntax added to support faster, less CPU-intensive compression levels 100 and 101.

load database	Syntax added to support TSM. The keyword **syb_tsm** invokes the Sybase interface with the TSM API (libsyb_tsm).
	`database_name from "syb_tsm::`
	`[[-S source_server_name]`
	`[-D source_database_name]::]object_name"`

load transaction	Syntax added to support TSM. The keyword **syb_tsm** invokes the Sybase interface with the TSM API (libsyb_tsm).
	`database_name from "syb_tsm::`
	`[[-S source_server_name]`
	`[-D source_database_name]::]object_name"`

select into	Syntax added to specify the DML logging level for tables created by selecting into in-memory or relaxed-durability databases.
set	Adds: • **dml_logging** parameter for specifying the amount of logging for a session. • **deferred_name_resolution** for activating deferred name resolution at the connection level. • **builtin_date_strings** *number* Values are: • 0 - if a string is given as an argument to a chronological system function, the server interprets it as a *datetime* value regardless of its apparent precision. This is the default. • 1 - makes the server interpret the argument string as *bigdatetime*. This affects the result of chronological system functions. .

See the *Reference Manual: Commands*.

Configuration Parameters

Adaptive Server 15.5 introduces the **deferred name resolution** configuration parameter.

Configuration parameter	Description
deferred name resolution	Allows you to create procedures using deferred name resolution. Values are: • 0 – disables deferred name resolution. This is the default. • 1 – enables deferred name resolution.
builtin date strings	Values are: • 0 – causes string literals given to a chronological builtin function as an argument to be interpreted as a datetime type. • 1 – causes string literals given to a chronological builtin function as an argument to be interpreted as a bigdatetime type.

See the *System Administration Guide: Volume 1* and the *Transact-SQL Users Guide*.

Monitoring Tables

Adaptive Server 15.5 introduces the *monTableTransfer* monitoring table.

Monitoring table	Description
monTableTransfer	Provides historical transfer information for tables in the Adaptive Server active memory.

See the *Reference Manual: Tables*.

System Tables

New and changed system tables in Adaptive Server 15.5

Table 7. New system tables

System table	Description
spt_TableTransfer	Stores the results from table transfers.

Table 8. Changed system tables

System table	Change description
sysdevices	Lists the in-memory storage cache under the *name* and *phyname* columns. In-memory devices do not include a full path to the disk device, instead, they store the name of the cache on which the in-memory device has been created.
sysdatabases	Adds the *durability* column, which indicates the durability level of the database. The *durability* column has the *int* datatype. Its values are: • 1 – full • 5– at_shutdown • 6 – no_recovery

See the *Reference Manual: Tables*.

Utilities

New and changed utilities in Adaptive Server 15.5

Table 9. New utilities

Utility	Description
openssl	Performs all certificate management tasks implemented by certreq, certauth and certpk12. Sybase includes this binary as a convenience, and is not responsible for any issues incurred using the binary. See *www.openssl.org* for details.

Table 10. Changed utilities

Utility	Change
backupserver	Syntax change supports another verbosity level (V4) for the -V parameter. V4 displays all -V0 messages except "Connection from Server" messages printed for each connection event.

See the *Utility Guide*.

Auditing

Auditing options are added in support of in-memory and relaxed-durability databases, incremental data transfer, and deferred name resolution.

Table 11. Auditing enhancements

Audit option	Command or access to be audited	Event	Information in extrainfo
all, create	**transfer table**	136	Keywords or options: transfer_table
all, create	**alter table**	3	If **alter table** contains **set transfer table on**, Adaptive Server prints this to extrainfo: SET TRANSFER TABLE ON. If **alter table** contains **set transfer table off**, Adaptive Server prints this to extrainfo: SET TRANSFER TABLE OFF.
all, create	**create table**	12	If **create table** contains **with transfer table on**, Adaptive Server prints this to extrainfo: WITH TRANSFER TABLE ON. If **create table** contains **with transfer table off**, Adaptive Server prints this to extrainfo: WITH TRANSFER TABLE OFF.
all, create	**create database**	9	Keywords or options: inmemory
all, create	**alter database**	2	Keywords or options: inmemory

Audit option	Command or access to be audited	Event	Information in extrainfo
all, create	**create procedure**	11	Keywords or options: deferred_name_resolution

New Features in Adaptive Server 15.0.3

Adaptive Server 15.0.3 introduces distributed transaction management, enhancements to the Java interface, virtually hashed tables, huge pages, updates to the Adaptive Server Plug-in, directions for upgrading during a High Availability configuration, and support for SQL statement replication.

SQL Statement Replication

Replication Server 15.2 supports SQL statement replication for Adaptive Server databases

Adaptive Server Enterprise 15.0.3 introduces SQL statement replication, which is supported by Replication Server 15.2 and later.

See the Replication Server documentation.

Security Enhancements

Adaptive Server version 15.0.3 introduces several new security enhancements.

LDAPS User Authentication Enhancement

Modifying the CA trusted root file no longer requires a server restart.

In earlier versions of Adaptive Server, if you modify the Certifying Authority (CA) trusted root file, you must restart Adaptive Server for the modifications to take effect. Adaptive Server version 15.0.3 and later supports modifications to the trusted root file, so that restarting the the server is unnecessary. A new subcommand, **reinit_descriptors**, which unbinds the LDAP server descriptors and reinitializes the user authentication subsystem.

Automatic LDAP User Authentication and Failback

The Adaptive Server housekeeping utility can automatically activate a failed LDAP server

Adaptive Server 15.0.3 provides support for a secondary LDAP server. Previously, after bringing a failed primary LDAP server online, it was necessary to activate the LDAP server manually, in order to authenticate new LDAP logins and move them to the primary LDAP server.

In versions 15.0.3 and later, a new chore has been added to Adaptive Server's housekeeping utility to activate an LDAP server automatically: **'set_failback_interval'**.

After you set the failback interval, the housekeeper task checks for failed LDAP servers each time it sweeps through its chores. When it finds a failed LDAP server, it attempts to activate the LDAP server when the failback time interval expires.

Login Mapping of External Authentication

Adaptive Server can map one unique mapping of an external user to an internal Adaptive Server login

When you configure an external authentication mechanism, if there is exactly one mapping of an external user to an internal Adaptive Server login, and if it is successfully authenticated, Adaptive Server updates the internal login's password to match the external user's password.

For example, under these conditions:

1. USER1 has an Adaptive Server login name of "user_ase" with password "user_password".
2. Another user has an LDAP login name of "user_ldap" with password "user_ldappasswd".
3. Adaptive Server has a one to one mapping for "user_ldap" to "user_ase".
4. User "user_ldap" logs in to Adaptive Server using password "user_ldappasswd". Adaptive Server updates the "user_ase" password to "user_ldapppasswd".

Using SSL to Specify a Common Name

Use a fully-qualified domain name for the SSL certificate common name

The server name specified in the directory service entry can be different from the common name the SSL server certificate uses to perform an SSL handshake. This allows you to use a fully-qualified domain name for the SSL certificate common name (for example, `server1.bigcompany.com`).

To add a common name to the interfaces file, use:

```
ase1
   master tcp ether host_name port_number ssl="CN='common_name'"
   query tcp ether host_name port_number ssl="CN='common_name'"
```

When clients use SSL to connect to an Adaptive Server that also uses SSL, the SSL filter is placed after the port number in the `interfaces` file. The directory service includes the common name, which you add either by using **dsedit** or a text editor.

sp_listener includes the **CN=common_name** parameter, which allows you to specify a common name for the SSL certificate.

Concurrent Kerberos Authentication

Adaptive Server can establish multiple Kerberos authentication sessions

Adaptive Server version 15.0.3 supports concurrent Kerberos authentication, whereas earlier versions used locking mechanisms during Kerberos authentication to protect internal data structures.

When there are concurrent logins using Kerberos authentication, Adaptive Server now establishes multiple Kerberos authentication sessions.

Version 15.0.3 also resolves an issue with concurrent login sessions, which may be blocked during Kerberos authentication. This concurrency issue occurs when you use prior versions of Adaptive Server with MIT version 1.3.x and 1.4.x Kerberos GSSAPI libraries.

Virtually Hashed Tables

Create virtually hashed tables to efficiently organize tables.

Note: Virtually hashed tables are available only on Linux pSeries.

You can perform hash-based index scans using nonclustered indexes or clustered indexes on data-only-locked tables. During this scan, each worker process navigates the higher levels of the index and reads the leaf-level pages of the index. Each worker process then hashes on either the data page ID or the key value in a separate hash table to determine which data pages or data rows to process.

A virtually hashed table can be a more efficient way to organize a table because it does not require a separate hash table. Instead, it stores the rows so that, using the hash key, the query processor can determine the row ID (based on the row's ordinal number) and the location of the data. Because it does not use a separate hash table to hold the information, it is called a "virtually" hashed table.

For systems that must make more efficient use of their central-processing unit (CPU), the virtually hashed table is a good option.

To create a virtually hashed table, specify the maximum value for the hash region using the **create table** command.

Huge Pages

Enable huge pages to use fewer pages to cover the physical address space.

Note: This feature is available only on Linux pSeries.

The CPU-Cache translation lookaside buffer (TLB) stores information about conversions from an virtual page address to the physical page address, and every byte access to physical memory requires a conversion (called a "cache miss"). Although these cache misses are very expensive, you can improve the TLB hits by enabling "huge pages."

Once enabled, huge pages use fewer pages to cover the physical address space, so the size of "book keeping" (mapping from the virtual to the physical address) decreases, requiring fewer entries in the TLB and improving the system performance.

Adaptive Server version 15.0.3 and later allocates shared memory using huge pages by default. However, if the system does not have enough huge pages—or is not configured for huge pages—Adaptive Server uses regular pages.

To enable huge pages, start Adaptive Server with traceflag 1653. Adaptive Server adjusts its shared memory up to the nearest multiple of 256MB.

Upgrading During a High Availability Configuration

After you have enabled high availability (HA), follow the upgrade instructions in the Adaptive Server installation guide for your platform

The instructions in this section supplement those in the *Installation Guide*.

Reinstalling System Stored Procedures

Reinstall the system stored procedures after enabling high availability

1. Disable HA in the primary server:

   ```
   sp_companion secondary-server-name, 'drop'
   sp_configure 'enable HA', 0
   ```

2. Disable HA in the secondary server:

   ```
   sp_configure 'enable HA', 0
   ```

3. Restart the servers.
4. Run the `installmaster` script on both servers.
5. Enable the HA property on both servers:

   ```
   sp_configure 'enable HA', 1
   ```

6. Restart both servers.
7. Run the `installhasvss` script on both servers. This script is located in `$SYBASE/$SYBASE_ASE/scripts`.
8. Reestablish companionship:

   ```
   sp_companion [companion_server_name], configure     [,
   with_proxydb]
   ```

Distributed Transaction Management (DTM)

Adaptive Serve automatically prevents SQL commands that are intended to execute inside a distributed transaction from executing outside it.

A distributed, or external, transaction is managed by an external transaction coordinator, such as XA Transaction Manager (TM).

In versions of Adaptive Server earlier than 15.0.3, user applications determined whether an external transaction was rolled back while executing DML commands. If Adaptive Server

implicitly aborted an external transaction without the application's knowledge, DML commands that would normally run inside this transaction might instead be executed inside an implicit transaction started by Adaptive Server. This behavior could result in inconsistent business data.

In versions 15.0.3 and later, Adaptive Server does not allow any DML commands to be executed on the connection attached to the external transaction until the transaction manager sends a detach request. The detach request indicates the end of a batch of commands intended for the external transaction.

In versions 15.0.3 and later, Adaptive Serve automatically prevents SQL commands that are intended to execute inside a distributed transaction from executing outside it. The user application no longer has to check the global variable before every command; when a transaction is implicitly aborted, an error message (3953) appears, saying "Cannot execute the command because the external transaction has been rolled back." This message disappears when a **detach transaction** command is issued.

Adaptive Server Plug-in Updates

The Adaptive Server Plug-in now runs on Sybase Central 6.00, and contains several new features.

The Adaptive Server Plug-in for Sybase Central manages various Adaptive Server Enterprise products. In versions earlier than 15.0.3, the Adaptive Server Plug-in ran on Sybase Central 4.3. In 15.0.3 the Adaptive Server Plug-in runs on Sybase Central 6.00. These features are new to version 15.0.3, Sybase Central 6.00:

- A Search tool helps you find objects displayed by plug-ins.
- The Connection Profile Description, Import, and Export options allow you to add a text description to a profile connection.
- There is better support for Windows Vista.

These features are new to the version 15.0.3 Adaptive Server plug-in.

- You can create objects by selecting the Add icon from a context-sensitive toolbar.
- Stored procedures and SQLJ procedures are located in the Procedures folder.
- Scalar functions, or user-defined functions, are now supported.
- Utilities items are now accessible from the menu on the context-sensitive toolbar.

DBISQL11, which was previously shipped as part of Adaptive Server Plug-in, is now a separate product, version 11.0, and includes these features:

- The number of multiple result sets is no longer limited to 10.
- The login dialog for Adaptive Server now retains and displays the last five connected server names.

- DBISQL11, or interactive SQL, now supports connection favorites, which are similar to connection profiles.
- The SQL statements pane now contains line numbers.
- The Results pane now shows using **select all**, **insert**/**update**/**delete** SQL statements, and sorting and generating, from selected rows.

The Java Interface

Java in Adaptive Server now supports third-party JRE and JVM components such as J2SE.

Adaptive Server version 15.0.3 lets you plug in off-the-shelf Java Runtime environment (JRE) and JVM components, such as J2SE, to Adaptive Server. This Adaptive Server Java framework is called the pluggable component interface (PCI), which includes pluggable component adaptors (PCAs). Any JVM configured for Adaptive Server is called a "plug-in."

The Adaptive Server Java framework allows you to build on the Java solution in Adaptive Server versions 15.0.2 and earlier without losing significant functionality. Any Java applications you developed using Adaptive Server versions earlier than 15.0.3 should run seamlessly with Java applications you create using the framework in versions 15.0.3 and later.

After you configure Adaptive Server to run with the PCI, you can include any standard JVM that supports Java 6 or later. This separates your Java applications from Adaptive Server, allowing you to change or upgrade your Java applications independent of Adaptive Server and to take advantage of new Java functionality as it becomes available.

See *Java in Adaptive Server Enterprise* for a complete description of the new Adaptive Server Java interface.

System Changes in Adaptive Server 15.0.3

Adaptive Server 15.0.3 introduces new and changed functions, system stored procedures, commands, configuration parameters, and system tables.

Functions

Adaptive Server 15.0.3 introduces the **password_random** and **pssinfo** functions.

Function	Description
password_random	Generates a pseudorandom password that satisfies the global password complexity checks defined on Adaptive Server.
pssinfo	Returns information from the process status structure, with an option that retrieves the transaction isolation level of any spid.

See the *Reference Manual: Commands*.

System Stored Procedures

New and changed system stored procedures in Adaptive Server 15.0.3

Table 12. New system stored procedures

Stored procedure	Description
sp_tabsuspectptn	A range-partitioned table on character-based partition keys can become suspect after a sort-order change, and hash-partitioned tables can become suspect after a cross-platform dump load.
sp_jreconfig	This is a Java stored procedure.

Table 13. Changed System Stored Procedures

Procedure	Change
sp_ldapadmin	Supports the new parameters **set_failback_interval** and **reinit_descriptors**.

Procedure	Change
sp_addserver	Supports the **filter** parameter to add a remote server for remote procedure calls.
sp_passwordpolicy	Supports the **validate password options** parameter.
sp_pciconfig	Supports Java in the database.
sp_sysmon	Supports additional counters.

See the *Reference Manual: Procedures*.

Commands

Adaptive Server 15.0.3 introduces changes to the **create table** and **update statistics** commands.

Command	Description of change
create table	The table you create with this option is available only to **BCP IN** and **'alter table unpartition'** operations.
update statistics	Resets the data change counters for global nonclustered indexes.

See the *Reference Manual: Commands*.

Configuration Parameters

New configuration parameters for Adaptive Server 15.0.3

Procedure	Change
enable pci	This is a Java configuration parameter.
maximum nesting level	The maximum nesting level has been increased to 100.

Procedure	Change
mnc_full_index_filter	Prevents Adaptive Server from considering non-covered indexes that do not have a limiting search argument at the server level if there is: • A column in the index • A predicate that does not have a histogram
pci memory size	This is a Java configuration parameter

See the *System Administration Guide: Volume 1*.

Monitoring Tables

New and changed monitoring tables for Adaptive Server 15.0.3

Table 14. New monitoring tables

Table	Description
monSQLRepActivity	Provides statistics for all open objects on DML statements replicated using SQL statement replication.
monSQLRepMisses	Provides statistics for replicated operations for which SQL statement replication was not used. The *threshold*, *querylimitation* and *configuration* columns indicate the number of times that one of these factors prevented SQL statement replication for the object

Table 15. Changed monitoring tables

Monitoring table	Description of change
monSysStatement, *monSysPlanText*, and *monSysSQLText*	The values of the columns *BatchID*, *ContextID*, *ProcedureID*, and *PlanID* have been modified.
monSysStatement	Supports two new columns: *ProcNestLevel* and *StatementNumber*.

See the *Reference Manual: Tables*.

System Tables

Adaptive Server 15.0.3 adds new columns to several system tables.

Table	Changes
sysqueryplans	New columns: • *dbid, int null* • *qpdate, datetime null* • *sprocid, int null* • *hashkey2,int null* • *key1, int null* • *key2, int null* • *key3, int null* • *key4, int null* **Note:** These columns are reserved for future use.
sysprocedures	New column *qp_setting varbinary(6) null* **Note:** This column is reserved for future use.
sysprocesses	New column *clientport* • Displays client port numbers for client processes • Displays 0 for system processes • Datatype: unsigned *smallint*
sysservers	The column *srvnetname* has changed from *varchar(32)* to *varchar(255)*.

See the *Reference Manual: Tables*.

New Features in Adaptive Server 15.0.2

Adaptive Server 15.0.2 introduces many new features and enhancements. They include enhancements to security, encrypted columns, and performance. New features include archive database access, deferred compilation, eager and lazy aggregate processing, and user-defined SQL functions.

Encrypted Columns

Adaptive Server Enterprise 15.0.2 provides enhancements to encrypted columns

The new features:

- Protect data from administrator. You can protect keys and encrypted columns with your own password to ensure privacy of data against the power of the DBO or System Administrator.
- Maintain application transparency using key copies protected by login passwords. That is, you can create key copies and assign them to individual users. Users can encrypt their key copies using their login passwords. Once a key copy is associated with a login password, users do not have to supply the key encryption password when they access data encrypted with the key.
- Provide for key recovery. You can recover access to a key after losing a password. The key owner sets up a recovery key copy, which can later be used to reencrypt the key after losing the password.
- Return a default value for users without decrypt permission. You can create or alter a table to allow **select** statements to return specified default values for users who do not have decrypt permission. This allows you to run existing applications and reports without generating a permission error, while keeping private data secure against unauthorized users. Reports generated by unauthorized users do not reveal the encrypted data.
- Restrict automatic decrypt permissions. When the **restricted decrypt permission** configuration parameter is enabled, the System Security Officer explicitly grants decrypt permission, restricting access to data. When **restricted decrypt permission** is enabled:
  - Table owners are not implicitly granted decrypt permission. The schema owner does not have automatic and implicit access to user data, even in systems that rely on the system encryption password to access the keys.
  - Only users with the sso_role can grant decrypt permission. The **with grant** option is supported for decrypt permission.
  - Implicit access through ownership chains across view and tables or procedures and tables is restricted.

- Adds datatypes. You can encrypt these additional datatypes: *date*, *time*, *datetime*, *smalldatetime*, *money*, *smallmoney*, *big int*, *unsigned big int*, *bit*, *unichar* and *univarchar*.

Archive Database Access

Validate or selectively recover data from a database dump (an “archive”) by making the dump appear as if it is a traditional, read-only database (an "archive database")

Unlike a traditional database, an archive database uses the actual database dump as its main disk storage device, with a minimum amount of traditional storage to represent new or modified pages that result from the recovery of the database dump. A database dump already contains the images of many (if not most) of the database pages, therefore, an archive database can be loaded without having to use Backup Server to transfer pages from the archive to traditional database storage. Consequently, the load is significantly faster than a traditional database.

Archive database access enables a variety of operations to be performed directly on a database dump.

An archive database does not have to be a complete copy of the original database. Depending on the optimization used when dumping the database using **sp_dumpoptimize**, an archive database may be fully populated (every page in the database is in the database dump), or partially populated (only allocated pages are stored in the database dump).

Because the database dump is presented as a read-only database, a database administrator can query it using familiar tools and techniques such as:

- Running database consistency checks on the most recent copy of a dump made from a production database. These checks can be offloaded to a different server to avoid resource contention in the production environment. If resources are not a concern, the archive can be directly checked in the same server in which it was created. Verification on the archive provides the assurance needed prior to performing a restore operation.
- If the integrity of a database dump is in question, loading it into an archive database can be a quick test for success, and therefore a good tool to identify the appropriate database dump that should be used to restore a traditional database.
- Object-level restoration from the database dump. Lost data is recovered using **select into** to copy the to-be-restored rows from the table within the archive database. The **select into** operation can be performed either directly in the server hosting the archive database, or by using Component Integration Services proxy tables if the archive database is available on a different server than that of the object requiring restoration.

In addition, transaction logs can be loaded into an archive database, thereby providing assurance that the same load sequence can be applied when performing a restore operation.

Finding Slow-Running Queries

Adaptive Server 15.0.2 introduces new **set** commands that collect information about slow-running queries.

These parameters for the **set** command enable you to collect diagnostic information about poorly running queries without having to previously enable **showplan** or other investigatory parameters:

- **tracefile** – saves diagnostics to a trace file
- **show_sqltext** – displays SQL text
- **export_options** – retains session settings

Deferred Compilation

Adaptive Server 15.0.2 introduces deferred compiling. Using deferred compilation, the optimizer can compile stored procedural statments that reference real runtime values.

The optimizer can now perform a runtime compilation of procedural statements that reference local variables and temporary tables, so that the query is optimized with real runtime values, instead of magic numbers.

- Adaptive Server uses deferred compilation for queries that reference local variables and parameters in search clauses, queries where a join is used with a temporary table created in the same procedure, and queries where a subquery references a temporary table.
- A statement qualified for deferred compilation is compiled at the first execution of the stored procedure. Those statements that are not executed the first time the stored procedure is invoked, for example statements omitted due to **IF** clauses, are not compiled until a subsequent execution of the stored procedure actually executes these statements.
- Once a conditional statement is compiled, whether at the first or at a subsequent procedure execution, it is integrated into the query plan and is not recompiled.
- Queries that reference procedure parameters previously were compiled and optimized only with the the value of those parameters upon entry to the stored procedure. In versions 15.0.2 and later, these statements are optimized with the parameter values they had when the query was first executed. If the parameter value alters during the course of the stored procedure execution, the current value is used in optimization.
- You can switch off deferred compilation by starting the server with the global switch -T7730.

Case-Insensitive Sort Orders for Chinese and Japanese Character Sets

This section describes case-insensitive sort orders for these Chinese and Japanese character sets:

- EUC-GB
- GB-18030
- CP-936
- EUC-JIS
- SJIS
- DECKANJI

Table 16. Sort orders available for Simplified Chinese and Japanese

Language or script	Character sets	Sort orders
Simplified Chinese	EUC-GB, GB-18030, CP936	General purpose case-insensitive dictionary ordering
Japanese	EUCJIS, SJIS, DECKANJI	General purpose case-insensitive dictionary ordering

Statistical Aggregate Functions

Adaptive Server 15.0.2 introduces statistical aggregate functions to compute variance and standard deviation

Aggregate functions summarize data over a group of rows from the database. The groups are formed using the **group by** clause of the **select** statement.

Simple aggregate functions, such as **sum**, **avg**, **max**, **min**, **count_big**, and **count** are allowed only in the **select** list and in the **having** and **order by** clauses as well as the **compute** clause of a **select** statement. These functions summarize data over a group of rows from the database.

Adaptive Server Enterprise now supports statistical aggregate functions, which permit statistical analysis of numeric data. These functions include **stddev**, **stddev_samp**, **stddev_pop**, **variance**, **var_samp**, and **var_pop**.

These functions, including **stddev** and **variance**, are true aggregate functions in that they can compute values for a group of rows as determined by the query's **group by** clause. As with other basic aggregate functions such as **max** or **min**, their computation ignores null values in the input. Also, regardless of the domain of the expression being analyzed, all

variance and standard deviation computation uses IEEE double-precision floating-point standard.

If the input to any variance or standard deviation function is the empty set, then each function returns as its result a null value. If the input to any variance or standard deviation function is a single value, then each function returns 0 as its result.

Standard Deviation and Variance

Learn about the new statistical aggregate functions and their aliases.

- **stddev_pop** (also **stdevp**) – standard deviation of a population.Computes the population standard deviation of the provided value expression evaluated for each row of the group (if **distinct** was specified, then each row that remains after duplicates have been eliminated), defined as the square root of the population variance.
- **stddev_samp** (also **stdev**, **stddev**) – standard deviation of a sample. Computes the population standard deviation of the provided value expression evaluated for each row of the group (if **distinct** was specified, then each row that remains after duplicates have been eliminated), defined as the square root of the sample variance.
- **var_pop** (also **varp**) – variance of a population. Computes the population variance of value expression evaluated for each row of the group (if **distinct** was specified, then each row that remains after duplicates have been eliminated), defined as the sum of squares of the difference of value expression from the mean of value expression, divided by the number of rows in the group.
- **var_samp** (also **var**, **variance**) – variance of a sample. Computes the sample variance of value expression evaluated for each row of the group (if **distinct** was specified, then each row that remains after duplicates have been eliminated), defined as the sum of squares of the difference from the mean of the value expression, divided by one less than the number of rows in the group.

Eager and Lazy Aggregation

Aggregate processing summarizes large amounts of data with an aggregated value.

Aggregate processing is one of the most useful operations in DBMS environments. It summarizes these values:

- The minimum, maximum, sum, or average value of a column in a specified set of rows
- The count of rows that match a condition
- Other statistical functions

In SQL, aggregate processing is performed using the aggregation functions **min()**, **max()**, **count()**, **sum()**, and **avg()**, and **group by** and **having** clauses. The SQL language implements two aggregate processing types, *vector aggregation* and *scalar aggregation*. A **select-project-join** (SPJ) query illustrates these two types of aggregate processing:

```
select r1, s1
from r, s
where r2 = s2
```

Vector and Scalar Aggregation

Adaptive Server 15.0.2 supports vector and scalar aggregation

In vector aggregation, the SPJ result set is grouped on the **group by** clause expressions, and then the **select** clause aggregation functions are applied to each group. The query produces one result row per group:

```
select r1, sum (s1)
from r, s
where r2 = s2
group by r1
```

In scalar aggregation, there is no **group by** clause and the entire SPJ result set is aggregated, as a single group, by the same **select** clause aggregate functions. The query produces a single result row:

```
select sum (s1)
from r, s
where r2 = s2
```

Improved Performance for Data Insertion

Adaptive Server 15.0.2 optimizes performance of data insertion

These 15.0.2 features can enhance performance for data insertion:

- Fast **bcp** can copy data into tables with nonclustered indexes or triggers, improving Adaptive Server's performance for inserting huge volumes of data.
- Adaptive Server version 15.0.2 includes a separate user log cache (ULC) for the session's temporary database, so multidatabase transactions that include a single user database and the session's temporary database do not require ULC flushes when the users switch between the databases or if all of the following conditions are met:
  - Adaptive Server is currently committing the transaction.
  - All the log records are in the ULC
  - There are no post-commit log records.

  The configuration option, **session tempdb log cache size**, allows you to configure the size of the ULC, helping to determine how often it needs flushing.
- When Adaptive Server splits an index or data page, it moves some rows from the original pages to the newly created page. The operation of moving the rows is not logged. Adaptive Server version 15.0.2 uses asynchronous writes to disk that do not require the server to block as it waits for the write to complete. Adaptive Server version 15.0.2 uses these asynchronous writes automatically and requires no configuration on your part.

- Improved throughput of tempdb transactions
- Adaptive Server version 15.0.2 provides post-commit optimization. The server performs two scans of the log: the first scan looks for data page deallocation and unreserved pages, the second scan looks for log page deallocation. These scans are an internal optimization, transparent to users, and are performed automatically; you cannot switch the scans on or off.

  With post-commit optimization, Adaptive Server remembers the “next” log page (in the backward direction) containing these log records. During the post-commit phase, Adaptive Server moves to the “next” page requiring post-commit work after processing records from a page. In a concurrent environment, where many users log their transactions to *syslogs* at the same time, post-commit optimization can improve the performance of post commit operation by avoiding reads or scans of unnecessary log pages.

Using Asynchronous Writes During a Page Split

When Adaptive Server splits an index or data page, it moves some rows from the original pages to the newly created page. The operation of moving the rows is not logged. Adaptive Server 15.0.2 uses asynchronous writes to ensure consistency.

To ensure both consistency and durability, Adaptive Server must satisfy these conditions:

- Adaptive Server writes the new page to disk before writing the modified page (with the rows removed) to disk. This ensures that Adaptive Server can restore the previous version of the page if the transaction is undone. Adaptive Server can find these rows on the new page and move them back to the old page even if the rows are missing in the old page, and their row contents were not logged.
- The new page reaches the disk before the transaction commits, which ensures that Adaptive Server cannot loose the committed data. If the transaction was committed, Adaptive Server is not required to redo the transaction for the new page, which would be impossible since the movement of the rows was not logged. In the case of undo, the new page’s allocation is backed out; there's no page pre-image to restore.

Previous versions of Adaptive Server ensured these two conditions were met by synchronously writing the new page to disk. However, because the server could block until the synchronous write returned, this caused a degradation in performance.

Adaptive Server version 15.0.2 uses asynchronous writes to disk that satisfy the conditions described above and do not require the server to block as it waits for the write to complete.

Adaptive Server version 15.0.2 uses these asynchronous writes automatically and requires no configuration on your part.

Improving Throughput of tempdb Transactions

Earlier versions of Adaptive Server flushed the data pages and single log records (SLRs) because crash recovery was not supported for *tempdb* or any databases not requiring recovery.

SLRs are log records that force a flush of the user log cache (ULC) to *syslogs* immediately after the record is logged. SLRs are created for OAM modifications, and Adaptive Server creates log records affecting allocation pages in a mixed log and data database as SLRs.

- For regular databases, a ULC containing SLRs is flushed immediately to avoid any undetected deadlocks caused during buffer "pinning". Avoiding a ULC flush for SLRs reduces log semaphore contention, improving the performance.

  A ULC flush avoids the deadlock caused by buffer pinning. Because Adaptive Server does not pin the buffers for databases that do not need recovery, it avoids this deadlock and does not have to flush the ULC for SLRs.
- For databases that require recovery, Adaptive Server flushes dirty pages to disk during the checkpoint. This ensures that, if Adaptive Server crashes, all the committed data is saved to disk. However, for databases which do not require recovery, Adaptive Server supports a runtime rollback, but not a crash recovery. This allows it to avoid flushing dirty data pages at a checkpoint and improves performance.
- Adaptive Server does not support write ahead logging on databases that do not require recovery. Write-ahead logging guarantees that data for committed transactions can be recovered by "redoing" the log (reperforming the transactions listed in the log), and "undoing" the changes done by aborted or rolled back transactions to maintain database consistency. Write-ahead logging is implemented by the "buffer pinning" mechanism. Since Adaptive Server does not ensure write-ahead logging on databases not needing recovery, it does not pin buffers for these databases, so it can skip flushing the log when it commits a transaction.

Post-commit Optimization

Adaptive Server version 15.0.2 performs two scans of the log: the first scan looks for data page deallocation and unreserved pages, the second scan looks for log page deallocation. These scans are an internal optimization, transparent to users, and are performed automatically; you cannot switch the scans on or off.

Previous versions of Adaptive Server:

- Used three scans of the log record after a committed transaction, one each for page deallocation, unreserved pages, and log deallocation.
- Performed the backward scan of log pages using page linkages. Some of the pages did not have log records requiring post-commit work.

With post-commit optimization, Adaptive Server remembers the "next" log page (in the backward direction) containing these log records. During the post-commit phase, Adaptive Server moves to the "next" page requiring post-commit work after processing records from a page. In a concurrent environment, where many users log their transactions to *syslogs* at the

same time, post-commit optimization can improve the performance of post commit operation by avoiding reads or scans of unnecessary log pages.

The optimization does not show up in any diagnostics.

Changes to the Query Processor

These modifications to the query processor describe behavior changes that can affect query plan selection.

Deferred Compilation

The query processor for Adaptive Server version 15.0.2 defers the optimization of statements in a stored procedure until it executes the statement.

Deferring the optimization of statements benefits the query processor because the values for local variables are available for optimization for their respective statements.

Earlier versions of Adaptive Server used default guesses for selectivity estimates on predicates using local variables.

Non-binary Character Set Histogram Interpolation

Adaptive Server version 15.0.2 allows selectivity estimates to have the same accuracy as the binary character set.

In earlier versions of Adaptive Server, only the default binary character set benefited from histogram interpolation, which is used to estimate the selectivity of range predicates. For all other character sets, Adaptive Server made a selectivity estimate of 50 percent for a histogram cell. This typically required Adaptive Server to use a large number of histogram cells for character column histograms to reduce the error associated with this estimate.

Adaptive Server version 15.0.2 allows selectivity estimates to have the same accuracy as the binary character set, without requiring an excessive number of histogram steps. This benefits queries like the following with range predicates:

```
select * from t1 where charcolumn > "LMC0021" and
charcolumn <= "LMC0029"
```

If ranges specified falls into the same histogram cell, Adaptive Server can much more accurately estimate this selectivity.

Expression Histogramming Selectivity Estimates

Adaptive Server version 15.0.2 applies histogramming estimates to single column predicates if the histogram exists on the column.

Earlier versions of Adaptive Server used default “guesses” for selectivity estimates.

Adaptive Server version 15.0.2 applies histogramming estimates to single column predicates if the histogram exists on the column. This results in more accurate row estimates, and improves the join order selection for query plans.

In this example, if the expression is very selective, it may be better to place table *t1* at the beginning of the join order:

```
select * from t1,t2 where substring(t1.charcol, 1, 3)
= "LMC" and t1.a1 = t2.b
```

Viewing Current Optimizer Settings

A new fake table *sysoptions* and a new stored procedure **sp_options** let you view optimizer settings.

To let you easily view optimizer settings, Adaptive Server version 15.0.2 introduces:

- *sysoptions* – a new fake table that stores information about each **set** option, its category, and its current and default settings. *sysoptions* also contains a bitmap containing further detailed information about the option.
- **sp_options** – a new stored procedure that shows option values.

New Security Features

New security features in Adaptive Server 15.0.2

PAM Support in 64-bit Adaptive Server on AIX

Adaptive Server 15.0.1 supports PAMUA on AIX 5.2.

Adaptive Server version 15.0.2 on AIX 5.2 64-bit supports Pluggable Authentication Module-based User Authentication (PAMUA). Contact your support representative to make sure you have the latest patch for PAM available on your IBM host.

To enable PAMUA in 64-bit Adaptive Server 15.0.2 on AIX 5.2, you must supply the PAM module in `/etc/pam.conf` file. For example:

```
ase auth required /usr/lib/security/pam_aix
```

Global Login Triggers Set Automatically

For Adaptive Server version 15.0.2 and later, any exportable option set or unset in a login trigger takes effect in the login process when the server starts.

Adaptive Server versions 15.0.1, 12.5.4, and earlier required that you start Adaptive Server with trace flag 4073 to enable the options for a login trigger.

Any exportable option set or unset in a login trigger now takes effect in the login process when the server starts

To disable the new behavior, execute **set export_options off** inside the login trigger.

SSL Support

Adaptive Server 15.0.2 supports SSL on Windows 2003 x64 Enterprise Edition.

Adaptive Server version 15.0.2 supports Secure Sockets Layer (SSL) on Windows 2003 X64 Enterprise Edition. Windows 2003 X64 implements SSL functionality using OpenSSL libraries.

SSL functionality is the same as Adaptive Servers on other platforms. It supports the same cipher suites and pre-defined lists as shown in the *System Administration Guide*. OpenSSL libraries for Adaptive Server on Windows 2003 X64 communicates with SSLPlus clients and servers using the same cipher suite names used in SSLPlus. The cipher suite names conform to the Transport Layer Security (TLS). TLS is an enhanced version of SSL 3.0, and is compatible with the SSL version 3.0 Cipher Suites.

Improved Password Security

Adaptive Server 15.0.2 improves password security.

Adaptive Server 15.0.2 adds a higher level of security to the existing password protection mechanisms through:

- Stronger encryption for passwords sent across the network
- Stronger encryption for passwords stored in *syslogins* (on disk) and in memory
- Managing login and password use with new time stamp data and additional account reporting

Auditing Enhancements

Adaptive Server version 15.0.2 introduces two additional auditing security features.

Hiding System Stored Procedure and Command Password Parameters

System stored procedure and command passwords can be replaced with asterisks in audit records.

When auditing is configured and enabled, and the **sp_audit** option **'cmdtext'** is set, system stored procedure and command password parameters are replaced with a fixed length string of asterisks in the audit records contained in the audit logs.

For example, execute:

```
sp_password 'oldpassword', 'newpassword'
```

When auditing is enabled and **sp_audit cmdtext** is set, the results in output are similar to:

```
sp_password '******', '******'
```

This protects passwords from being seen by other with access to the audit log.

Monitoring Failed Login Attempts

Monitor failed logins attempts with the **login_locked** audit option.

The new audit option **login_locked** and the event **Locked Login (value 112)** record when a login account is locked due to exceeding the configured number of failed login attempts. This event is enabled when audit option **login_locked** is set. To set **login_locked**, enter:

```
sp_audit "login_locked","all","all","ON"
```

If the audit tables are full and the event cannot be logged, a message with the information is sent to the errorlog.

The hostname and network IP address are included in the audit record. Monitoring the audit logs for the **Locked Login** event (112) helps to identify attacks on login accounts.

High Availability Considerations

The Adaptive Server 15.0.2 password security changes affect High Availability (HA).

HA configuration

The primary and companion servers must have equivalent "**allow password downgrade**" values before they are configured for HA. A new quorum attribute "**allow password downgrade**" has been added to check whether the value of "**allow password downgrade**" is same on both primary and secondary servers. This HA advisory check succeeds when the value for the quorum attribute is the same, and HA advisory check fails when the values are different.

Changed password behavior in syslogins with HA

After upgrading to Adaptive Server 15.0.2 and successful configuring HA, on the first connection to the primary server, the password of the user login is updated on both the primary and companion servers. This synchronizes the login password on primary and companion with the same on-disk encryption format. This is done to avoid password reset or locking when the "**allow password downgrade**" period ends as described in **sp_passwordpolicy** and with password downgrade to earlier Adaptive Server 15.0 versions with **sp_downgrade**. By synchronizing the password encryption format, the login passwords can continue to be used without being reset or locked by **sp_passwordpolicy** or **sp_downgrade**.

Installing and Editing Monitoring Tables

Adaptive Server version 15.0.2 includes new installation and editing features for monitoring tables.

- Installing monitoring table - you no longer need to run the `installmontables` script to install the monitoring tables.

Versions of Adaptive Server earlier than 15.0.2 required you to run the `installmontables` script to install the monitoring tables. Adaptive Server version 15.0.2 includes the monitor tables installation in the `installmaster` script.

- Remotely accessing and editing monitoring tables - Sybase provides `installmontables` as a sample script showing how to remotely access monitoring tables. Run `installmontables` to view the instructions for editing.

Monitoring Tables for the Statement Cache

Two new monitoring tables let you analyze the contents of the statement cache.

The Adaptive Server statement cache stores SQL text of ad-hoc **update**, **delete** and **select** statements and other statements likely to be reused. When the statement cache is enabled, these statements are converted into lightweight procedures and their plans are saved for reuse. When a new statement is issued, Adaptive Server searches the statement cache for a plan to reuse. If Adaptive Server finds a plan to reuse, it avoids recompiling the statement, leading to performance enhancements.

The introduction of literal parameterization in Adaptive Server version 15.0.1 allows it to recognize queries that are the same except for differences in literal values, saving recompiling costs while using statement cache. In addition to performance benefits, literal parameterization leads to enormous space reduction while storing the metrics and statements in the cache.

Adaptive Server version 15.0.2 introduces two new monitoring tables that allow you to easily analyze the contents of the statement cache:

- *monStatementCache* provides a summary snapshot of the statement cache.
- *monCachedStatement* shows detailed information about each statement cached

The columns in each table allow two attributes, “counter” if the column has a counter value, and “reset” if the column can be reset using mechanisms like **sp_sysmon**.

Row-Level Locking for System Tables

Adaptive Server version 15.0.2 uses row-level locking on system tables to enhance performance.

Versions of Adaptive Server earlier than 15.0.2 used exclusive table locks on system tables while executing data definition language (DDL) and utility commands. The set of system tables Adaptive Server locked depended on the type of DDL operation you executed. If another DDL running concurrently tried to take a conflicting exclusive table lock on the same system table, this DDL had to wait to acquire the lock on any system catalogs. These DDL operations were executed serially.

This methodology impeded performance in temporary databases, where their DDL activity is very high also, and consequently their catalog contention is very high. This limited the Adaptive Server throughput for applications using temporary tables.

Adaptive Server version 15.0.2 uses row-level locking to resolve these issues:

- System-table contention, caused a bottleneck for many DDLs and utilities.
- *tempdb* contention. Because the system tables are locked at the row level, Adaptive Server 15.0.2 eliminates *tempdb* contention.
- Shared or exclusive table-level locks while executing DDLs and utilities. Earlier versions converted most system tables to data-only locking (DOL), but still created shared or exclusive table-level locks while executing DDLs and utilities. Using row-level locks for system tables eliminates this contention.

  Adaptive Server sets intent locks on catalogs only, which removes potential contention (An intent lock indicates that page-level or row-level locks are currently held on a table.).
- DDLs and utilities blocking each other. Adaptive Server 15.0.2 allows DDLs and utilities to run in parallel.

Earlier versions of Adaptive Server used table locks to achieve system catalog synchronization. Adaptive Server 15.0.2 uses intent locks for table-level synchronization and row locks for row-level synchronization. Earlier releases of Adaptive Server locked the entire system catalog while performing operations on the object, so a single lock request was made. However, Adaptive Server version 15.0.2 requests locks for all applicable rows while performing operations on the object if there are multiple rows corresponding to an object in a system catalog.

This change means that Adaptive Server 15.0.2 requests more locks to perform the same operation than earlier releases, and increases the number of lock resources the system needs. Consequently, you may need to change the **number of locks** configuration option after you upgrade Adaptive Server.

The xmltable() Function

xmltable() creates a SQL table from elements in an XML table.

xmltable() extracts a sequence of multivalued elements from an XML document, and assembles a SQL table of those elements. A single call to **xmltable()** replaces a Transact-SQL loop that performs multiple calls to **xmlextract()** on each iteration. **xmltable()** is invoked as a derived table (a parenthesized subquery specified in the **from** clause of a different SQL query). Calling **xmltable()** is equivalent to executing a single **xmlextract()** expression for each row of the table generated by **xmltable()**.

xmltable() is a generalization of **xmlextract()**. Both functions return data extracted from an XML document that is an argument in the function. The differences are:

- **xmlextract()** returns the data identified by a single XPath query.

- **xmltable()** extracts the sequence, or row pattern, of the data identified by an XPath query, and extracts from each element of that sequence the data identified by a list of other XPath queries, the column patterns. It returns all the data in a SQL table.

Relocated Joins

Relocation joins permits joins between local and remote tables to locate to remote server.

Adaptive Server version 15.0.2 introduces relocated joins, which allow joins between local and remote tables to be relocated to a remote server. The remote system executes the join using a dynamically created proxy table referring back to the local table. With the remote system executing the join, a significant amount of network traffic is avoided.

User-Defined SQL Functions

Adaptive Server 15.0.2 introduces user-defined SQL functions.

Use **create function** to create user-defined functions, and **drop function** to remove a user-defined function.

You can include these elements in a scalar function:

- **declare** statements to define data variables and cursors that are local to the function.
- Assigned values to objects local to the function (for example, assigning values to scalar and variables local to a table with **select** or **set** commands).
- Cursor operations that reference local cursors that are declared, opened, closed, and deallocated in the function.
- Control-of-flow statements.
- **set** options (only valid in the scope of the function).

Adaptive Server does not allow **fetch** statements in a scalar function that return data to the client. You cannot include :

- **select** or **fetch** statements that returns data to the client.
- **insert**, **update**, or **delete** statements.
- Utility commands, such as **dbcc**, **dump** and **load** commands.
- **print** statements
- Statement that references **rand**, **rand2**, **getdate**, or **newid**.

You can include **select** or **fetch** statements that assign values only to local variable.

instead of Triggers

Use **instead of** triggers to override default triggering actions.

Views are commonly used to separate logical database schema from physical schema. **instead of** triggers can be defined on a view to replace the standard action of an **update**, **insert**, or **delete** statement. The **instead of** trigger allows all views, including those that are not otherwise updatable, to be updated.

instead of triggers are special stored procedures that override the default action of a triggering statement (**insert**, **update**, and **delete**), and perform user-defined actions.

The **instead of** trigger, like the **for** trigger, executes each time a data modification statement executes on a specific view. A for trigger fires after an **insert**/**update**/**delete** statement on a table, and is sometimes called an **after** trigger. A single **instead of** trigger can apply to one specific triggering action:

```
instead of update
```

It can also apply to multiple actions, in which the same trigger executes all the actions listed:

```
instead of insert,update,delete
```

Like **for** triggers, **instead of** triggers use the logical **inserted** and **deleted** tables to store modified records while the trigger is active. Each column in these tables maps directly to a column in the base view referenced in the trigger. For example, if a view named V1 contains columns named C1, C2, C3, and C4, the **inserted** and **deleted** tables contain the values for all four columns, even if the trigger modifies only columns C1 and C3. Adaptive Server automatically creates and manages the **inserted** and **deleted** tables as memory-resident objects.

instead of triggers allow views to support updates, and allow implementation of code logic that requires rejecting parts of a batch, while allowing other parts to succeed.

An **instead of** trigger is fired only once per data modification statement. A complex query containing a **while** loop may repeat an **update** or **insert** statement many times, firing the **instead of** trigger each time.

System Changes in Adaptive Server 15.0.2

Changes to commands, functions, utilities, system procedures, system tables, configuration parameters, functions, and global variables in Adaptive Server 15.0.2

Trace Flags

New trace flags in Adaptive Server 15.0.2

- 15340 enables server wide, no matter advanced_aggregation
- 15341 disables server wide, no matter advanced_aggregation

Commands

Changed commands in Adaptive Server 15.0.2

Table 17. Changed commands

Command	Description of change
disk init, **disk reinit**	Display a warning message if you attempt to create a block device on a platform that Sybase recommends that you not use block device. Sybase recommends that you use block devices as a database device only on the HP-UX, Windows, and Linux platforms.
create proxy table, create table at remote server, and **alter table.**	Do not support SQL UDF.
disk init, **disk resize**	When *skip_alloc* is set to be true, it allows users to avoid initialization of pages with zeros. The default of *skip_alloc* is false. It is supported for devices created on non-Windows file systems and on Windows raw systems.
dump transaction	**dump transaction** can now include **string** and **char_variable** parameters. However, you cannot supply string and **char_variable** options to execute **begin transaction**, **commit**, **connect to**, **declare cursor**, **rollback**, **dbcc**, **use** or nested **execute** commands.

See the *Reference Manual: Commands*.

Changes to the set Command

Adaptive Server 15.0.2 introduces several changes to the **set** command.

set command change
set advanced_aggregation enables and disables advanced aggregation at the session level.
set switch allows you to set trace flags and switch names locally and server-wide.
The compile-time behavior has changed for some abstract plan **set** parameters when you use them to create stored procedures or run them in Transact-SQL batches.
The **set** command can include **string** and **char_variable** parameters.

Utilities

Adaptive Server 15.0.2 supports enhancements to the **ddlgen** utility.

Change	Description
ddlgen prompts for password.	In version prior to 15.0.2, failing to include the -P password parameter returned an error. **ddlgen** now prompts for the password.
ddlgen supports both pre-15.0.2 and 15.0.2 encryption.	Sample syntax: `ddlgen -Usa -P -Sserver -TEK -NSampleKeysDB.dbo.ssn_key`
ddlgen without the **-XOD** flag	Two things can happen: • If users do not specify a password when the encryption key is created, **ddlgen** generates DDL with no password. • If users specify a password when the encryption key is created, **ddlgen** generates a default password.
ddlgen with the **-XOD** flag	Generates a system encryption password.
Key copy support	**ddlgen** can generate DDL for key copies and for the base key.

See the *Utility Guide*.

System Stored Procedures

New and changed system procedures in Adaptive Server 15.0.2

Table 18. New system stored procedures

System stored procedure	Description
sp_downgrade	Validates readiness for downgrade to an earlier 15.0.x release. Also downgrades the system catalog changes Adaptive Server 15.0.2 modified.
sp_spaceusage	Reports the space usage for a table, index, or transaction log and estimates the amount of fragmentation for tables and indexes in a database. The estimates are computed using an average row-length for data and index rows, and the number of rows in a table. You can archive the space usage and fragmentation data for future reporting and trends analysis. **sp_spaceusage** supports a number of actions, including **help**, **display**, **archive** and **report**, to indicate the current Adaptive Server space usage.

Table 19. Changed system stored procedures

System stored procedure	Description of change
sp_autoformat	Now accepts columns of datatypes *int* (*smallint*, *bigint*, *tinyint*, *unsigned int*), *numeric*, *money*, *date/time*, and *float*, *real*, and *double* precision.
sp_changedbowner	Changes the owner of a database. You can now execute it with either sa_role or sso_role privileges. The owner of thresholds for that database is also changed to the specified user.
sp_checksource	Encrypts the text of user-defined functions.
sp_configure	Now displays non-default value settings.
sp_depends	Checks for any object dependencies for user-defined functions.
sp_deviceattr	Displays a warning message if the dsync option is disabled for a database device on a file system.

System stored procedure	Description of change

sp_displaylogin	includes these changes:
	• supports both a wildcard expression and a server user ID, and displays matching logins:
	`sp_displaylogin ['user_id' | '[loginame | wildcard]'`
	• **user_id** – user ID (*suid*) of the user whose login you are displaying.
	• **wildcard** – wildcard character used for search purposes.
	Displays the login account for the user with a suid of 56:
	`sp_displaylogin '56'`
	Displays the login account information for all users whose logins begin with "st":
	`sp_displaylogin 'st%'`

sp_droplogin	When **sp_droplogin** is unable to drop a login due to the existence—in any database—of a user in *sysusers* referencing the login **suid**, the names of databases in which the references are found are now displayed in the error message.
sp_help	Displays information about user-defined functions.
sp_helpdevice	The *description* column of **sp_helpdevice** displays information about the device type. The device type is one of: `raw device`, `block device`, or `file system device`.
sp_helprotect	The new option, **permission_name**, in **sp_helprotect** provides information (grantor name, grantee name, table/column name, and grantability) for any specific permission granted in a given database.
sp_hidetext	Encrypts the text for user-defined functions.
sp_locklogin	The **lock** option to **sp_locklogin**, when used with a value for **number of inactive days**, locks inactive accounts that have not authenticated within that period. The following example locks all login accounts that have not authenticated within the past 60 days.
sp_modifylogin account	A new value for the **'max failed_logins'** option indicates that the failed login count in the *syslogins* column login count, is updated whenever an authentication failure occurs, but that the account is not locked.

System stored procedure	Description of change
sp_modifystats	Allows the System Administrator, or any user with permission to execute the procedure and update statistics on the target table, to modify the density values of columns in *sysstatistics*
sp_monitorconfig	Enhanced to create a table to hold the result set, if the user passes a table name for **result_tabl_name** that does not already exist.
sp_passwordpolicy	• The **set** and **clear** commands in **sp_passwordpolicy** are now audited, through audit event 115, "Password Administration." • Additional syntax: `sp_passwordpolicy "enable last login updates", "allow password downgrade" "regenerate keypair", "expire login passwords", "[login_name | wildcard]" "expire role passwords", "[role_name | wildcard]" "expire stale login passwords", "datetime" "expire stale role passwords", "datetime" "maximum failed logins", -1`
sp_fixindex	Now works on a set of indexes rather than on a single index. **sp_fixindex** rebuilds the data layer if the target table has a placement or clustered index (it reclaims the unused space in the data layer while working on the placement or clustered index of a system table).
sp_sendmsg	In previous releases, the maximum length for a message sent with this system procedure was 255 characters. For Adaptive Server release 15.0.2, the maximum length of a **sp_sendmsg** message is 4096 characters.
sp_who	A new column, *tempdbname*, displays temporary database names of all active sessions.
sp_helptext	Reports the text of user-defined functions, and introduces the **numlines** and **printops** parameters.
sp_ldapadmin	Introduces new parameters: **set_max_ldapua_desc**, **set_num_retries**, and **set_log_interval**.
sp_monitor	Enhances the **event** and **help** parameters.
sp_tempdb	Introduces the **show** and **who** parameters.

See the *Reference Manual: Procedures*.

System Tables

New and changed system tables in Adaptive Server 15.0.2.

Table 20. New system tables

System table	Description
sysoptions	*sysoptions* is a fake table queried by **sp_options**. The columns names are case-sensitive.

Table 21. Changed system tables

System table	Description of change
sysquerymetrics	Adaptive Server version 15.0.2 increases the number of metrics shared among user IDs. The change reduces the number of entries in *sysquerymetrics* (a view of *sysqueryplans*), and automatically aggregates the metrics for identical queries across different user IDs.
syscolumns	Adds a status bit to the *status2* column that indicates an encrypted column has a *decrypt_default* attached to it: 0x00001000 (4096) – column has a decrypt default
sysobjects	The *type* column of *sysobjects* includes an entry of "*DD*" for each object that has a decrypt default.

sysaudits	Changes include:
	• The *Alter Encryption Key* audit event name is changed to *AEK As/Not Default*
	• Adaptive Server release 15.0.2 supports these audit events and numbers:
	  • 118 – AEK Modify Encryption
	  • 119 – AEK Add Encryption
	  • 120 – AEK Drop Encryption
	  • 121 – AEK Modify Owner
	  • 122 – AEK Key Recovery

System table	Description of change

sysattributes	*sysattributes* includes these changes:
	• Adds two classes:
	  • Class 31 *allow password downgrade* – when set to 1, *allow password downgrade* enables special handling of login passwords for compatibility with Adaptive Server release 15.0 and earlier.
	  • Class 32 *enable last login updates* – when set to 1, *enable last login updates* enables system tables to store the date of the last login.
	• *sysattributes* includes information about default decrypt. These are the changes to the columns:
	  • *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* – includes the decrypt default ID.
	  • *object_info_1* – includes the table ID for a table whose encrypted column defines the decrypt default.
	  • *object_info2* – specifies the **colid** of the encrypted column that includes the decrypt default.

System table	Description of change

sysencryptkeys	Changes to *sysencryptkeys* includes
	• New types:
	  • *EK_KEYCOPY* – 0x0010,
	  • *EK_KEYBASE* – 0x0020
	  • *EK_RECOVERY* – 0x0040
	• New status bits:
	  • *EK_KEYRECOVERY*(0x00000004) – keys encrypted for lost password protection.
	  • *EK_LOGINACCESS*(0x00000008) – key encrypted for login access
	  • *EK_LOGINPASS* (0x00000010) – key encrypted with login password
	  • *EK_USERPWD*(0x00000100) – keys encrypted with user-encryption passwords
	• Changes to the description for the *uid* column – user access or key recovery row. *uid* contains the user ID (*uid*) identifying the database user associated with current row.
	Previous versions of Adaptive Server did not use the *uid* column.

syslogins	In previous releases the maximum length of the *password* column was 30 bytes. In Adaptive Server release 15.0.2, the maximum length of the *password* column is 128 bytes. New columns are: *lastlogindate*, *crdate*, *locksuid*, *lockreason*, and *lockdate.*
syssrvroles	In previous releases the maximum length of the *password* column was 30 bytes. In Adaptive Server release 15.0.2, the maximum length of the *password* column is 128 bytes.

See the *Reference Manual: Tables*.

Configuration Parameters

New and changed configuration parameters in Adaptive Server 15.0.2

Table 22. New configuration parameters

Configuration parameter	Description
enable merge join	Enables or disables merge joins at the server level.
cost of a logical io	Specifies the cost of a single logical IO.
cost of a physical io	Specifies the cost of a single physical IO.
cost of a cpu unit	Specifies the cost of a single CPU operation.
enable encrypted columns	Enables and disables the encrypted columns feature when encrypted columns is licensed.
max online q engines	Required for MQ series, max online q engines specifies the maximum number of Q engines allowed online.
metrics elap max	If the elapsed time of the query is less than the value of metrics elap max, then the metrics associated with the query are not written to the system tables. enable metrics capture must be on.
metrics exec max	If the execution time of the query is less than the value of metrics exec max, then the metrics associated with the query are not written to the system tables. enable metrics capture must be on.
metrics lio max	If the logical IO time of the query is less than the value of metrics lio max, then the metrics associated with the query are not written to the system tables. enable metrics capture must be on.
metrics pio max	If the physical IO time of the query is less than the value of metrics pio max, then the metrics associated with the query are not written to the system tables. enable metrics capture must be on.
min pages for parallel scan	Lets you scan larger queries in parallel.
not password encryption read	Requires all incoming login authentication requests using Adaptive Server authentication to encrypt the login password when transmitted on the network.

Configuration parameter	Description
number of Q engines at startup	Required for MQ series, specifies the maximum number of Q engines you can have online.
prod-consumer overlap factor	Affects optimization. Adaptive Server changes the group by algorithm, and you cannot use **set statistics IO** with parallel plans.
send doneinproc tokens	Replaces the **dbcc tune** option **doneinproc**.

Table 23. Changed configuration parameters

Configuration parameter	Description of change
max async i/os per engine	The default value has changed from 2147483647 to 1024.
maximum failed logins	The new -1 value indicates that the failed login count in the *syslogins* column *logincount* is updated whenever an authentical failure occurs, but that the account is not locked.
print deadlock information	Adds a new parameter value of 2, which lets you print a summary of deadlock information to the error log.

See the *Reference Manual: Tables*.

Functions

New and changed functions in Adaptive Server 15.0.2

Table 24. New functions

Function	Description
authmec()	Returns the authentication method used for a logged-in server process ID session.
index_name()	Returns the name of an index, when you specify the ID of the index and the database, and the object on which the index is defined.
hashbytes()	Produces a fixed-length, hash value expression.

Table 25. Changed functions

Function	Description of change
used_pages()	In all-pages-locked tables with clustered indexes, **used_pages()**is now passed only the used pages in the data layer, for a value of **indid = 0**. When **indid = 1** is passed, the used pages at the data layer and at the clustered index layer are returned.
When a function is created, Adaptive Server checks to see if it is a SQL UDF or a SQLJ UDF. If it is a SQLJ UDF, Adaptive Server checks for "sa" permissions. If it is a SQL function, Adaptive Server checks for **create function** privileges.	

Global Variables

Adaptive Server 15.0.2 introduces the *@@lastlogindate* global variable.

Table 26. New global variables

Global variable	Description
@@lastlogindate	Global T-SQL variable **@@lastlogindate** is available to each user login session. A *datetime* datatype, its value is the *lastlogindate* column for the login account before the current session was established. This variable is specific to each login session and can be used by that session to determine the previous login to the account. If the account has not been used previously or "**sp_passwordpolicy 'set', enable last login updates**" is 0, then the value of **@@lastlogindate** is NULL.

Table 27. Changed global variables

Global variable	Description of change
@@opttimeoutlimit	Previous version of Adaptive Server user documenation listed **@@opttimeout** as a server global variable that displays the current optimization timeout limit for query optimization. This is incorrect. The actual name of the global variable that displays the current optimization timeout limit for query optimization is **@@opttimeoutlimit**

New Features in Adaptive Server 15.0.1

Adaptive Server 15.0.1 introduces several enhancements to abstract plans and lets you automatically convert literal values in SQL queries to parameter descriptions.

Changes to Abstract Plans

Adaptive Server 15.0.1 provides enhancements to abstract plans.

The enhancements include:

- The abstract plan syntax has been extended to allow several query level setting that were previously available only at the session level.
- Adaptive Server accepts both the h_join and hash-join keywords in the extended abstract plan syntax.
- The **set** command supports the *opt criteria* parameter to turn on and off the current optimization goal setting.

New Query-Level Settings

The abstract plan syntax, used by Adaptive Server Enterprise to force the query plan chosen by the optimizer, has been extended to allow several query level setting that were previously available only at the session level.

The optimization criteria are handled at the session level by the following **set** statements:

```
set
    nl_join|merge_join|hash_join|..
    0|1
```

The **use ...** abstract plan syntax has been extended to accept any number of use forms before the abstract plan derived table. Previously, the optgoal and opttimeout could not be in the same abstract plan with a derived table. For example this statement would need to be separate from a optgoal statement in a query:

```
select ...
    plan
    "(use opttimeoutlimit 10) (i_scan r)"
```

However, with Adaptive Server 15.0.1, you can include several statements in the same abstract plan in two ways:

- By using several **use** statements, for example:

  ```
  select ...
         plan
         "(use optgoal allrows_dss)
             (use nl_join off) (...)"
  ```

- By placing several items within one **use** form, for example:

```
select ...
       plan
       "(use (optgoal allrows_dss) (nl_join off)) (...)"
```

At the query level, use the optimization goal (**opt_goal**) or timeout (**opttimeout**) setting with the **use ...** abstract plan syntax. At the session level, use these settings with the **set plan ...** syntax:

- Optimization goal
- Optimization timeout

Operator Name Alignment for the Abstract Plan and the Optimizer Criteria

Adaptive Server accepts both the h_join and hash_join keywords in abstract plans.

The names of algorithms differ between their usage in abstract plans and in the **set** command. For example, a hash join is called h_join in abstract plans, but is called hash_join in the **set** command. Adaptive Server accepts both keywords in the extended abstract plan syntax. For example:

```
select ...
    plan
    "(h_join (t_scan r) (t_scan s))"
```

is equivalent to:

```
select ...
    plan
    "(hash_join (t_scan r) (t_scan s))"
```

and:

```
select ...
    plan
    "(use h_join on)"
```

and:

```
select ...
    plan
    "(use hash_join on)"
```

When a table abstract plan is present, it takes precedence:

```
select
from r, s, t
...
    plan
        "(use hash_join off)(h_join (t_scan r)
            (t_scan s))"
```

The query uses the hash_join for **r** and **s** scans; but for the join with **t** it does not use hash_join as specified by the use abstract plan form, since it was not specified in the table abstract plan.

Extending the Optimizer Criteria Set Syntax

The **set** command supports the *opt criteria* parameter to turn on and off the current optimization goal setting.

The **set** <*opt criteria*> statement, with a 0 or 1 syntax, accepts on/off/default, where default indicates that you are using the current optimization goal setting for this optimization criteria.

See the *Reference Manual: Commands*).

Literal Parameterization

Adaptive Server version 15.0.1 lets you automatically convert literal values in SQL queries to parameter descriptions (similar to variables).

In earlier versions of Adaptive Server, two queries that were identical except for one or more literal values resulted in the statement cache storing two separate query plans, or two additional rows in *sysqueryplans*. For example, the query plans for these queries were stored separately, even though they are almost identical:

```
select count(*) from titles where total_sales > 100
select count(*) from titles where total_sales > 200
```

Adaptive Server version 15.0.1 allows you to automatically convert literal values in SQL queries to parameter descriptions (similar to variables). A new **sp_configure** option supports this feature, which is called **enable literal autoparam**.

To enable or disable **enable literal autoparam** server-wide, use:

```
sp_configure "enable literal autoparam", [0 | 1]
```

Where 1 automatically converts literals to parameters, and 0 disables the feature. The default is 1.

You can set literal parameterization at the session level with the **set** command:

```
set literal_autoparam [off | on]
```

System Changes in Adaptive Server 15.0.1

Functions

Adaptive Server 15.0.1 introduces three new functions.

Function	Description
isdate	Determines whether an input expression is a valid *datetime* value.
isnumeric	Determines if an expression is a valid *numeric* datatype.
partition_object_id	Displays the object ID for a specified partition ID and database ID

See the *Reference Manual: Commands*.

Configuration Parameters

New and changed configuration parameters for Adaptive Server 15.0.1

Table 28. New configuration parameters

Configuration parameter	Description
startup delay	Controls when RepAgent is started during the server start. By default, RepAgent starts at the same time as Adaptive Server. Adaptive Server writes a message to the error log stating the wait time.
enable literal autoparam	Enables and disables literal server-wide parameterization.

Configuration parameter	Description
cis idle connection timeout.	configures Adaptive Server to check for CIS connections to any remote server that have been unused longer than the specified number of seconds. Adaptive Server deletes the unused connections and reallocates their resources. Although the number you specify is in seconds, the housekeeper task wakes up at most once a minute, so idle connections may be idle for much longer than the configured value. Adaptive Server does not drop idle connections if a transaction is active on the connection, and reestablishes the connection automatically if the user executes any command that accesses the connection.
sproc optimize timeout limit	Specifies the amount of time Adaptive Server can spend optimizing a system procedure as a fraction of the estimated execution time.

Table 29. Changed configuration parameters

Configuration parameter	Description of change
optimization timeout limit	The range of values available for **optimization timeout limit** has changed. With version 15.0.1, it is 0 - 1000. A value of 0 indicates no optimization timeout.
max parallel degree	If **max parallel degree** is set to 1 (enabled), Adaptive Server forces serial query execution and the optimizer may select plans with a higher parallel degree than if it is disabled.
number of worker processes	If you have not configured **number of worker processes** for a sufficient number of threads from the worker thread pool, Adaptive Server adjusts query plans at runtime to use fewer worker threads. If Adaptive Server cannot adjust the queries at run-time, the queries recompile serially. However, **alter table** and **execute immediate** commands are aborted if they do not have sufficient worker threads.

See the *System Administration Guide: Volume 1*.

Commands

Adaptive Server 15.0.1 introduces syntax and other changes to **alter table**, **create index**,**create existing table** , **update statistics**, and the **set** command.

Table 30. Changed commands

Table	Description of change
alter table	Use the **alter table** command to drop one or more list or range partitions. You cannot use **alter table** to drop a hash or round-robin partition. The syntax is: `alter table table_name drop partition partition_name [, partition_name]...`
create index	When you create a unique local index on range-, list-, and hash-partitioned tables, the index key list is a superset of the partition-key list.
create existing table	Includes syntax to determine whether an RPC uses the current or a separate connection: `create existing table (<column_list>) EXTERNAL [non_transactional |transactional] PROCEDURE at 'location'` • `non_transactional` – a separate connection is used to execute the RPC. • `transactional` – the existing connection is used to execute the RPC. The default behavior is transactional.
update statistics	Adaptive Server 15.0.1 adds the ability to run **update statistics** on a global index. `update table statistics table_name [partition data_partition_name] [index_name [partition index_partition_name]]` Because running **update table statistics** incurs the I/O cost of running **update statistics**, use **update statistics** to generate both column and table statistics. You can create, and then drop, a global index to generate global statistics.

Table 31. New set command options

New set command options	Description
set literal_autoparam on|off	Enables and disables literal parameterization at the session level.
set opttimeoutlimit	The range of values for **opttimeoutlimit** has been changed to 0 – 4000, with 0 indicating no optimization limit.
set index_union on | off	When enabled, **set index_union** limits the scan of a table with an **or** clause. Index unions (also known as an **or** strategy) are used for queries that contain **or** clauses. For example: `select * from titleauthor where au_id = "409-56-7008" or title_id = "PC8888"` If you have enabled **index_union**, this example uses an index on *au_id* to find the row IDs (RIDs) of all *titleauthor* tuples with `au_id = "409-56-7008"`, and uses an index on *title_id* to find the RIDs of all *titleauthor* tuples with `title_id = "PC8888"`. Adaptive Server then performs a union on all RIDs to eliminate duplicates. The resulting RIDs are joined with a **RidJoin** to access the data tuples. If **index_union** is disabled, Adaptive Server does not use an index union strategy in a query to limit the table scan. Instead, it uses other access paths on the table (in the example above, it would use a table scan for table *titleauthor*), and applies the **or** clause as a filter in the scan operator.

See the *Reference Manual: Commands*.

Monitoring Tables

Adaptive Server 15.0.1 introduces two new monitoring tables: *monProcedureCacheMemoryUsage* and *monProcedureCacheModuleUsage.*

Monitoring table	Description
monProcedureCacheMemoryUsage	Has one row for each procedure cache allocator. An allocator is identified by an allocator ID, which is internal to Adaptive Server.
monProcedureCacheModuleUsage.	Has 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.

See the *Performance and Tuning Guide: Monitoring and Analyzing* and the *Reference Manual: Tables*.

New Features in Adaptive Server 15.0

Adaptive Server 15.0 introduces support for data partitions, row-locked system catalogs, an enhanced query processor, large identifiers, computed columns scollable cursors, new datatype support, XML enhancements, support for interactive SQL, enhancements to SySAM and the Adaptive Server Plug-in, user-defined Web service support, and security enhancements.

Partition Support

Partitioning is useful in managing large tables and indexes by dividing them into smaller, more manageable pieces. Partitions, like a large-scale index, provide faster and easier access to data.

Partitions are database objects and can be managed independently. You can, for example load data, and **create index** cannot be done at a partition level.. Yet partitions are transparent to the end user, who can select, insert, and delete data using the same commands whether the table is partitioned or not.

Adaptive Server 15.0 supports horizontal partitioning, in which a selection of table rows can be distributed among partitions on different disk devices. Individual table or index rows are assigned to a partition according to a semantic or to a round-robin partitioning strategy.

Semantic partitioning strategies use the data values in specified, key columns in each row to determine the partition assignment of that row. The round-robin partitioning strategy assigns rows randomly without reference to data values.

Partitioning strategies are:

- *Hash partitioning* (semantic) – a system-supplied hash function determines the partition assignment for each row.
- *List partitioning* (semantic) – values in key columns are compared with sets of user-supplied values specific to each partition. Exact matches determine the partition assignment.
- *Range partitioning* (semantic) – values in key columns are compared with a user-supplied set of upper and lower bounds associated with each partition. Key column values falling within the stated bounds determine the partition assignment.
- *Round-robin partitioning* – rows are assigned randomly to partitions in a round-robin manner so that each partition contains a more or less equal number of rows. This is the default strategy.

You can:

- Create partitions when you create a table or index using the **create table** and **create index** commands.
- Alter a table's partitioning strategy using the **alter table** command.
- Add a partition to an existing table with **add partition**.
- You can use partitioning to expedite the loading of large amounts of table data—even when the table eventually will be used as an unpartitioned table.

Row-Locked System Catalogs

Many system catalogs can now use a datarows locking scheme.

Adaptive Server version 15.0 converts most system catalogs to a datarows locking scheme. These system catalogs continue to use allpages locking scheme:

- Materialized tables such as *syslocks* and *sysprocesses*. These tables are generated during run-time and their locking schemes are irrelavent for concurrency.
- *sysmessages* and *sysusermessages*, which are read-only tables
- Auditing tables in *sybsecurity*, which are write-once and read many times.

Adaptive Server's internal upgrade process converts the system table locking schemes during an installation, upgrade, or load upgrade.

Because DDLs in Adaptive Server release 15.0 use the same table-level locks as 12.5.x and earlier versions, there is no concurrency improvement when you run DDLs.

Query Processor

Adaptive Server 15.0 provides an enhanced query processor

The Adaptive Server version 15.0 query processor is self-tuning, requiring fewer interventions than earlier versions. This version of Adaptive Server has less reliance on worktables for materialization between steps since the engine supports data flow between steps. However, more worktables could be used in cases where Adaptive Server determines that hash and merge operations are effective.

New features include support for:

- Both vertical and horizontal parallelism for query processing
- Improved index selection, especially for joins with OR clauses and joins and search arguments (SARGs) with mismatched but compatible datatypes
- More efficient algorithms
- Improved costing, using join histograms for joins with data skews in joining columns
- Improved query plan selection that enhances performance through:

  - New index union and index intersection strategies for queries with **and**/**or** predicates on different indexes
  - On-the-fly grouping and ordering using in-memory sorting and hashing for queries with **group by** and **order by** clauses
  - Cost-based pruning and timeout mechanisms that use permutation search strategies for large, multi-way joins, and for star and snowflake schema joins
- Improved problem diagnosis and resolution using:
  - Searchable XML format trace outputs
  - Diagnostic output from new **set** commands
- Joins involving a large number of tables
- Data and index partitioning, which are especially beneficial for very large data sets

Partitioning is the basic building block for parallelism.

Adaptive Server release 15.0 provides roundrobin partitioning. Round robin partitionin is equivalent to the 12.5 style of partitioning. During the upgrade to Adaptive Server release 15.0, all existing partitioned tables are unpartitioned and automatically converted to 1-way round robin partitioned tables.

Large Identifiers

Adaptive Server 15.0 introduces expanded limits for delimited identifiers

There are new limits for the length of object names or identifiers: 255 bytes for regular identifiers, and 253 bytes for delimited identifiers. The new limit applies to most user-defined identifiers including table name, column name, index name and so on. Due to the expanded limits, some system tables (catalogs) and built-in functions have been expanded.

For variables, “@” count as 1 byte, and the allowed name for the variable i 254 bytes.

Computed Columns

Computed columns and function-based indexes provide easier data manipulation and faster data access.

Computed columns provide easier data manipulation and faster data access by allowing you to create computed columns, computed column indexes, and function-based indexes.

- Computed columns – defined by an expression, whether from regular columns in the same row, functions, arithmetic operators,or path names.
- Indexes on computed columns, or computed column indexes – indexes that contain one or more computed columns as index keys.
- *Function-based indexes – indexes that contain one or more expressions as index keys.*

- Deterministic property – a property assuring that an expression always returns the same results from a specified set of inputs.

Computed columns and function-based indexes similarly allow you to use an expression or a function as the basis for a more complex function.

Differences Between Computed Columns and Function-Based Indexes

Computed columns and function-based indexes differ in some respects.

- A computed column provides both shorthand for an expression and indexability, while a function-based index provides no shorthand; it allows you to index the expression directly.
- A computed column can be either deterministic or nondeterministic, but a function-based index must be deterministic. “Deterministic” means that if the input values in an expression are the same, the return values must also be the same.

Differences Between Materialized and Not Materialized Computed Columns

Computed columns can be materialized or not materialized.

- Columns that are materialized are preevaluated and stored in the table when base columns are inserted or updated. The values associated with them are stored in both the data row and the index row. Any subsequent access to a materialized column does not require reevaluation; its preevaluated result is accessed. Once a column is materialized, each access to it returns the same value.
- Columns that are not materialized are also called virtual columns; virtual columns become materialized when they are accessed. If a column is virtual, or not materialized, its result value must be evaluated each time the column is accessed. This means that if the virtual computed column is expression-based on, or calls a nondeterministic expression, it may return different values each time you access it.You may also encounter run-time exceptions, such as domain errors, when you access virtual computed columns.

Scrollable Cursors

Adaptive Server Enterprise 15.0 supports both scrollable and nonscrollable cursors.

“Scrollable” means that you can scroll through the cursor result set by fetching any, or many, rows, rather than one row at a time; you can also scan the result set repeatedly. You must use Transact-SQL or JDBC to declare a scrollable cursor, and you must have the query engine provided in Adaptive Server 15.0 or later. A scrollable cursor allows you to set the position of the cursor anywhere in the cursor result set for as long as the cursor is open, by specifying the option **first**, **last**, **absolute**, **next**, **prior**, or **relative** in a **fetch** statement.

To fetch the last row in a result set, enter:

```
fetch last [from] <cursor_name>
```

Or, to select a specific row in the result set, in this case the 500th row, enter:

```
fetch absolute 500 [from] <cursor_name>
```

"Insensitive" or "semi-sensitive" refers to the extent to which data changes from outside the cursor are visible to the cursor. A cursor can be semi-sensitive but not scrollable.

All scrollable cursors are read-only. All **update** cursors are nonscrollable.

unitext Datatype Support

Adaptive Server 15.0 supports the vaiable-length *unitext* datatype.

The variable-length *unitext* datatype can hold up to 1,073,741,823 Unicode characters (2,147,483,646 bytes). You can use *unitext* anywhere you use the *text* datatype, with the same semantics. *unitext* columns are stored in UTF-16 encoding, regardless of the Adaptive Server default character set.

The benefits of *unitext* include:

- Large Unicode character data. Together with *unichar* and *univarchar* datatypes, Adaptive Server provides complete Unicode datatype support, which is best for incremental multilingual applications.
- *unitext* stores data in UTF-16, which is the native encoding datatype for Windows and Java environments.

  See the *System Administration Guide*.

big int Datatype Support

Adaptive Server version 15.0 supports the exact numeric datatype *bigint*.

This is the range of numbers allowed by the *bigint* datatype:

Datatype	Range of signed datatypes
bigint	Whole numbers between -2^{63} and $2^{63} - 1$ (from -9,223,372,036,854,775,808 to +9,223,372,036,854,775,807, inclusive.

Adaptive Server *bigint* support also adds the **hextobigint**, **biginttohex**, and **count_big** functions.

See the *Reference Manual: Blocks*.

Unsigned Integer Datatype Support

Adaptive Server 15.0 supports unsigned integer datatypes.

These unsigned integer datatypes allow you to extend the range of the positive numbers for the existing integer types without increasing the required storage size. That is, the signed versions of these datatypes extend both in the negative direction and the positive direction (for example, from -32 to +32). However, the unsigned versions extend only in the positive direction. This is the range for signed and unsigned datatypes:

Datatype	Range of signed datatypes	Range of unsigned datatypes
bigint	Whole numbers between -2^{63} and $2^{63} - 1$ (from -9,223,372,036,854,775,808 to +9,223,372,036,854,775,807, inclusive	Whole numbers between 0 and 18,446,744,073,709,551,615
int	Whole numbers between -2^{31} and $2^{31} - 1$ (-2,147,483,648 and 2,147,483,647), inclusive	Whole numbers between 0 and 4,294,967,295
smallint	Whole numbers between -2^{15} and $2^{15} -1$ (-32,768 and 32,767), inclusive	Whole numbers between 0 and 65535

Integer Identity

Adaptive Server 15.0 allows you to use certain datatypes as identity values.

Use these datatypes as identity values:

- *bigint*
- *int*
- *numeric*
- *smallint*
- *tinyint*
- *unsigned bigint*
- *unsigned int*
- *unsigned smallint*

Enhancements to XML Services

XML enhancements in 15.0 include XML schema support, **for xml** clause enhancements, and Unicode (I18N) support.

XML Schema Support

You can validate XML documents against either a DTD or an XML schema. The DTD or schema can be specified either in the **xmlvalidate** command or in the document itself.

You can parse, store, and query XML documents with XML schema declarations.

for xml Enhancements

In Transact-SQL, an expression subquery is a parenthesized subquery. It has a single column, the value of which is the expression subquery result, and must return a single row. You can use an expression subquery almost anywhere you can use an expression.

For more information about subqueries, see the *Transact-SQL® User's Guide*.The **for xml** subqueries feature allows you to use any subquery containing a **for xml** clause as an expression subquery. For the syntax of **for xml** subqueries, see the *XML Services Guide*.

unicode Internationalization (I18N) Support

The I18N extensions fall into three categories:

- I18N support in the **for xml** clause. The columns of the result set you map to XML can contain non-ASCII data.Such data can be represented in the generated SQLX XML document either as plain characters or as numeric character representations (NCRs).
- I18N in **xmlparse** and **xmlvalidate**, to store and validate documents containing non-ASCII data.
- I18N in **xmlextract** and **xmltest**, to process XML documents and queries containing non-ASCII data.

Adaptive Server Plug-in Enhancements

Adaptive Server 15.0 includes enhancements to the Adaptive Server Plug-in.

Enhancements to the Adaptive Server Plug-in improve efficiency and convenience:

- An enterprise view that includes Server Discovery (which enables you to find available servers on the system) and automatic server status.
- The ability to update servers, administrate remote servers, and manage server logs.
- SQL Preview and Job Scheduler integration.

- A graphical query plan viewer.
- The ability to integrate external tools.

Interactive SQL

Interactive SQL allows you to execute SQL statements, build scripts, and display database data to the server.

You can run Interactive SQL individually or from the Adaptive Server Plug-in. It has been integrated in the Adaptive Server Plug-in as the standard query tool. You can use Interactive SQL to:

- Browse the information in a database.
- Test SQL statements that you plan to include in an application.
- Load data into a database and carrying out administrative tasks.

In addition, Interactive SQL can run command files or script files. For example, you can build repeatable scripts to run against a database and then use Interactive SQL to execute these scripts as batches.

User-Defined Web Services

Web Services lets you create Web services and execute SQL in Adaptive Server.

In addition to the Web methods provided by the Adaptive Server Web Services Engine, Web Services lets you create Web services and execute SQL commands in Adaptive Server Enterprise using either a Web browser or a SOAP client. These user-defined Web services use existing security and auditing control inherent in Adaptive Server Enterprise.

You can create a user-defined Web service with the **create service** command, which enables you to specify the SQL to be executed, create a first-class object for which permissions can be controlled with the **grant** command, and control whether the service can be invoked with a Web browser or a SOAP client. The ASE Web Services Engine automatically generates WSDL for user-defined Web services.

See the Adaptive Server Enterprise Web Services *User's Guide*.

Very Large Storage Support

Adaptive Server 15.0 extends the allowable number of disk devices and the allowable number of 2K blocks for each device.

In pre-15.0 versions of Adaptive Server, a virtual page is described internally in a 32-bit integer: the first byte holds the device number (**vdevno**) and the succeeding three bytes describe the page offset within the device in units of 2K bytes (the virtual page number). This

architecture limits the number of devices to 256 and the size of each device to 32 gigabytes —for a maximum storage limit of 8 terabytes in the entire server.

With Adaptive Server 15.0, the device number and the page offset are stored in separate 32-bit integers. The new architecture allows you to create up to 2,147,483,647 disk devices, each of which can be as large as 2,147,483,648 2K blocks or 4 terabytes.

Note: Because of schema changes to the *sysdevices* and *sysusages* system tables, you may need to modify scripts and stored procedures that access these tables. The device identifier must now be obtained from the *vdevno* columns of *sysdevices* and *sysusages*. The *high*, *low*, and *vstart* columns of these tables no longer store the device and virtual page number—they store only the virtual page numbers.

Automatic Running of update statistics

Run the **update statistics** command automatically at times that suit your site.

Instead of manually running **update statistics** at a certain time, you can set **update statistics** to run automatically at the time that best suits your site, and avoid running it at times that hamper your system. The best time to run **update statistics** is based on the feedback from the **datachange** function. **datachange** also helps to ensure that you do not unnecessarily run **update statistics**. You can use these templates to determine the objects, schedules, priority, and **datachange** thresholds that trigger **update statistics**, which ensures that critical resources are used only when the query processor generates more efficient plans.

Because it is a resource intensive task, the decision to run **update statistics** should be based on a specific set of criteria. Some of the key parameters that can help you determine a good time to run **update statistics** are:

- How much has the data characteristics changed since you last ran update statistics? This is known as the “datachange” parameter.
- Are there sufficient resources available to run **update statistics**? These include resources such as the number of idle cpu cycles and making sure that critical online activity does not occur during **update statistics**.

Datachange is a key metric that helps you measure the amount of altered data since you last ran **update statistics**, and is tracked by the **datachange** function. Using this metric and the criteria for resource availability, you can automate the process of running **update statistics**. The Job Scheduler provides the mechanism to automatically run update statistics. Job Scheduler includes a set of customizable templates that determine when **update statistics** should be run. These inputs include all parameters to **update statistics**, the **datachange** threshold values, and the time when to run **update statistics**. The Job Scheduler runs update statistics at a low priority so it does not affect critical jobs that are running concurrently.

SySAM License Management

The Sybase Software Asset Management (SySAM) implementation has changed.

The changes include:

- Asset management and reporting tools are provided with SySAM version 2.0. These tools allow you to monitor license usage and compliance.
- A single installation method supports all Adaptive Server editions.
- SySAM configuration is no longer optional.
- Flexible SySAM configuration options are provided.
- SySAM licenses are no longer shipped along with order fulfillment. You must obtain license certificates from the Sybase Product Download Center (SPDC).
- SySAM license keys include information about the support plan you purchased. You must update these licenses whenever you renew your support plan.
- Licensing policies are strictly and consistently enforced.
- Adaptive Server can function under grace periods if it is not able to obtain a license. These grace periods allow customers reasonable time to respond to the issues causing license failure. Adaptive Server continues to function normally during the grace period. Adaptive Server features or the server itself will shut down at the end of the grace period if the licensing issues are not resolved.
- You can receive real-time e-mail notifications about licensing events.
- Licenses issued from SPDC include information about the host machine where the licenses will be deployed. These licenses cannot be used on another machine without being reissued from SPDC.

These changes affect the Adaptive Server installation and configuration process. See the SySAM Configuration chapter of the *Configuration Guide* for details on SySAM configuration and deployment options. See the *Adaptive Server Installation Guide* for your platform on pre-installation planning and SySAM installation information.

Plan your SySAM deployment before installing Adaptive Server.

Warning! SySAM provides for grace periods when it encounters licensing problems. When Adaptive Server enters such a grace period, the Adaptive Server error log is updated with this information. Optionally, e-mail notifications can be configured for such events. You must fix the problems causing Adaptive Server to go into grace. While Adaptive Server functions normally during this grace period, it may shutdown or disable the licensed features if the problem causing license failure is not fixed within the grace period.

Query Processing Metrics (qp Metrics)

Query processing (QP) metrics identify and compare empirical metric values in query execution. When a query is executed, it is associated with a set of defined metrics that are the basis for comparison in QP metrics.

The metrics captured include:

- CPU execution time – the time, in milliseconds, it takes to execute the query.
- Elapsed time – the difference in milliseconds between the time the command started and the current time, as taken from the operating system clock.
- Logical IO (LIO) reads – the number od Logical IO reads.
- Physical IO (PIO) reads – the number of Physical IO reads.
- Count – the number of times a query is executed.
- Abort count – the number of times a query is aborted by the resource governor due to a resource limit being exceeded.

Each metric has three values: minimum, maximum, and average. Count and abort count are not included.

Updates to Abstract Plans

Adaptive Server 15.0 supports enhancements to the query processor.

For a description of the new and changed abstract plans, see the *Query Processing Guide*.

showplan Changes

The Adaptive Server 15.0 version of **showplan** better represents the steps performed by the query processor.

Adaptive Server changes the format of the **showplan** messages to better convey the shape of the query plan. Instead of the **showplan** messages displayed in a vertical format:

```
delete
from authors
where au_lname = "Willis"
and au_fname = "Max"
```

```
QUERY PLAN FOR STATEMENT 1 (at line 1).

The type of query is DELETE.
        The update mode is direct.

        FROM TABLE
           authors
```

```
        Nested iteration.
        Using Clustered Index.
        Index : au_names_ix
        Forward scan.
        Positioning by key.
        Keys are:
           au_lname  ASC
           au_fname  ASC
        Using I/O Size 2 Kbytes for index leaf pages.
        With LRU Buffer Replacement Strategy for index leaf pages.
        Using I/O Size 2 Kbytes for data pages.
        With LRU Buffer Replacement Strategy for data pages.
        TO TABLE
           authors
```

The Adaptive Server 15.0 version of **showplan** displays a series of "pipes" (the "|" symbol) to distinguish each of the steps performed by the operators.

In the following query, there are three operators, EMIT, DELETE, and SCAN, so this query includes three sets of pipes to display this organization:

```
delete
from authors
where au_lname = "Willis"
and au_fname = "Max"
```

```
QUERY PLAN FOR STATEMENT 1 (at line 1).
2 operator(s) under root

The type of query is DELETE.
ROOT:EMIT Operator
  |DELETE Operator
  |  The update mode is direct.
  |
  |  |  |SCAN Operator
  |  |  |  FROM TABLE
  |  |  |  authors
  |  |  |  Index : aumind
  |  |  |  Forward Scan.
  |  |  |  Positioning by key.
  |  |  |  Keys are:
  |  |  |       au_lname ASC
  |  |  |       au_fname ASC
  |  |  |  Using I/O Size 8 Kbytes for index leaf pages.
  |  |  |  Using LRU Buffer Replacement Strategy for index leaf pages
  |  |  |  Using I/O Size 8 Kbytes for data pages.
  |  |  |  With LRU Buffer Replacement Strategy for data pages.
  |
  |  TO TABLE
  |   authors
  |   Using I/O Size 8 Kbytes for data pages.
```

Note: This version of Adaptive Server also includes the ability to display **showplan** messages in XML.

Secure Socket Layer Uses FIPS 140-2

In Adaptive Server 15.0, the SSL uses cryptographic modules validated for FIPS 140-2, level 1.

Secure Socket Layer (SSL) is the standard for securing the transmission of sensitive information – such as credit card numbers, stock trades, and banking transactions – over the Internet. SSL relies on public key and secret key cryptography.

The SSL used in Adaptive Server release 15.0 uses cryptographic modules validated for FIPS 140-2, level 1. The cryptographic modules are Certicom Security Builder GSE for Adaptive Server products running on Windows, Solaris, AIX and HPUX operating systems. For more information, see validation certificate #542, dated June 2, 2005 at NIST website, http://csrc.nist.gov/cryptval/140-1/1401val.htm.

System Changes in Adaptive Server 15.0

Adaptive Server 15.0 introduces changes in utilities, commands, system tables, functions, stored procedures, reserved words, and monitoring tables.

Utilities

Adaptive Server 15.0 supports changes to the **bcp**, **dataserver**, **sqlsrvr**, **ddlgen**, and **preupgrade** utilities.

Table 32. Changes to utility programs

Utility	Change
bcp	Adds new parameters **--sho-fi** and **--hide-vcc**, to support computed columns and functional indexes. Adds new parmeter **--maxconn** to support for parallel loading into partitioned tables. **bcp** interface has changed to now allow you to run **bcp in** and **bcp out** to and from specific partitions.
dataserver	Specifies the **-b master_database_size** parameter in terabytes.
sqlsrvr	Specifies the **-b master_database_size** parameter in terabytes.
ddlgen	Adds the WS object type for the **-T object_type** parameter to support user-defined Web services.
preupgrade	**preupgrade** includes options to perform incremental checks for various upgrade checks and is enhanced to run on a single database that is undergoing an upgrade using **load database**

See the *Utility Guide*.

Reserved Words

Adaptive Server 15.0 supports new reserved words in support of scrollable cursors and XML services.

- **insensitive** – supports scrollable cursors
- **xmlextract** – supports XML services
- **xmlparse** – supports XML services

- **xmltest** – supports XML services

You must change all database names that are new reserved words before you can upgrade from an earlier release of the server. You can change table, view, and column names or use delimited identifiers. Once you upgrade to version 15.0, you cannot use database objects whose names are new reserved words until you modify your procedures, SQL scripts, and applications.

See the Reference Manual: Blocks.

Global Variables

Adaptive Server 15.0 introduces new global variables.

Table 33. New global variables

Variable	What it displays	Value
@@rowcount	Enhanced to display the number of rows moved by a cursor, scrollable or nonscrollable. In a nonscrollable cursor, the rows are fetched from the underlying tables to the client. In a scrollable cursor, the rows counted are fetched from the current result set, not from the underlying tables.	The value of the global variable **@@rowcount** is affected by the specified cursor type. The default, non-scrollable cursor moves forward one row at a time; the maximum value is the number of rows in the result set. In a scrollable cursor the value of continues to increment, whatever the direction of the **fetch** command; there is no maximum value.
@@fetch_status	The status of a **fetch** command used for a scrollable cursor.	0 – **fetch** statement successfully executed. -1 – either the **fetch** statement failed, or the row requested is outside the result set. -2 – value reserved.

Variable	What it displays	Value
@@cursor_rows	The total number of rows in the cursor result set.	0 – No cursors are open, or no rows qualify for the last open cursor. -1 – Semi-sensitive and scrollable, but the scrolling worktable is not yet populated. The number of rows that qualify the cursor is unknown. *n* – The last opened or fetched cursor result set is fully populated; the value returned is the total number of rows in the result set.

Configuration Parameters

Adaptive Server 15.0 introduces new and changed configuration parameters.

Table 34. New configuration parameters

Function	Description
enable metrics capture	Enables Adaptive Server to capture metrics at the server level.
enable semantic partitioning	Enables semantic (hash-, list-, range-) partitioning of tables and indexes at a licensed site.
enable web services	Enables web services
enable xml	Enables the XML services
max native threads per engine	Defines the maximum number of native threads the server spawns per engine
max partition degree	Configures the amount of dynamic repartitioning Adaptive Server requires, which enables Adaptive Server to use horizontal parallelism
max resource granularity	Sets the maximum percentage of the system's resources a query can use
number of devices	Specifies the number of database devices Adaptive Server can use
number of dump threads	Controls the number of threads that Adaptive Server spawns to perform a memory dump
number of open partitions	Specifies the number of partitions that Adaptive Server can access at one time.

Function	Description
optimization goal	Allows you to configure for three optimization goals, which you can specify at three tiers: server level, session level, and query level
optimization timeout limit	Specifies the amount of time Adaptive Server can spend optimizing a query as a percentage of the total time spent processing the query
rtm thread idle wait period	Defines the time a native thread used by Adaptive Server waits when it has no work to do
sysstatistics flush interval	Determines the length of the interval (in minutes) between flushes of *sysstatistics*
statement cache size	Increases the server allocation of procedure cache memory and limits the amount of memory from the procedure cache pool used for cached statements. The statement cache feature is enabled server-wide.

Table 35. Changed configuration parameters

Parameter	Change
default network packet size	Previous versions of Adaptive Server used a **default network packet size** of 512. As of Adaptive Server version 15.0, the **default network packet size** is 2048.

See the *Administration Guide: Volume 1*.

Functions

New and changed functions for Adaptive Server 15.0

Table 36. New functions

Function	Description
biginttohex	Returns the platform-independent hexadecimal equivalent of the specified integer
count_big	Returns the number of (distinct) non-null values or the number of selected rows as a *bigint*
datachange	Measures the amount of change in the data distribution since update statistics
data_pages	Returns the number of pages used by the specified table, index, or a specific

Function	Description
hextobigint	Returns the *bigint* value equivalent of a hexadecimal string
is_quiesced	Returns 1 if the database is quiesced and 0 if it is not.
partition_id	Returns the partition id of the specified data or index partition name.
partition_name	The explicit name of a new partition, partition_name returns the partition name
reserved_pages	Reports the number of pages reserved to a table, index or a specific partition.
row_count	Returns an estimate of the number of rows in the specified table.
showplan_in_xml	Returns the execution plan in XML.
ssel_message	Returns the message text when you specify a message ID.
tran_dumpable_status	Returns a true/false indication of whether **dump transaction** is allowed.
used_pages	Reports the number of pages used by a table, an index, or a specific partition.
xmlvalidate	Validates XML documents, including those containing non-ASCII characters (I18N). Described in *XML Services*.

Several function names have been replaced with more readable names.

Table 37. Superceded functions with their new function names

Superceded function name	New function name
data_pgs	**data_pages**
used_pgs	**used_pages**
reserved_pgs	**reserved_pages**
rowcnt	**row_count**
ptn_data_pgs	**data_pages**

Commands

New and changed commands in Adaptive Server 15.0

Table 38. New commands

Command	Function
create service	For creating a user-defined Web service.
drop service	For creating a user-defined Web service.
update table statistics	Updatex *systabstats* statistics for a table or a partition.

Table 39. New set command options

set Option	Description
set delayed_commit	Allows you to determine when log records are written to disk. With the **delayed_commit** parameter set to true, the log records are asynchronously written to the disk and control is returned to the client without waiting for the IO to complete
set plan optgoal	Sets the optimization goals at the session level.
set plan opttimeoutlimit	Sets the limit the time taken by long-running and complex queries at the session level.
set metrics_capture on/off	Activates QP metrics at the session level.

Table 40. Changed commands

Command	Change
alter table	Syntax added to support computed and materialized or non-materialized columns. Adds support for paritions.
create index	Enhanced to allow computed columns to be used as index keys, in the same way as regular columns, and to create function-based indexes. Adds support for paritions.
create table	Syntax added to support computed and materialized or non-materialized columns. Adds support for paritions.
dbcc	Adds support for partitions.

Command	Change
declare cursor	Syntax added for scrollable cursors. Syntax added to support **semi_sensitive**, **insensitive**, and **scrollable** cursors.
delete statistics	Adds support for partitions.
disk init	The **size** parameter can be specified in terabytes. Adds the **directio** parameter, which allows you to configure Adaptive Server to transfer data directly to disk, bypassing the operating system buffer cache
disk reinit	Adds the **directio** parameter, which allows you to configure Adaptive Server to transfer data directly to disk, bypassing the operating system buffer cache
fetch	**fetch_orientation** options added to support scrollable cursors: **next**, **prior**, **first**, **last**, **absolute**, and **relative**.
reorg	Adds support for partitions.
select	**for xml** clauses added to support XML services. Adds support for partitions.
truncate table	Adds support for partitions.
update all statistics	Adds support for partitions.
update statistics	Adds support for partitions.
update partition statistics	Made obsolete.

System Stored Procedures

New and changed system stored procedures in Adaptive Server 15.0

Table 41. New stored procedures

Stored procedure	Function
sp_helpcomputed-column	Reports information on all the computed columns in a specified table
sp_version	Returns the version information of the installation scripts (`install-master`, `installdbccdb`, and so on) that was last run and whether it was successful.

Table 42. Changed system stored procedures

Stored procedure	Change
sp_checksource	Checks the existence of computed columns source text.
sp_help	Reports information on computed columns, function-based indexes, and partitions.
sp_helpartion	Adds detailed partition information to its output.
sp_helpindex	Reports information on computed column indexes, function-based indexes, and partitions.
sp_helptext	Displays the source text of computed columns, function-based index definitions, and partitions.
sp_hidetext	Hides the text of computed columns, function-based index keys, and partition condition.
sp_modifylogin	Adds option **"enable logins during recovery"**.
sp_webservices	Adds **addalias**, **deploy**, **dropalias**, **listudws**, **listalias**, and **undeploy** options to support user-defined Web Services.
sp_monitorconfig	Supports the **number of open partitions** configuration parameter.
sp_countmetadata	Supports the **number of open partitions** configuration parameter.
sp_helpsegment	Prints segment bindings for objects and partitions.
sp_objectsegment	Displays segment information for all partitions for an object.
sp_placeobject	Enables future allocations for a partition from a new segment.
sp_dbcc_faultreport	Creates reports for a specific OPID or fault type.
sp_sysmon	Reports information related to open partitions Metadata Cache Management section of the configuration file.

See the *Reference Manual: Procedures*.

System Tables

New and changed system tables in Adaptive Server 15.0

Adaptive Server version 15.0 provides the necessary row-locked catalog infrastructure to support enhanced, multi-user-concurrent data-definition language (DDL) operations. However, this release does not change the catalog locking behavior for DDL operations. Applications that perform heavy multi-user DDL operations (for example, creating or

dropping tables in *tempdb*, will not see any change in behaviour in this release for catalog blocking, or any increased DDL concurency.

Table 43. New system tables

Table	Description
syspartitions	*syspartitions* is completely changed from the pre-15.0 version of the table. All columns are new. *syspartitions* supports both semantic and round-robin partitioning of tables and indexes.
syspartitionkeys	Contains a row for each column in a partition key for each hash-, range-, and list-partitioned table.

Table 44. Changed system tables

Table	Change
syscolumns	New fields: • **computedcol** • **status3** New columns: • *encrtype* –Type of encryption • *encrlen* – Length of encrypted column • *encrkeydid* – Encryption key id • *encrkeydb* – Database name containing encryption key • *encrdate* – Date column was encrypted. New bits in *status2* field: • Hex: 0x00000010, Decimal 16 – the column is a computed column. • Hex: 0x00000020, Decimal 32 – the column is a materialized computed column. • Hex: 0x00000040, Decimal 64 – the column is a computed column in a view.
sysconstraints	New internal bit in status field: Hex 0x0100, decimal 265 – indicates a computed column object.

Table	Change
sysdevices	New columns: • *vdevno* – device identification number • *crdate* – date device created • *resizedate* – date size of device changed • *status2* – Additional status2 bits.
sysusages	New column: *vdevno* – device identification number
sysstatistics	New columns: • *indid* – index ID of the data partition. Always 0. • *partitionid* – ID of the data partition • *ststatus* – Internal status bits Unique placement index on *id*, *indid*, *partitionid*, *statid*, *colidarry*, *formatid*, *sequence*
systabstats	New columns: • *partitionid* – ID of data or index partition • *statmoddate* – Date when statistics were last modified on disk. • *unusedpgcnt* – Number of unused pages. • *oampagecnt* – Number of OAM pages for each partition.
syspartitions	*syspartitions* is completely changed from the pre-15.0 version of the table. All columns are new. *syspartitions* supports both semantic and round-robin partitioning of tables and indexes.
syscomments	New column: *partitionid* – ID of data or index partition Table enhanced to store the text of computed column or function-based index key expression.
sysindexes	New columns: • *partitiontype* – partitioning strategy: 1 – range, 2 – hash, 3 – round-robin, 4 – list • *conditionid* – ID of the partition condition New rows: contains one row for each function-based index or index created on a computed column. One new internal status bit added to the status2 field: Hex 0x8000, decimal 32768 – the index is a function-based index.

Table	Change
syslocks	New columns: • *nodeid* – Reserved for future use. • *partitionid* – ID of data or index partition. Reserved for future use. Always 0.
sysobjects	New object in *type* column: *N* – partition condition New column: • *identburnmax* – For an identity column, maximum burned identity value • *spacestates* – Number of space states being tracked. (Only applies for DOL tables.) • *erlchgts* – Timestamp when expected row length was last changed. (Only applies to DOL tables.) New row: one row for each computed column and function-based index key object • *type* field: type "C" added to the type field, when the object is a computed column • *status2* field: new bit added to indicate that the table contains one or more function-based indexes.
sysprocedures	Stores a sequence tree for each computed column or function-based index definition, in binary form

Table 45. Datatype changes in system table columns

System table	Changed column	Datatype changes	Identifier name
sysattributes	*object_cinfo* *char_info*	*varchar(30) null* to *varchar(255) null* *varchar(255)* to *varchar(768)*	Identifier for the object
sysaudits01 – sysaudits08	*objname*	*varchar(30) not null* to *varchar(255) not null*	Object name
syscolumns	*name*	*varchar(30) not null* to *varchar(255) not null*	Column name
	remote_name	*varchar(30) null* to *varchar(255) null*	Maps local names to remote names
sysconfigures	*name*	*varchar(80) null* to *varchar(255) null*	

System table	Changed column	Datatype changes	Identifier name
sysindexes	*name*	*varchar(30) null* to *varchar(255) null*	Index for the table name
sysjars	*jname*	*varchar(30) null* to *varchar(255) null*	JAR name
sysobjects	*name*	*varchar(30) not null* to *varchar(255) not null*	Object name
sysprocesses	*hostname*	*char(10) not null* to *varchar(30) null*	Host computer name
	program_name	*char(16) not null* to *varchar(30) null*	Name of *front_end* module
	hostprocess	*char(8) not null* to *varchar(30) null*	Host process ID number
	cmd	*char (16) not null* to *varchar(30) null*	Command or process currently being executed. Evaluation of a conditional statement, such as an **if** or **while** loop, returns.
systimeranges	*name*	*varchar(30) not null* to *varchar(255) not null*	Unique name of the time range
systypes	*name*	*varchar(30)* to *varchar(255)*	Datatype name
sysdatabases	*def_remote_loc*	*varchar(255) null* to *varchar(349) null.*	

See the *Reference Manual: Tables*.

Monitoring Tables

New and enhanced monitoring tables in Adaptive Server 15.0

Table 46. New monitoring tables

Monitoring table	Description
monOpenPartitionAcitivity	Provides monitoring information for partitions

Table 47. Changed monitoring tables

Monitoring table	Changes
monEngine	New columns for housekeeper GC task
monCachedObject	New columns for partitions
monProcessObject	New columns for partitions

See the *Reference Manual: Tables*.

New Features in Adaptive Server 12.5.4

Adaptive Server 12.5.4 introduces security enhancements, archive database access support, changes to shared directories, support for new data providers, dynamically loaded TIBCO libraries, JRE support, modulo arithmetic for numeric datatypes, support for terminal server shared-memory requirements in Windows, and enhancements to login triggers.

Kerberos Enhancements

Adaptive Server 12.5.4 expands Kerberos capabilities

- New option to specify a Kerberos principal name different from the Adaptive Server name.
- Expanded platform support for Kerberos.
- You can now use the **sp_modifylogin** and the option **authenticate with** to require Kerberos authentication for an individual login.
- **sybmapname** is a customizable utility to convert external user principal names to the name space of Adaptive Server logins.

LDAP User Authentication Enhancements

Adaptive Server 12.5.4 supports enhanced LDAP user authentication capabilities

- Automatic failover from authentication against a primary LDAP server to a secondary LDAP server.
- Adaptive Server recovers from errors encountered when communicating with the LDAP server.
- Enhancements to the communication of password-expiration-related messages obtained from the LDAP server to Adaptive Server clients.

Password Complexity Enhancements

Adaptive Server version 12.5.4 lets you make rules about passwords for new logins or for reset passwords.

Adaptive Server 12.5.4 lets you specify a wide range of password complexity attributes:

- Set a minimum number of alphabetic characters for the password.
- Specify that the login name should not be a substring of the password.
- Set a minimum number of special characters for the password.
- Set a minimum number of upper case letters for the password.

- Set a minimum number of lower case letters for the password.
- Specify that the password must be reset at first log on.

You can set each of these new options in ASE Plug-in or you can use a new stored procedure **sp_passwordpolicy**.

Archive Database Access Support

Archive database access enables a variety of operations to be performed directly on a database dump

Archive database access provides a database administrator with the capability to validate or selectively recover data from a database dump by making the dump appear as if it were a traditional, read-only database.

Shared Directory Changes

Adaptive Server version 12.5.4 includes changes to the shared directory structure

Table 48. Shared directory structure changes for UNIX and Linux platforms

Component	Old location	New location
Shared directory	$SYBASE/shared-1_0	$SYBASE/shared
Sybase Central	$SYBASE/sybcent41	$SYBASE/shared/sybcentral43
JRE	$SYBASE/shared-1_0/JRE-1_3	$SYBASE/shared/jre142
Shared jar file	$SYBASE/shared-1_0/lib	$SYBASE/shared/lib

Table 49. Shared directory structure changes for Microsoft Windows platforms

Component	Old location	New location
Shared directory	%SYBASE%\shared-1_0	%SYBASE%\Shared
Sybase Central	%SYBASE%\sybcent41	%SYBASE%\Shared\Sybase Central 4.3
JRE	%SYBASE% \shared-1_0\JRE-1_3	%SYBASE%\Shared\Sun\jre142
Shared jar file	%SYBASE%\shared-1_0\lib	%SYBASE%\Shared\lib

Sybase Driver Support

Adaptive Server 12.5.4 supports new Adaptive Server Data Providers

The new drivers are:

- ASE ADO.NET Data Provider 1.1
- ASE OLE DB Provider by Sybase 12.5.1
- ASE ODBC Driver by Sybase 12.5.1

Dynamically Loaded TIBCO Libraries

TIBCO JMS libraries are now dynamically loaded rather than statically linked to the Adaptive Server executable.

Adaptive Server version 12.5.4 dynamically loads the messaging libraries it needs to interact with TIBCO EMS and IBM MQ message buses.

The Adaptive Server Messaging libraries contain the messaging logic and act as a wrapper on top of the messaging libraries provided by the vendors like TIBCO and IBM. These Adaptive Server messaging libraries are available with the purchase of RTDS 3.5 and later. After you install Adaptive Server, you must install RTDS 3.5 to install the Adaptive Server messaging DLLs. Install these libraries in the `$SYBASE/ASE-12_5/lib` directory. In addition to Adaptive Server Messaging libraries, you must have the messaging libraries and DLLs from vendors (TIBCO and IBM.)

JRE Support

Adaptive Server 12.5.4 includes JRE 1.4.

JRE 1.4 is installed in full and typical installations by default, and in custom installations whenever a component that requires the JRE to run is selected for installation.

Adaptive Server Plug-in Changes

Adaptive Server Plug-in supports archive database access and password complexity.

The Sybase Central Adaptive Server Plug-in supports archive database access and configuring password complexity options in Adaptive Server version 12.5.4.

Use the Sybase Central Adaptive Server plug-in to manage archive databases. There is a new folder called Archive Databases under the Databases folder

Use the Sybase Central Adaptive Server plug-in to configure Adaptive Server for password complexity options from the Server Property Sheet. For Adaptive Server version 12.5.4, there is a new tab in the Server Property Sheet called Login Password Configuration.

See *Managing Adaptive Server Enterpise.*

Updating System Catalogs

The server-wide command **allow updates to system catalogs** takes precedence over the stored procedure settings for **allow updates**.

Monitoring Tables Changes

Monitoring tables *monSysStatement* and *monProcessStatement* have been changed.

Table 50. New columns in monitoring tables

Column	Monitoring tables	Description
RowsAffected	*monSysStatement* and *monProcessStatement*	Indicates the number of rows affected by the current statement. Helpful when looking for queries that may be using an inefficient query plan, because these queries probably show a high number of logical I/Os per returned row.

See the *Reference Manual:Tables*.

syscomments Changes

When the source text of a stored procedure or trigger is stored in the system table *syscomments*, a query using **select *** is stored in *syscomments* expanding the column-list referenced in the **select ***.

Shared-Memory Support in Windows Terminal Server Environments

To accommodate Windows terminal server shared-memory requirements, Adaptive Server 12.5.4 introduces the new environment variable SYBASE_TS_MODE.

Global Login Trigger

Adaptive Server version 12.5.4 provides the ability to set a new global login trigger that is executed at each user login.

Exporting set Options from a Login Trigger

Adaptive Server version 12.5.4 enables **set** options inside login triggers to remain valid for the entire user session.

Modulo Arithmetic for Numeric Datatypes

In Adaptive Server version 12.5.4, you can perform modulo arithmetic on reals, floats, decimals, numerics, as well as on integers.

System Changes in Adaptive Server 12.5.4

New and changed commands, configuration parameters, and functions for Adaptive Server 12.5.4

Commands

The **shutdown**, **dump database**, and **load database** commands have changed in Adaptive Server 12.5.4. There are no new commands.

Table 51. Changes to commands

Command	Changes
shutdown	Supports new **with wait** and **with nowait** parameters that allow you to shut down the server immediately or gracefully in a specified time.
dump database	Introduces new **with verify** parameter that lets you dump a database with verification.
load database	Introduces new **with verify** parameter that lets you load a database with verification.

See the *Reference Manual:Commands*.

Configuration parameters

New configuration parameters in Adaptive Server 12.5.4

Table 52. New configuration parameter descriptions

Configuration parameter	Description
allow updates to system catalogs	A system-wide parameter that takes precedence over the stored procedure settings for **allow updates**. If not enabled server-wide stored procedure settings determine whether you can modify system catalogs.

See the *System Administration Guide*.

Functions

New functions in Adaptive Server 12.5.4

Table 53. New function descriptions

Function	Description
pssinfo	Returns information from the pss. IPv6 architecture has an IP address length of 64 bytes. As the size of *sysprocesses* can not be increased, **passinfo** retrieves the full IP address.
xa_bqual	Returns the binary version of the bqual component of an ASCII XA transaction ID.
xa_gtrid	Returns the binary version of the *gtrid* component of an ASCII XA transaction ID.

See the *Reference Manual: Blocks*.

Transact-SQL Commands

Changes to Transact-SQL commands in Adaptive Server 12.5.4

Table 54. Changes to Transact-SQL commands

Transact-SQL command	Change
select *	In 12.5.4, the expanding of the column-list is enhanced in a way that identifiers (table-names, column-names, and so on) are checked if they comply with the rules for identifiers. For all identifiers used in the text expanding a *select **, brackets are added when the identifier does not comply with the rules for identifiers. You must add brackets around identifiers to make sure Adaptive Server can use the SQL-text while performing an upgrade to a more recent release.

See the *Transact-SQL Guide*.

Obtaining Help and Additional Information

Use the Sybase Getting Started CD, SyBooks™ Online, and online help, to learn more about this product release.

- The Getting Started CD (or download) – contains release bulletins and installation guides in PDF format, and also contains other documents or updated information not included on the SyBooks CD.
- SyBooks Online at *http://sybooks.sybase.com/* – is an online version of SyBooks that you can access using a standard Web browser. You can browse documents online, or download them as PDFs. In addition to product manuals, the Web site also has links to EBFs/ Maintenance, Technical Documents, Case Management, Solved Cases, newsgroups, and other resources.
- Online help in the product, if available.

To read or print PDF documents, you need Adobe Acrobat Reader, which is available as a free download from the *Adobe* Web site.

Note: A more recent release bulletin, with critical product or document information added after the product release, may be available from SyBooks Online.

Index

F

G

H

I

J

K

L

M

T

U

V

W

X