



**Microsoft SQL Server to Adaptive Server® Enterprise Migration Guide  
for Microsoft SQL Server 2000 and Adaptive Server Enterprise 12.5**

A Technical White Paper

robust

## **Abstract**

This document is intended to aid customers who need to migrate their applications from Microsoft SQL Server to Sybase Adaptive Server Enterprise or ensure application portability between the products. It explains the differences between the products and presents a migration process to assist a customer with migration from Microsoft SQL Server. It also provides lists of issues in the appendices to assist with migration and application portability between the two products.

## Table of Contents

Introduction.....	1
How To Use This Document .....	2
Overview of Product Differences .....	2
Server Structure .....	2
Administration.....	8
Development .....	15
The Migration Process .....	20
Summary .....	24
Appendix A Portability List: Development Features .....	25
T-SQL Statement Differences .....	25
Session Option Setting Differences .....	30
Global Variable Differences.....	31
Reserved Keyword Differences .....	32
OLE DB Interfaces and Types Support .....	33
ODBC Function Compliance and Types Support.....	35
Appendix B Portability List: Administrative Features .....	37
Server Configuration Option Differences .....	37
Database Configuration Differences .....	38
DBA Command Differences .....	39
System Table Differences.....	42

## Introduction

Microsoft SQL Server (MSSQL) and Sybase Adaptive Server Enterprise (ASE) are both SQL-based client/server relational database management systems used to support information systems. For various reasons, users of MSSQL may need to migrate their applications to ASE. Other customers may need to ensure that their applications are easily portable between the two products. Luckily, the unique history of these two products makes such conversion and portability relatively straightforward.

The two products have a common heritage because, until version 4.2, Microsoft simply licensed Sybase's database server software; they were in fact two marketing entities for the same product<sup>1</sup>. Since that point, however, they have diverged. Microsoft has produced versions 6.0, 6.5, 7.0, and 2000 of Microsoft SQL Server, while Sybase has produced versions 4.8, 4.9, System 10 and System 11 of SQL Server and versions 11.5, 11.9, 12, and 12.5 of SQL Server's descendant, Adaptive Server Enterprise.

NOTE: This document refers to specific features, functions, and syntax of MSSQL 2000 SP3 (Build 8.00.731) and Sybase ASE 12.5.0.3. As newer versions of ASE are released, Sybase plans on continuing the support for both ANSI SQL and MSSQL-specific features by adding those features and thus reducing the differences between MSSQL and ASE.

This document is primarily intended to assist with the application migration process from MSSQL to ASE. By migration, in this context, we mean the process of changing an application so that it uses ASE, rather than MSSQL, as its underlying database management system; this involves moving the schema and data from MSSQL to ASE as well as re-directing the application. It is also intended to be of assistance to those software developers needing to develop applications that can be easily migrated between the products.

The white paper presents an overview of the differences between the two products, from the points of view of both the application developer and the DBA, and then describes a straightforward and systematic process for performing a migration from MSSQL to ASE.

The migration process between the two is relatively easy and involves:

- Creating a suitable ASE server
- Migrating the application database structure
- Migrating the application's data
- Ensuring that administration and security procedures are appropriate for the new environment
- Checking that application SQL will work in the ASE environment
- Ensuring that programming interfaces will migrate correctly
- Testing the resulting migrated system to ensure compatibility with the original.

In the appendices a series of lists are provided that should assist with migration planning and application design so that future applications can remain portable between MSSQL and ASE.

This document attempts to provide a comprehensive overview of the differences between MSSQL and ASE and the migration issues to be faced when converting from MSSQL to ASE. However, it is not a complete list. Please consult the referenced ASE product manuals for complete syntax, sample usage, administration, and performance tuning issues. ASE product manuals are distributed with the ASE software on the Technical Library CD-ROM and can also be found online at <http://www.sybase.com/support/manuals>.

Some of the features for MSSQL and ASE mentioned in this document are bundled with the respective software products and some features are purchased separately as add-on options. Please consult with your Sybase sales representative for the list of bundled features and add-on options available for each version and packaging of ASE.

<sup>1</sup> At that time, Sybase Adaptive Server Enterprise was known as Sybase SQL Server.

## How To Use This Document

If you are reading this document, it is likely that you are in one of two situations. Either you have an existing MSSQL application that you wish to migrate to ASE, or you are considering building an application and wish to maximize its portability between the products.

For the first case where there is an existing application, you should read the third and fourth sections of this document (Overview of Product Differences and The Migration Process). These sections will provide you with the information you need to migrate your application from MSSQL to ASE.

For the second case where the aim is to maximize application portability, you should read the third section of this document (Overview of Product Differences) and then read the appendices which contain "portability lists". These sections will provide you with the information you need to maximize your application's portability and to minimize any migration problems later.

While this document assumes that you are using at least MSSQL 2000 SP3 (8.00.731) and ASE 12.5.0.3 much of the advice still applies to earlier releases of the products as well.

## Overview of Product Differences

This section discusses the differences between the two products. Not every difference between the two products is presented here, but all of the significant differences that need to be considered when porting an application are considered. The "portability list" appendices detail additional low-level components to consider.

The differences between ASE and MSSQL are presented as server structure differences, database administration differences, and developer-related differences such as language and application programming interfaces.

## Server Structure

MSSQL and ASE servers are quite similar because much of the original architecture has been preserved. Both still use the original master, model, and tempdb system databases as well as many of the system procedures, database options, and configuration settings. However, over time they have differed in how to add features beyond the core database services. Microsoft tends to add features to the database server itself while Sybase usually creates new, dedicated, special-purpose servers to offload work from the database engine.

Database Component	MSSQL	ASE
Backups	Integrated	Separated into Backup Server™
Monitoring	Integrated with Windows PerfMon and Event Manager	Separated into Monitor Server
Extended Stored Procedures	Integrated	Separated into XP Server
Replication	Integrated	Integrated with smaller ASE Replicator as well as separated into the larger and more robust Replication Server®
Distributed Queries	Integrated with Linked Servers using OLE DB datasources for both homogeneous and heterogeneous queries	Integrated with Component Integration Services (CIS) for ASE-to-ASE queries and separated into Enterprise Connect Data Access (ECDA) for heterogeneous queries
Distributed Transactions	Integrated with MS Distributed Transaction Coordinator (DTC)	ASE provides a built-in transaction coordinator, referred to as ASTC, for transparent two-phase commit coordination. ASE can also participate in distributed transactions managed by third party coordinators such as Tuxedo, Encina or DTC
Full-Text Searching	Separated into Full-Text Search Service	Separated into Full-Text Search Engine
User-Defined Functions	Integrated using T-SQL as the base language	Integrated using a Java Virtual Machine with Java as the base language
XML Support	Integrated	Can be used as standalone or integrated into using ASE Java support

### System Databases

MSSQL and ASE each added system databases to the original set of master, model, and tempdb. The ASE sybssystemprocs database holds the bulk of the system stored procedures, which in MSSQL, still reside in master. MSSQL has an msdb database to store information for alerts, job scheduling, and a set of history tables for backup and restore. MSSQL may also have a distribution database when the server is configured for replication.

The ASE sybssystemdb database tracks information about distributed transactions and manages two-phase commits. ASE also added a dbccdb database for holding information gathered by DBCC CHECKSTORAGE concerning database integrity faults and a sybsecurity database for its auditing subsystem. ASE servers can also have an optional sybsyntax database for storing syntax information retrievable through sp\_syntax. In ASE, it is possible to define multiple user temporary databases (via CREATE TEMPORARY DATABASE) to reduce system table contention in tempdb when creating session-specific temporary tables (e.g., #temp\_table) and for database space reclamation of dropped temporary databases.

The ASE and MSSQL master and user databases still share quite a lot of the original system tables. For a comprehensive list, see System Table Differences in Appendix B.

## Logical Data Storage

The logical unit of data storage in both products is a data page, and a grouping of 8 pages is an extent. The MSSQL page size is fixed at 8KB. Objects are allocated one extent at object creation, but more than one object may reside on a mixed extent until they are able to occupy a full extent. The maximum size of a single row of non-image or text data is 8060 bytes (just under 8KB).

In ASE, a page size for the server can be one of 2KB, 4KB, 8KB, or 16KB and can only be specified when the server is created and therefore applies to all tables and indexes in the server. A larger page size raises the storage capacity of columns and rows up to the maximum of 16300 bytes (just under 16KB). Refer to limits for rows with variable and fixed-length columns for tables in the ASE Reference Manual for further details. Other ASE SQL and server structures (such as character expressions, variables and stored procedure arguments) have a maximum size of 16384 bytes no matter the page size. Each table and index is allocated space in units of one extent whether or not the extent is fully utilized.

## Memory Management

Both servers use a shared memory pool to maintain the state of server structures (such as user connections, locks, and active database object metadata), and to minimize physical disk I/O using a data buffer cache for data and index pages and a procedure cache for stored procedures, triggers, views, constraints, rules, defaults, and ad-hoc SQL batches. MSSQL dynamically grows and shrinks its data buffer cache to keep available physical memory on the machine to between 4MB and 10MB. Alternatively, by using the configuration option "set working set size", MSSQL reserves physical memory up to "max server memory".

With ASE, the DBA sets the maximum amount of memory for the server to use with the configuration option "max memory". The DBA configures the specific sizes and options of each of the data caches with `sp_cacheconfig`. The sizes of the procedure cache and server structures can be set and tuned dynamically (i.e., without a database server restart) via `sp_configure`. Physical memory can be allocated dynamically as the structures are consumed by users (via the configuration option "dynamic allocation on demand") or all at once at start-up. You can also allocate all of "max memory" at once (via the configuration option "allocate max shared memory") to increase performance. Unlike MSSQL, ASE never dynamically reduces the amount of physical memory that it uses.

Two additional features of ASE memory management are named caches and large I/O buffer pools. Named caches allow the data buffer cache to be partitioned into multiple, smaller caches that can be dedicated to a particular type of workload (e.g., static lookup tables in one cache, OLTP tables in another, DSS in another, etc.). The default data cache and other named caches can be further subdivided into buffer pools of varying I/O sizes of 1, 2, 4 or 8 pages via `sp_poolconfig`. A buffer pool with an I/O size that is eight times the server page size will read an entire extent in a single disk I/O operation. Judicious use of named caches and buffer pools can produce dramatic improvements in server throughput by reducing interference between competing workloads. For information on how to correlate the server configuration parameters for tuning memory, see Server Configuration Option Differences in Appendix B.

## **Disk I/O**

The disk I/O processing performed by the database server is critical to achieving acceptable performance for large applications. Both products attempt to maximize the speed of disk I/O in order to maximize server throughput.

MSSQL disk I/O is rather simple with the database server doing all I/O in units of one 8KB page. However, the MSSQL advanced scan feature allows multiple tasks to share full table scans by joining the tasks at the start of each request and then re-looping through the scan to provide missing rows to the late-comers. Also, the MSSQL Storage Engine performs scatter-gather I/O allowing the transfer of data between cache and disk from non-contiguous areas of memory.

ASE I/O buffer pools, as discussed above, allow the disk I/O sizes to occur at units of 1, 2, 4, or 8 pages at a time. Depending on the server page size, disk I/O sizes range from 2KB to 16KB for a 2KB-page server up to 16KB to 128KB for a 16KB-page server.

MSSQL read-ahead processing allows more data to be read from disk than is immediately required, presuming that it will be used in the future – it is automatic and not a tuneable component. ASE offers an analogous feature known asynchronous pre-fetch (APF); however the DBA may tune the maximum amount of memory in each of the I/O buffer pools to use for APF reads using `sp_poolconfig`.

Careful use of ASE named caches, I/O buffer pools, and the APF feature will result in a very highly tuned database server that can achieve extremely efficient disk I/O even for servers supporting mixed workloads.

## **Query Processor**

ASE and MSSQL feature similar cost-based query optimizers using slightly different execution strategies, statistics, and optimization approaches. For example, ASE allows fine DBA control over table, index, and column histogram statistics, including allowing the direct update of the statistics. Most ASE table and index statistics are dynamically maintained while column distribution statistics must be manually updated using the `UPDATE STATISTICS` command. MSSQL enforces a particular set of statistics that it attempts to collect automatically.

DBAs will need to review the differences between the statistics in order to ensure migrated applications perform well and to ensure application portability.

## **Locking**

Both products offer table, page, and row-level lock granularities. MSSQL decides which locking strategy to use for tables and indexes dynamically. To override this feature, `sp_indexoption` is used to set whether a combination of page and table-level locking or row-level locking should be allowed or disallowed.

ASE allows the table owner to select one of three specific locking schemes for each table. For a table created with the `datarows` locking scheme, ASE locks only individual data rows within a data page (no index rows or pages are locked) which provides the highest degree of concurrency, but with the highest overhead. For a table created with the `datapages` locking scheme, ASE locks a data page only (no index rows or pages are locked). For the `allpages` locking scheme, ASE locks both data pages and index pages which provides the lowest degree of concurrency, but with the lowest overhead. In all three schemes, the server can decide to promote to table-level locking should the lock promotion thresholds set by the DBA be crossed. These locking mechanisms provide a wide degree of flexibility for DBAs who need to optimize concurrency, resource overhead, and space management in their applications.

## **Distributed Queries**

MSSQL offers the linked servers feature to allow access to objects in remote OLE DB data sources via rowsets. ASE offers a feature known as Component Integration Services (CIS) to create proxy tables as references to tables located on remote servers. ASE and CIS allow for native distributed data across Sybase servers. CIS in conjunction with Sybase Enterprise Connect Data Access (ECDA) allows for remote access to objects in many types of remote data sources, including MSSQL, Oracle, DB2 (for OS/390, AS/400, Windows, and Unix), Informix and many ODBC data sources.

## **Distributed Transactions**

Distributed transactions in MSSQL are handled via Microsoft's Distributed Transaction Coordinator (MS DTC) allowing transactions to be coordinated between multiple MSSQL servers and other database servers. MS DTC also allows MSSQL server to act as a resource manager within an XA distributed transaction. MSSQL stores the outcome of the transactions in msdb.

For ASE, distributed transactions may be coordinated natively or through an external transaction manager. The built-in Adaptive Server Transaction Coordinator (ASTC) automatically coordinates transactions distributed via CIS allowing transparent two-phase commit operations between ASE and Oracle, while providing best-effort coordination between other third-party databases (such as MSSQL, DB2, Informix) using Sybase's ECDA servers.

The ASE Distributed Transaction Manager (DTM) communicates with third party transaction managers (such as CICS, Tuxedo, Encina, or MS DTC) to participate in both XA and MSDTC distributed transactions. ASE stores the outcome of distributed transactions in sybssystemdb. See the ASE product manual Using Adaptive Server Distributed Transaction Management Features for more information.

## **Process Management**

ASE provides a Logical Process Manager (LPM) to allow the workload (particularly for multi-processor machines) to be managed within the server. In particular, work can be partitioned into groups with differing priorities, and groups can be limited to use certain ASE (CPU-based) engines. LPM allows work from an application, login, or stored procedure name to be granted special priority (high, medium or low) and affinity to specific CPUs (all or some) via `sp_addengine`, `sp_addexclass`, and `sp_bindexclass`.

MSSQL does not provide such extensive management and so applications needing to be portable may need to consider a more complex multi-server solution in the MSSQL environment. MSSQL will allocate threads from all instances to every CPU unless the "affinity mask" configuration option specifies which instance may not use a specific CPU.

## **Resource Control**

Both products offer some ability to control resources that any particular query can consume. MSSQL offers a simple "query governor cost limit" option to define the maximum estimated elapsed time for a query.

ASE provides a full-featured Resource Governor to control the estimated or actual I/O amounts, actual row counts or elapsed execution time (via `sp_add_resource_limit`). The DBA can choose whether to enforce these limits prior to execution or during execution and can also specify the action to be taken when a resource limit is breached such as issuing a warning to aborting the transaction and user connection. Limits can be applied to specific logins or named applications and at specific time ranges of the day.

## **Parallel Execution**

Intra-query parallelism refers to the ability of a database server to break a large query into smaller sub-parts and execute each of these sub-parts concurrently. Both products support this form of parallelism.

MSSQL provides a simple parallel query facility that will automatically attempt to split up large SELECT statements and execute each part in parallel when executing on multi-processor machines.

ASE provides a more sophisticated parallel execution facility using worker processes that provide the DBA with the ability to control the degree of parallelism used in various access modes (partition-based or hash-based scans) and in various situations (single-table accesses, joins, sorts, index creation, etc.). The optimizer also considers many factors when generating parallel execution plans (including physical data organization and the current server resources) to help to ensure that parallelism is used appropriately and does not make matters worse on already busy servers or where underlying storage organization does not lend itself to parallel execution.

## **Platform Portability**

MSSQL is only available for Windows operating systems. ASE is available on a number of platforms in addition to Windows, including Intel Linux, Sun Solaris, Hewlett Packard HP-UX, IBM AIX, Compaq Tru64 Unix, Silicon Graphics IRIX, and Mac OS X. The larger platform selection means that ASE applications can be migrated between hardware platforms with little or no change.

## **Storage of Text and Image Data**

MSSQL can store smaller text or image datatype data inline to the table using the `sp_tableoption` parameter "text in row"; otherwise, a pointer is stored to the separate data structure containing the text or image data much in the same manner as ASE. MSSQL can also target large data to specific files and filegroups with CREATE TABLE option `TEXTIMAGE_ON`. ASE stores text and image as a linked list of data pages separate from the table with pointers to data inline to the row. ASE can specify the target device segment for the large object by using `sp_placeobject`.

## **Replication and Warm Standby Servers**

Both ASE and MSSQL have replication and standby database support, and both products allow transaction logs to be shipped to a standby database. Replication support for MSSQL is integrated into the server. ASE offers two replication strategies with ASE Replicator and Sybase Replication Server. ASE Replicator is integrated within ASE in much the same manner as MSSQL's replication and includes a comparable set of system procedures. Sybase Replication Server is a separate and more robust product enterprise-wide data replication between ASE servers and/or heterogeneous database servers, including MSSQL, Oracle, and IBM DB2, and is also used extensively for providing warm-standby capabilities in disaster-recovery scenarios.

For more information on how to add replication support, please see ASE Replicator User's Guide for native ASE replication capabilities or the Replication Server product manuals set for larger scale or heterogeneous replication capabilities.

## Scalability

MSSQL can scale up to 32 processors using Windows 2000 Data Center or 8 processors using Windows 2000 Advanced Server. To support larger systems, MSSQL uses a feature called Federated Database Servers (FDS) to provide horizontal partitioning across multiple machines. FDS requires the DBA to manually partition the member tables by key across the member databases and access the virtual table using a distributed partitioned view.

ASE can scale up to 256 processors on Unix machines, depending on the hardware, and does not directly support horizontal partitioning. However, ASE does provide proxy tables, proxy databases, and union views that can be used to simulate certain FDS functions.

For allocating large amounts of shared memory, both ASE and MSSQL take advantage of 64-bit addressing support by the operating system. Through Address Windowing Extension (AWE), MSSQL can allocate as much as 64GB of shared memory on Windows 2000 Data Center. However, when AWE is used, dynamic memory management cannot be used. For ASE, where 64-bit addressing is supported (Solaris, HP-UX, AIX), maximum shared memory size is several terabytes and ASE's dynamic management can be used.

Both MSSQL and ASE support High Availability systems through the use of clustered servers. Please see the manual Using Sybase Failover in A High Availability System for further information on the use of these fault tolerant features in ASE.

MSSQL allows up to only 16 distinct instances simultaneously on a single machine. There is no such limit enforced on the number of ASE instances that can run on a single machine. Both products, however, are limited by practical considerations such as available memory.

## Summary

Though there are some differences in server structure that were introduced over time, the fundamental structure of the two products is really quite similar. Where differences exist, there are often analogous structures in each product, meaning fewer difficulties will arise during application migration due to server structure. See the manual What's New in Sybase Adaptive Server Enterprise? for a comprehensive overview of features added to successive ASE releases from 11.0 through 12.5.

## Administration

In this section, the administrative differences between the two products are considered.

### Storage Management and Database Creation

MSSQL maps a database to a distinct set of physical data files and filegroups. ASE utilizes database devices (e.g., operating system files, Windows disk partitions, or Unix raw partitions) where each device can be used by one or more databases, and a database can use space from one or more devices. When creating a database the DBA chooses exactly how much space is to be used from each device. Logical areas of storage are managed via file groups within MSSQL and segments within ASE databases.

MSSQL can be configured to silently increase the size of its disk file (data and log) to accommodate growth using the FILEGROWTH clause of the CREATE DATABASE command. ASE can not increase the size of its database devices automatically, but they can be manually resized larger at a later time. MSSQL also allows for a database to be resized lower whereas ASE does not.

Both products benefit from the performance and recoverability features of RAID devices. In addition, ASE has its own built-in mirroring commands for environments without RAID or Logical Volume Management (LVM).

## Database Transaction Log

Fundamentally both products have similar transaction logging mechanisms, with each database having its own write-ahead transaction log to capture database events, such as inserted, updated and deleted rows as well as allocation and deallocation of data and index pages.

The MSSQL transaction log is implemented as a set of virtual logs on one or more physical data files which autogrow in a round-robin fashion. The log files can also be set to shrink automatically to reclaim space after a backup.

Within ASE, the transaction log is an integrated part of the database and is actually a system table called syslogs created on its own segment named logsegment. The devices for the segment logsegment are specified at database creation, but may be changed later via ALTER DATABASE or sp\_logdevice. The DBA can decide whether the transaction log, either for performance or recoverability, should be stored on separate database devices from the rest of the database's data.

## Database Server Configuration

Both ASE and MSSQL use the sp\_configure system procedure to specify server-wide options, but the set of configuration options differs between the two products. For a comparison of configuration parameters, see Server Configuration Option Differences in Appendix B.

MSSQL requires the RECONFIGURE command as a necessary last step to activate the changed parameter(s). Most ASE parameters are dynamic, meaning they take effect immediately after executing sp\_configure. For both servers, a stop and restart may be required for certain static parameters to take effect.

Both products store current configuration parameters in the sysconfigures table, while current runtime values are reflected in syscurconfigs. In ASE, a text configuration file also stores the current configuration settings, conventionally called "SERVER\_NAME.cfg" for an ASE server named "SERVER\_NAME". This file is updated each time a parameter is changed with sp\_configure and it can also be edited manually. The configuration file may be used to share configuration settings between servers, to start servers with different pre-set configurations, or to start a server with a previously known good configuration file should the server fail to properly start after a parameter change.

## Setting Database Options

To set database options, ASE uses the sp\_dboption procedure. MSSQL still supports this procedure but is phasing it out in favor of enhancements to the ALTER DATABASE command. The ASE version of this procedure requires a CHECKPOINT command to be run in the database in order for the changed options to take effect. For a comparison of database configuration options, see Database Option Differences in Appendix B.

## Backup and Restore

Both products offer very similar commands for backing up and restoring databases and transaction logs. MSSQL currently favors the commands BACKUP and RESTORE while ASE uses DUMP and LOAD. In actuality, MSSQL provides synonyms for its older DUMP and LOAD commands to point to the newer BACKUP and RESTORE, but there are plans to remove the older commands in a future version.

MSSQL backups can be used to selectively restore damaged files or filegroups without restoring all the files in a database. MSSQL provides for striping of database files across a maximum of 64 backup devices.

ASE backups are performed by the Backup Server to offload processing from the ASE server itself. Backup Server dumps an entire database image (i.e., all the used pages in the database) to up to 512 simultaneous dump devices (tapes or files), and that image must be restored in its entirety (that is, no partial loads).

MSSQL transaction logs are backed up and restored in the same manner as ASE and both have analogous recovery models. The MSSQL "simple" recovery model (set with ALTER DATABASE) truncates the transaction log when a checkpoint occurs and is analogous to the ASE database option "trunc log on chkpt". Both products allow transaction logs to be restored to a specified point in time, though MSSQL also has recovery to a specific point of work via a marked or named transaction.

Through the MSSQL differential backup option, only the pages that have changed since the last backup are recorded. ASE does not provide this feature, however ASE does have an option for dump file compression that can reduce the dump file size by upwards of 80% and which can substantially reduce the time to perform a backup. In addition, ASE provides the QUIESCE DATABASE command to suspend update activity for short periods of time where a high-speed disk copy program can make backups of the database devices. Third party vendors provide these programs in support of 24x7 operations of VLDB databases.

The MSSQL backup commands require the type of media to be specified (e.g., tape, disk, or named pipe) unless a logical dump device is used. The ASE Backup Server will automatically sense the type of device (tape or disk) and will act accordingly. For more information on option differences between the BACKUP and DUMP commands, see DBA Command Differences in Appendix B.

### **Data Import and Export**

Both products provide facilities for importing data from and exporting data to files. The bcp (Bulk Copy) utility provided by both products shares some common options including exporting the data using a native binary format (the -n option) or human-readable character format (the -c option).

MSSQL includes Data Transformation Services (DTS) to transfer data from tables or queries between OLE DB and ODBC data sources. These services are not available natively in ASE, however Sybase partners with companies to provide Extract, Transform, and Load (ETL) services.

Unfortunately the native binary files produced by the bcp -n native option cannot be reliably used to migrate data from MSSQL to ASE due to differences in how the two products store data values. Therefore the -c character option should be used to create portable and human-readable values for number, binary, and datetime datatype values. ASE provides direct data import/export using CIS and the services of Enterprise Connect / Data Access (ECDA).

### **Job Scheduling and Alerts**

MSSQL works in conjunction with the SQL Server Agent Service to schedule jobs, send alerts, and interact with operators (people assigned to manage the server). Scheduling of scripts or T-SQL batches for one-time or repetitive execution is configured using Enterprise Manager, SQL-DMO (Distributed Management Objects), or T-SQL (e.g. sp\_add\_job). SQL Server Alerts is a facility offered by SQL Server Agent which constantly monitors the Windows application log for MSSQL events and can perform some action in response to their occurrence.

Currently ASE does not provide these services natively; rather the DBA would need to configure software outside the ASE server to perform similar tasks. For example, the operating system typically offers task scheduling services (via Unix cron or Windows at commands) while alert handling can be achieved via a user-written log scanner or by a more comprehensive system management package supplied by a third party. Sybase is looking at adding a native job scheduler in a future version of ASE.

## Space Monitoring

MSSQL sends events to the Windows application log when data and logfiles grow or shrink. If an alert has been defined for that event, the SQL Server Agent can respond by notifying operators, executing a job, or forwarding the event to a remote Windows server. Files and filegroups are usually configured to autogrow as more space is required, so monitoring for free space often comes down to looking at the free space left on the Windows filesystem. One method often used to monitor free disk space is to write a custom job that the SQL Server Agent executes periodically.

ASE provides a built-in ability to monitor and respond to fluctuations in database space via its free space thresholds feature. For each database transaction log there is a last-chance threshold to monitor the minimum amount of space required for dumping the transaction log before it fills completely. A user can also define additional free space thresholds on any segment in the database. When thresholds are crossed, the server will automatically execute the designated stored procedure. These procedures can execute remote and extended stored procedures, invoke shell scripts, email the DBA, add more disk space, or dump the transaction log as appropriate.

## Open Transaction Monitoring

MSSQL's DBCC OPENTRAN command identifies the oldest active transaction and the oldest distributed and non-distributed replicated transactions, if there are any, in each database. ASE stores the equivalent oldest active transaction information in the table master..syslogshold. In addition, all active ASE transactions are stored in the table master..systransactions.

## Security Features

Security in a database environment can be controlled at the network, server, and database levels. To access a database, a client may establish trust with the server through a security mechanism involving identification and authentication via a central authority such as a directory service and then through the database server itself. Once a client gains access to the database, database-level security features are applied.

Both ASE and MSSQL can use network-based security through directory services. MSSQL can receive connections via windows authentication through Microsoft Active Directory with Windows 2000 and natively supports lightweight directory access protocol (LDAP) over secure sockets layer (SSL). ASE provides directory services for storing the network location of the ASE servers using LDAP or Distributed Computing Environment (DCE) directory services. SSL can also be used for the client-to-server session providing message protection.

Both products offer the facility to use trusted connections, authentication services on the Windows platform, such that Windows users and groups can be given database access without needing an additional password. ASE also offers CyberSAFE Kerberos and DCE security services for third-party authentication.

Inside the database, logins and users form the basis for imposing further discretionary access controls in the form of roles and permissions. MSSQL forms logon identities from local and global group accounts from a Windows domain as the foundation of its database security. MSSQL organizes these login accounts using user-defined roles and grants them permissions with GRANT and REVOKE. ASE retained the original concept of groups, which is a logical grouping of users in the database (for example, marketing, sales, etc.), and this is functionally similar to the MSSQL domain-defined groups. ASE DBAs can grant user-defined roles to logins and other roles.

For both products, roles can be organized in a hierarchy for efficient authorization management; however the commands differ in syntax and scope. ASE uses the SQL92 CREATE ROLE and GRANT ROLE commands; MSSQL uses sp\_addrole and sp\_addrolemember. The ASE system roles are server-wide in scope to ensure consistency and easy administration and include sa\_role, sso\_role, and oper\_role.

System roles in MSSQL are both server-wide and database-specific, so DBAs must take care to ensure that security is enforced consistently for multi-database applications. The MSSQL server-wide roles include sysadmin, serveradmin, setupadmin, etc., and the database-wide roles include db\_owner, db\_accessadmin, db\_datareader, db\_datawriter, and others.

MSSQL allows applications to enable additional roles for a user running a specific application. The DBA can use sp\_addapprole and sp\_setapprole to create and enable application roles. The ASE equivalent is user-defined roles with the password known only to the application.

ASE has a facility for controlling access to applications called the Application Context Facility (ACF). An application profile is first created by assigning attribute and value pairs. A profile can be initialized at login-time through a login trigger. The profile can then be used with Row Level Access Control (RLAC) to restrict data access to individual rows using access rules. ACF eliminates the need to maintain or write application logic to restrict data access since it can be easily maintained inside the database server with very little overhead.

### **Auditing**

MSSQL provides a graphical tool for monitoring any kind of database event using the monitoring tool SQL Profiler, which sends a trace of user and server events to a file or table. MSSQL also has audit modes to analyze classes of events ("c2 audit mode" option).

ASE has a comprehensive and powerful C2-compliant audit subsystem that allows audit trails of server activity to be captured at whatever level is appropriate for the organization. The audit subsystem has its own set of system procedures (e.g. sp\_audit) and the sybsecurity database to capture and manage the audit trail.

### **Server Trace Flags**

Both products offer a number of trace flags that are used to alter the behavior of the server in certain ways. Many of the commonly used trace flags have virtually identical behavior between the two products (for example 302, 310, 3604 and 3605). Others (such as 1204), while not identical, cause similar information to be displayed. As trace flags are low-level internal switches that often change from one release to another, it should not be assumed that any particular trace flag would work identically across both products. Sybase Customer Service and Support can provide advice on the use of particular trace flags.

### **System Stored Procedures**

The MSSQL product adds several of its own unique system procedures to the original set inherited from Sybase SQL Server. Examples include sp\_dbremove, sp\_helpsql and sp\_procoption. In addition there are a number of system stored procedures (e.g. sp\_dboption, sp\_serveroption) that have slightly different usage between the two products. These differences may mean that some application and administration scripts may need to be updated to account for these differences. A list of the potentially problematic system procedures is supplied in DBA Command Differences.

## **Character Set Support**

There are some slight differences in how each product handles character set and Unicode support. MSSQL allows character set and sort order combinations (collations) to be set at the database, table, and even the column level. Unicode is supported in addition to any other chosen collations.

ASE allows one character set and sort order to be set at the server level, but can also support many languages for system and user messages and default date format as required by multi-lingual applications. Unicode support is also provided through the UTF-8 character set, including support for UTF-16 encoding for unichar and univarchar datatypes. The COMPARE() and SORTKEY() functions enable application developers to work in sort orders other than the server's default.

## **sp\_tableoption Procedure**

MSSQL provides this stored procedure to allow tables to be marked as RAM resident or to be marked so that bulk loading will acquire a table lock rather than multiple row locks. With ASE you can define named data caches and then bind tables, indexes and databases to those caches for a similar effect. For bulk loading, ASE reduces locking problems by providing for automatic lock promotion from row or page locks to a single table lock.

## **Start-up Procedures**

MSSQL allows certain stored procedures to be marked as start-up procedures (via sp\_procoption) which the server will run automatically after it has started. ASE does not offer a direct equivalent but a DBA can augment the startup script on Unix or Windows to execute ASE commands or procedures.

## **The DBCC Command**

Both products offer the dbcc command for consistency checking of databases and other system level operations. Both offer parallel DBCC capabilities, running on as many processors as possible with multiple DBCC commands running at once. For MSSQL, DBCC is no longer required for regular maintenance except as a check before major system changes. For ASE, DBCC is still recommended for regular maintenance and has been extended with the dbccdb system database to check and fix page linkage, pointers and page allocation errors using DBCC checkstorage.

Keep in mind that due to its system level nature, the DBCC subcommands differ slightly between the two products. While core sub-commands (such as checkalloc) are the same, both products offer their own sub-commands; DBA scripts that use commands specific to one product (such as MSSQL's DBCC SHRINKDB) may need to be modified before they can be used with ASE. A list of potential problems is provided in DBA Command Differences in Appendix B.

## DBA Tools

DBAs have a choice of interfaces when administering database servers. Some DBAs will prefer to use T-SQL commands and scripts to perform administration as this gives them power and flexibility. Others will prefer to use an administration utility with a graphical interface because of its simplicity and ease of use. In practice, DBAs use both approaches depending on the need at the time.

Both products provide graphical DBA tools with MSSQL's SQL Enterprise Manager and ASE's Sybase Central™. These graphical environments are broadly equivalent in terms of function although they differ in specific usage and graphical display. In addition, a number of third party tools are available for both products with some tools, such as Embarcadero's DB Artisan, being capable of managing ASE and MSSQL servers in one interface.

Both ASE and MSSQL also provide a command line utility called isql that allows interactive SQL sessions as well as SQL scripts to be run against a server. The utility is very similar across the two products with many common command switches. MSSQL also has a utility called osql, which is very similar to isql but uses ODBC datasources to access the server. isql should be used in scripts that must be run against the two products for portability.

MSSQL and ASE possess a number of tools to provide administrative and developer support.

- Network configuration – MSSQL uses the Client Network Utility and the Server Network Utility to add new MSSQL aliases and supported network protocols. ASE has the dsedit program to configure network connections to servers. In ASE with the Security and Directory Services option, dsedit can be used to add a new ASE server name and network addressing information to a company's LDAP server.
- Query design – To design and test queries interactively, the MSSQL SQL Query Analyzer and Index Tuning Wizard are used together. ASE provides SQL Advantage, isql, and jisql (a Java-based graphical tool) to run statements, batches, and scripts interactively. For stored procedure and trigger debugging, an ASE utility called SQL Debugger can step through lines of code, set breakpoints, and attach and detach to tasks until analysis is complete. ASE's SQL Expert option allows for the analysis and optimization of SQL statements.
- Monitoring – MSSQL provides a system trace facility called SQL Profiler that can collect a large amount of raw data about server events. This information can be used to deduce performance problems with sufficient effort. It also provides system stored procedures (such as sp\_lock and sp\_monitor) and an integration into the Windows Performance Monitor. ASE offers the comprehensive monitoring system procedure sp\_sysmon as well as simpler procedures such as sp\_lock and sp\_monitorconfig. It also provides an integration into the Windows Performance Monitor and a comprehensive server monitoring facility via Sybase Monitor Server (included with ASE) and the graphical monitor applets within the Sybase Central management tool. The ASE MDA Monitoring feature accumulates information about which tables and indexes are chosen by the optimizer to satisfy query execution.
- Server Start-up/Shutdown – For MSSQL, the SQL Server Service Manager is used to start and stop the server and manage services associated with MSSQL such as the Distributed Transaction Coordinator, Microsoft Search, SQL Server OLAP Service, and SQL Server Agent. ASE services are controlled in different ways depending on the platform. On Windows, individual servers are started and stopped using the Windows Services Manager. On Unix, servers are started using the startserver program in conjunction with a shell script and stopped using the T-SQL SHUTDOWN command.

## Development

There are several differences between the servers in behavior, T-SQL language and in the development features they offer. This section lists some of the more prominent ones, but refer to Appendix A for a more complete list.

### Behavior, Limits, and Additional Components

- **DDL in Transactions**

By default, MSSQL allows Data Definition Language (DDL) statements, such as CREATE TABLE, to be part of a transaction. This feature can lock system tables, such as sysobjects, resulting in performance degradation. By default, this feature is disabled within ASE and may be turned on for a database using the "allow ddl in tran" database option.

- **Altering Objects**

MSSQL allows a stored procedure, trigger or view to be updated in place via the ALTER PROCEDURE, ALTER TRIGGER, and ALTER VIEW statements. This is not possible in ASE; objects must be dropped and then re-created when they need to be changed.

The abilities inherent in the ALTER TABLE syntax (adding/dropping/changing columns, types, defaults) are fairly compatible between the two products. There are a few features not present in one or the other. MSSQL's ALTER TABLE...ALTER COLUMN COLLATE syntax allows you to specify a collation sequence for a column; this feature is not supported in ASE. ASE's syntax allows properties related to table partitioning and storage to be specified; these are not supported in MSSQL. For a more complete list of differences, see Appendix A and T-SQL Statement Differences.

- **Name Resolution in Stored Procedures**

MSSQL uses deferred name resolution in stored procedures, meaning that objects referenced by a stored procedure do not need to exist when the stored procedure is created. In order to ensure run-time efficiency and integrity, ASE requires all referenced objects exist when the procedure is created.

- **Bound Connections**

MSSQL provides for a bound connection where a transaction may be separated from a user connection allowing for multiple user connections to participate in (be bound to) the same database transaction. ASE provides some of this functionality through the ODBC driver interface and also through XA-compliant transaction servers using ASE's Distributed Transaction Manager facility (DTM). DTM provides the most compatibility as a migration alternative to bound connections for use of the "suspend" and "join" semantics of XA-compliant transaction managers.

- **Object Identifiers**

MSSQL allows up to 128 characters in an object identifier, such as a table, column, or view name. ASE currently enforces the SQL92 specification of a maximum of 30 characters. MSSQL allows identifiers that have non standard characters or are reserved keywords to be enclosed using double quotation marks or square brackets (as in "SELECT \* FROM "Table X" where [order] = 123"). ASE currently only supports double quotation marks using the SET quoted\_identifier option. Sybase expects that a future release of ASE will support 128 character identifiers as well as bracketed identifiers.

Most MSSQL servers are case-insensitive, allowing objects to be referenced using upper, lower or mixed case (e.g., "AUTHORS" is the same as "Authors" or "authors"). ASE servers can be installed using either a case-sensitive or case-insensitive default sort order. If the ASE server uses a case-sensitive sort order, such as "binary", then object identifiers are also case-sensitive (e.g., "AUTHORS" is different from "Authors"). To ensure portability, use a case-insensitive sort order in the ASE server, such as "nocase".

- **Table Limits**

MSSQL allows up to 256 tables to be included as joins or subqueries in a single SELECT statement. ASE allows up to 64 tables for a single query. Similarly, MSSQL allows 1024 columns in a table whereas ASE allows 254 for variable length allpages-locked tables and 1024 for datapages and datarows tables.

- **Object Storage Parameters**

MSSQL uses the "fillfactor" server configuration option to influence how full the index pages should be when the index is created. This parameter can also be specified during CREATE INDEX but MSSQL recommends only using the server-wide configuration for fillfactor as it believes this default is best for most cases. ASE offers FILLFACTOR as well as MAX\_ROWS\_PER\_PAGE, EXP\_ROW\_SIZE, and RESERVEPAGEGAP options to the CREATE INDEX and CREATE TABLE commands.

- **Indexed Views**

MSSQL provides for an index to be created on a view. This feature creates a snapshot of the data that a view describes at the time of index creation, making the view no longer virtual. MSSQL must keep the data synchronized via the association made through the SCHEMABINDING clause of the CREATE VIEW command. ASE does not provide this functionality.

- **Computed Columns**

ASE does not currently provide the MSSQL computed columns feature of the CREATE TABLE command. A computed column is a virtual column that is defined through an expression involving one or more other columns in the same table, a function, or another column. For now, a computed column in ASE can be simulated using an actual column and a trigger or some other method. Sybase expects that a future release of ASE will support computed columns.

## **Language, Functions, Procedures and Commands**

- **T-SQL Language**

Typically, language differences account for most of the major problems when migrating applications from one relational database management system to another. However, in the case of MSSQL and ASE, the problems are substantially reduced by their shared heritage. Both database servers use Transact-SQL (T-SQL) as their data manipulation, data definition and data control language. The two products use slightly different versions of this SQL dialect (which is based upon ANSI SQL 92), however the differences are minor when the overall language is considered.

Aspects of T-SQL that can cause problems for migration or portability relate to ASE and MSSQL-specific extensions that were added over time. The following provides a discussion on some T-SQL differences that will need to be addressed; please see Appendix A for a more complete list of T-SQL features, differences and suggestions for migration.

### **The primary T-SQL features that may need reworking for migration include:**

- Datatypes – there are a small number of datatype differences between the two products. Specifically MSSQL's character and binary datatypes allow lengths up to 8000 characters while ASE's character and binary types range from 1960 to 16296 bytes, depending on the logical page size used to create the database and the locking scheme for the table.
- MSSQL has some additional datatypes not present in ASE. MSSQL has cursor (see later discussion on cursors) and uniqueidentifier (use ASE's identity instead).

To handle 8-bit integers, MSSQL provides bigint. In ASE, use the numeric(19,0) datatype instead.

- For a generalized 8-byte container that can hold int, decimal, char, binary, and nchar values, MSSQL provides the datatype sql\_variant, though there are a lot of caveats to this datatype. In ASE, you can use any appropriate character datatype.
- In addition, the table datatype holds an entire result set, as can be generated by a SELECT command. To simulate the table datatype, the application will need to be reworked, perhaps using temporary tables to store intermediate results.
- Identity and Uniqueness – Both products offer the identity column attribute that indicates that the server should assign a unique value for this column whenever a row is inserted. The MSSQL implementation allows a seed and increment to be specified and the column's datatype may be either tinyint, smallint, int, decimal(p,0) or numeric(p,0). MSSQL generates a value to insert into a table by adding the increment to the last inserted value.

In contrast, ASE does not allow a seed or increment to be specified (although the DBA can place a seed value in the table using `set identity_insert`) as the seed and increment are always 1. An identity column's datatype must be `numeric(p,0)`. ASE caches a block of identity values in memory (instead of generating them from disk one at a time) in order to increase performance. Because this block of values is memory-resident, it can be lost when the database is suddenly or abnormally shut down, resulting in gaps between identity values. To prevent large gaps, you can specify the block size to cache using the `WITH IDENTITY_GAP` option of the `CREATE TABLE` and `SELECT INTO` commands. Identity values can be explicitly inserted or updated using the commands `set identity_insert` and `set identity_update`.

The MSSQL identity column can be referred to by the pseudo name `IDENTITYCOL` whereas the ASE version uses the pseudo name `SYB_IDENTITY`. ASE has the "auto identity" database option to automatically include the `SYB_IDENTITY` column in each table that is created without one.

MSSQL provides a `uniqueidentifier` datatype (to support the use of Microsoft GUID identifiers). ASE does not support this datatype.

An application that makes use of identity columns may need to review its usage for these differences. If it cannot be migrated directly, alternatives, like a custom counter using an integer column and an insert trigger, can be used instead.

- ANSI Join Syntax – MSSQL supports the ANSI 92 SQL join operators of `JOIN`, `CROSS JOIN`, `INNER JOIN`, `LEFT OUTER JOIN`, `RIGHT OUTER JOIN`, and `FULL OUTER JOIN` as well as the traditional T-SQL join syntax using `'='` for inner joins and `'*='` and `'=*` for left and right outer joins. ASE supports all the above clauses except `CROSS JOIN` and `FULL OUTER JOIN`.
- SELECT Lock Syntax – MSSQL allows the locking strategy for a `SELECT` statement to be specified via the `NOLOCK`, `UPDLOCK`, `TABLOCK`, `PAGLOCK`, and `TABLOCKX` qualifiers. ASE does not support this syntax, but provides alternatives using the `HOLDLOCK`, `NOHOLDLOCK`, and `AT ISOLATION` keywords as well as the `LOCK TABLE` command to lock a table in exclusive or shared mode.
- Rowsets – The MSSQL `SELECT` statement allows the use of pseudo-database objects called rowsets resulting from OLE DB queries. Rowsets are not supported in ASE.
- BULK INSERT command – MSSQL provides this command to perform a T-SQL version of the `bcp` utility. ASE does not provide this command; however `bcp` can be executed by using the system-supplied extended stored procedure `xp_cmdshell`. In addition, if the bulk insert is to be performed across servers, ASE can use the `SELECT INTO EXISTING TABLE` command to insert the data using the bulk insert protocol.
- Cursors – Both MSSQL and ASE implement cursors as defined by the SQL-92 standard; however MSSQL extends their capabilities and syntax in a number of proprietary ways. Migration problems caused by MSSQL language enhancements include scrollable cursors and use of global or local cursors as parameters to stored procedures. ASE allows neither global nor local cursors as parameters to procedures. While MSSQL cursors can scroll forward, backward, to the first record, and the last, ASE cursors can currently only scroll forward. Sybase expects to support scrollable cursors in a future release.

To provide the scrolling capability, an MSSQL cursor does not lock the underlying rows, potentially allowing rows to be modified by other transactions. ASE ensures data integrity by retaining a lock on the current page or row.

MSSQL provides the global variable `@@FETCH_STATUS` to monitor the status of the most recent cursor fetch. ASE provides `@@SQLSTATUS` with different status values. See Global Variable Differences in Appendix A for the value differences.

- Variable Assignment – The `SELECT` command in both servers is used for variable assignment (e.g., `SELECT @variable = 123`). In addition to `SELECT`, MSSQL provides `SET` (e.g. `SET @variable = 123`). `SELECT` is currently recommended for portability, however Sybase expects to offer the `SET` command for variable assignment in a future release.

- INSERT with DEFAULT option – MSSQL allows the DEFAULT option in the INSERT command VALUES clause to indicate that a default value be used when inserting into a table. ASE currently does not provide this syntax, so the ISNULL() function can be used instead.
- LIKE operator – The SQL LIKE operator works almost identically between the two products with the exception of its handling of trailing spaces. The ASE LIKE operator ignores trailing spaces in the search pattern (i.e. "%A%" is the same as "%A% ") whereas the MSSQL LIKE operator will attempt a literal match for the space. In ASE to search for trailing spaces, replace each trailing space in the pattern with "[ ]" (i.e., "%A%[ ]").
- System Functions, Global Variables, Session Settings, and Keyword Lists

ASE and MSSQL provide many common functions (e.g. GETDATE()), global variables (e.g. @@VERSION), session-based settings (e.g. SET statistics io), and keywords. For more information on the differences, please see Appendix A.

- Extended Stored Procedures (ESPs)

For MSSQL, ESPs are part of the Open Data Services layer between the relational engine and the server net-libraries. For ASE, ESPs are provided through a separate XP Server. While ASE offers ESPs in a similar manner to MSSQL, the set of system-supplied ESPs is slightly different. In particular, the MSSQL xp\_grantlogin (also sp\_grantlogin), xp\_loginconfig, xp\_logininfo and xp\_revokelgin ESPs have been implemented as system stored procedures in ASE (called sp\_grantlogin, sp\_loginconfig, sp\_logininfo and sp\_revokelgin). Some MSSQL ESPs (e.g. xp\_msver and xp\_sprintf) are not available in ASE, but many are implemented (xp\_cmdshell, xp\_deletemail, xp\_enumgroups, xp\_findnextmsg, xp\_logevent, xp\_readmail, xp\_sendmail). Please see DBA Command Differences in Appendix B.

- Optimizer Hints and Abstract Query Plans

Both products allow optimizer hints to be supplied as part of the SELECT, UPDATE, INSERT and DELETE commands (e.g. "SELECT \* FROM t1 (<hint>)", though the set of hints is different. MSSQL hints are mainly concerned with the locking and join strategies used. The ASE hints include information about indexes, pre-fetch I/O buffer pool sizes, parallelism, and data cache replacement strategies (aspects of query processing not relevant to MSSQL).

One advantage of ASE over MSSQL is the ability to specify the exact query plan the optimizer should use through abstract query plans. Abstract query plans decrease the optimization time as the decisions on join order, indexing, scanning methods, and many other components are specified in the plan. See "Introduction to Abstract Plans" in the ASE Performance and Tuning Guide. When migrating queries between databases, all hints should be removed and the queries re-tuned in the appropriate environment as required. If problems arise in the query optimization, abstract plans may be used.

- RAISERROR Command

The MSSQL RAISERROR command uses a similar construct as the C language printf() function for specifying the error message so datatype conversions can be explicitly controlled. The ASE RAISERROR uses a simpler syntax with placeholders (e.g. RAISERROR 20001 "Error in %!", @var) and automatically converts argument types during variable substitution.

MSSQL provides a severity argument, WITH SETERROR clause, and WITH NOWAIT clause which are not provided by ASE. MSSQL provides the WITH LOG clause to indicate that a copy of the message should be placed in the server log. In ASE, the with\_log option can be specified when the message is added to the server via sp\_addmessage for the same result. sp\_addmessage also provides for adding the message in multiple languages.

- Distributed Transactions

To start a transaction that involves several MSSQL servers, the DISTRIBUTED clause must be added to the BEGIN TRANSACTION command to instruct the server to use MS DTC for transaction coordination. This clause is not used or necessary in ASE as it will implicitly coordinate a distributed transaction between several ASE servers.

## **Programming Interfaces**

- Windows APIs: OLE DB, ADO, ODBC, DB-Library, and CT-Library

Both products allow applications to access the servers through native OLE DB, ODBC, JDBC, and DB-Library drivers. ASE can also be accessed via CT-Library (ASE's descendent API of DB-Library). The ASE OLE DB Provider supports many features including backward and forward scrollable rowsets, Unicode, threading, the IDBSchemaRowset interface for ADO/OLE DB consumers, and the Rowset interface. For a complete list of supported OLE DB types and interfaces, see the section ASE OLE DB Provider Support in Appendix A.

MSSQL's DB-Library contains some additional function calls compared to ASE's version. Please refer to the Sybase manual Open Client DB-Library/C Reference Manual for further details on the functions.

The ASE ODBC drivers support all ODBC Core and Level 1 functions as well as compliance with Level 2 and 3 functions, depending on the driver. See the section ASE ODBC Driver Support in Appendix A for more information on functions and data type compliance.

- User-Defined Functions

Both products enable developers to augment the system-supplied functions with user-defined functions (UDFs) using the CREATE FUNCTION command. In MSSQL, UDFs are written using the T-SQL language and they can return most datatypes including table.

In ASE, UDFs are written in Java and conform to the ANSI SQLJ Part 1 specification. To use Java classes, they must first be installed as objects in an ASE database. Static methods of Java classes can be invoked as function calls and are executed within ASE's built-in Java Virtual Machine (JVM). UDFs are mapped to Java static methods and can return most datatypes; however, there is no equivalent for the MSSQL table datatype. Java in ASE can also be used to create SQLJ stored procedures. ASE Java-based UDFs and stored procedures are able to manipulate XML content, access content from the web, make network connections, send email, and perform many other activities.

- XML

Both products offer XML support, but each implementation is slightly different. For MSSQL, XML Data is retrieved using Xpath queries or the SELECT...FOR XML clause. The OPENXML() rowset function is also used in joins and for data manipulation. XML views provide support for XML-Data Reduced (XDR) schema mapping to tables, views, and columns, and are also referenced using XPath.

ASE can store, index and query XML data natively. Users can query stored XML data or get XML results from stored XML or SQL data. ASE's current XML support is provided through the Java capabilities as an extender, which can be run standalone or integrated into the ASE engine as a Java classes. Querying of internal (stored in text datatype columns) and external (stored on web or in files) XML documents is provided by the XML Query Language (XPATH) within the ASE Java classes. Data extraction and update capability is done through the ResultSetXML Java class. Users can use the following fragment of code to achieve results similar to that of FORXML clause of MSSQL Server: SELECT TO\_SQLX (" select query user wants to execute goes here ...") Sybase expects to support the SELECT...FOR XML clause in a future release. Also in the upcoming release XML data management capabilities are being integrated into the core engine.

## Additional Facilities

- Full-Text Search

MSSQL provides advanced searching through Microsoft Search Service, allowing text searching for phrases, the capability for matching inflection, and searches on image columns. Searching is provided through T-SQL predicates using the functions FREETEXT(), FREETEXTTABLE(), CONTAINS(), and CONTAINSTABLE().

ASE provides many of the same advanced searching plus additional features through the Full-Text Search Specialty Data Store engine. Advanced searches, synonym creation, results ranking, search topic creation through document clustering, administration (data store backup, restore, system reporting) are some of the features offered. See Full-Text Search Specialty Data Store User's Guide for details on installing the server, configuring for text searches, and performing searches.

- OLAP and Data Warehousing

MSSQL provides Analysis Services for data mining and to create and analyze relational and multidimensional data sources. ASE does not provide this support natively; however Sybase provides Adaptive Server IQ Multiplex for high-end data warehouses.

- English Query

In addition to T-SQL, MSSQL allows you to convert fully normalized MSSQL databases into applications that use English sentences to submit requests for data. ASE does not include this capability. Since there is a one to one mapping of objects and attributes in an English Query application to tables and columns, and since a MSSQL schema is required before starting one of these applications, these queries can be mapped back to Transact-SQL queries for reverse-engineering.

- Meta Data Storage

MSSQL Meta Data Services stores database and object metadata generated from information models to be used for applications in the system database msdb. It can provide meta-data information about a database's objects to third-party design and application. ASE does not have a comparable component; however, Sybase provides the PowerDesigner product for data modeling which can also be used to import, export, and store database metadata.

## The Migration Process

In this section, a systematic process is presented for migrating an application from MSSQL to ASE. The result of following this migration process should be a completely migrated application running in ASE.

Sybase provides a schema and data migration utility called PowerTransfer as a plug-in for its tool PowerDesigner®. PowerDesigner uses an ODBC interface to reverse engineer a schema from any of 30 different databases including MSSQL, Access, Oracle, DB2, FoxPro, and others. PowerTransfer then transfers the schema and data to ASE through the Open Client bulk library API for fast data loading.

### Step 1 Migrate the Server Structure

The aim of this step is to create an ASE server structure that is capable of supporting the existing MSSQL application in an ASE environment. Luckily, because the structure of the two products is so similar, this is unlikely to cause any significant migration problems.

#### Step 1.1 Create the ASE Server

An ASE server should be created to take the place of the MSSQL server. This is a straightforward step that can be performed using the graphical interface Server Config program on Windows or asecfg on Unix.

### **Step 1.2 Configure the ASE Server**

The server-wide options for the ASE server should be set using `sp_configure` or Sybase Central. These options should broadly match those of the existing MSSQL server (with any appropriate mapping for configuration settings that are different between the two products). See Server Configuration Option Differences in Appendix B for the option differences.

### **Step 1.3 Allocate Storage Space**

Storage space for databases should be allocated in the new ASE server. This normally involves creating a script of DISK INIT commands that are equivalent to the disk file statements used in the MSSQL. Devices should be created for both the data areas as well as the transaction logs.

### **Step 1.4 Create Server Logins**

The server logins for users of the databases should now be created. Passwords will need to be reassigned for ASE.

## **Step 2 Migrate the MSSQL Database**

The aim of this step is to migrate the MSSQL databases to the ASE server. The result of this step should be an entire empty database structure that is ready to support the application.

### **Step 2.1 Create the Databases**

The databases can now be created, using the scripts from the MSSQL server, if these are available, with the storage management clauses updated for ASE. Space should be allocated for both data and log in the CREATE DATABASE command. Once created, the database owner should be set to the appropriate server login.

### **Step 2.2 Set the Database Options**

The options for the newly created databases should now be set using `sp_dboption`. As discussed earlier, many of the database option settings can simply be copied from the MSSQL server. For those options that are not identical, see Appendix B for suggestions on resolving mismatches.

### **Step 2.3 Add Database Users and Groups and the Server Roles**

Add the users and groups to the database, assigning the users to their group as appropriate. Create the user-defined roles in the master database and then grant those roles to the appropriate logins.

## **Step 3 Migrate the Database Schema**

The purpose of this step is to create a database schema that can be populated with application data and then used by applications.

### **Step 3.1 Obtain MSSQL Database DDL**

The first step is to obtain the DDL used to create the database objects in the MSSQL server. If the original scripts are not readily available, a DBA tool (such as Microsoft SQL Enterprise Manager or Embarcadero DB-Artisan) or a CASE tool (such as Sybase PowerDesigner) can be used to reverse engineer them.

### **Step 3.2 Create ASE Database DDL**

The MSSQL DDL scripts should now be checked for MSSQL-specific statements and options. Refer to the appendices for portability lists to assist with this process. Where MSSQL specifics are found, change or remove them as appropriate to create the DDL that can be used with the MSSQL server.

### **Step 3.3 Run the DDL Against the Database**

Finally, the updated DDL scripts should be used to recreate the database structure by running them against the new ASE server.

### **Step 4 Review Security and Administration Procedures**

In this step, the object level security and administration procedures for the application environment are migrated to the new ASE server. When this step is complete, the database should have adequate security for the application's needs and it should be possible to administrate the environment effectively. Therefore, the database server should be ready to accept application data and then be used by the application.

#### **Step 4.1 Migrate Security Scripts**

It is now important to review any security-related statements (i.e., GRANT and REVOKE permissions on the database objects to users, groups, and roles) that are part of the application or any other aspect of the application that manipulates security-related objects. Rework those scripts for which there is no direct correlation with ASE's security model.

#### **Step 4.2 Migrate Administration Procedures**

Administrative procedures should be reviewed to ensure that they are optimal and appropriate for ASE, improving any areas that have proved to be less than ideal in the past. In general, existing MSSQL administration procedures should prove satisfactory in ASE. MSSQL-specific commands including BACKUP and RESTORE and UPDATE STATISTICS will need to be reworked. In addition, ASE-specific commands will need to be added, including REORG for datapages- and datarows-locked table reorganization.

If SQL Server Agent is used to schedule maintenance jobs, replace this for ASE to use an operating system scheduler such as cron for Unix or at for Windows.

### **Step 5 Migrate Data**

The data migration step involves extracting data from the MSSQL tables and inserting that data into the newly created ASE tables.

#### **Step 5.1 Extract MSSQL Data and Import into ASE**

There are several ways to extract the MSSQL data fairly easily between the two servers. The chosen method depends upon the amount of data to be moved and the budget available for support tools.

- Use ASE's PowerTransfer tool to stream the data from MSSQL to ASE.
- Use MSSQL's bcp to extract each table into a text file in character mode (with the '-c' and '-6' options) and then use ASE's bcp utility to load the data from the files;
- Use Sybase's InfoMaker tool and its Data Pipeline feature to move the data from MSSQL directly into ASE;
- Use MS DTS to move the data from MSSQL directly into ASE;
- Use a 3rd party product to extract the data into files or to move the data from MSSQL directly into ASE.
- Use ASE/CIS and Enterprise Connect / Data Access (ECDA) to extract data from MSSQL and insert directly into ASE.

In the case of some export approaches (e.g. using bcp), the import and export steps are separate. In general, before performing the import, indexes, triggers and constraints should be removed from the tables in the ASE server to allow the import to execute as fast as possible. The constraints can be reapplied, the indexes rebuilt and the triggers recreated after the import is complete.

## **Step 5.2 Data Validation**

With the data migration complete, data administration staff familiar with the application and its data should check the integrity of the data. Ideally, standardized scripts should be available to perform logical data integrity checks with a minimum of administrator effort. Problems discovered at this stage should be checked against the source MSSQL server data.

## **Step 6 Migrate Application Programs (Queries & Interfaces)**

This step involves migrating the application itself which in turn involves ensuring that the application's DML (i.e., SQL statements) will work against the new database and that the application programming interfaces are available against the new database server.

### **Step 6.1 Check DML for Incompatibilities**

The most fundamental part of migrating the application itself is to ensure that the queries will work correctly in the ASE environment. There are two aspects to this: checking for invalid syntax to allow compilation and checking for semantic differences between the products to ensure correct execution.

T-SQL batches, stored procedures, and triggers should be checked. Information to guide the checking process can be found in the Development section earlier in this document and in Appendix A. Where incompatibilities are found, change or remove them as appropriate for ASE.

### **Step 6.2 Test All Application Queries**

When the queries have been migrated, they should be tested to ensure that their execution in the ASE server is identical to their execution in the MSSQL server.

### **Step 6.3 Rebuild Client Programs**

With the application queries successfully migrated, the surrounding client programs can now be rebuilt if required. Obviously for programs written using 4GL environments like Visual Basic or PowerBuilder, little or no change to the client itself should be necessary provided that any MSSQL specifics were removed from their SQL batches in the previous step. For clients written in languages like 'C', it should be possible to recompile and link them using Sybase's DB-Library as a replacement for Microsoft's DB-Library.

Clients that use the OLE DB and ODBC interfaces to access the database should run against the ASE server without change provided that ASE OLE DB provider or ODBC driver supports the function calls made by the application. Refer to Appendix A for specifics on ASE OLE DB provider and ODBC driver support.

### **Step 6.4 Test Client Programs**

All of the clients should be tested to ensure that their behavior is identical to clients used with the MSSQL server.

## **Step 7 Validate Migration**

The final step is to validate the migration with a thorough testing process. Ideally, there will be an existing automated set of integration, system and user acceptance tests that will allow easy validation of the migration. If such tests are not available, then an alternative testing process must be performed in order to ensure that the migration has been completed successfully.

The testing should include validating:

- User interface transactions
- Batch processing
- Administration procedures
- Disaster recovery
- Application performance obtained

With all of this complete, the application can be considered to have successfully migrated from Microsoft SQL Server to Sybase Adaptive Server Enterprise.

## Summary

This document has attempted to help you migrate your application or ensure its portability between MSSQL and ASE. The two products are very similar; however, for the DBA and the developer, there are a number of differences that reflect divergence of the two products in later releases.

The migration process between the two is relatively easy and involves:

- Creating a suitable ASE server
- Migrating the application database structure
- Migrating the application's data
- Ensuring that administration and security procedures are appropriate for the new environment
- Checking that application SQL will work in the ASE environment
- Ensuring that programming interfaces will migrate correctly
- Testing the resulting migrated system to ensure compatibility with the original.

If these steps are followed, the migration of your MSSQL application should be fairly straightforward and deliver real benefits quickly.

The process of ensuring portability is simpler than migration and fundamentally involves understanding the areas where the two products differ in order to minimize an application's dependency on such features. The lists provided as appendices should help to guide you during this process.

## Appendix A Portability List: Development Features

In this appendix, several lists are presented describing the development feature differences in the T-SQL language that are likely to be encountered when migrating applications from Microsoft SQL Server to Sybase Adaptive Server Enterprise.

### T-SQL Statement Differences

When migrating T-SQL to ASE from MSSQL the following points must be addressed:

MSSQL T-SQL Statement	ASE Equivalent	Comment
CUBE ROLLUP	SELECT ...GROUP BY	Replace aggregate operators with a number of "SELECT ... GROUP BY" statements for the aggregates that they actually need.
SELECT ... TOP ... [WITH TIES]	SET rowcount n	Use the ASE session level setting to return a portion of a result set.
SET @local variable = expression	SELECT @local variable = expression	Setting variables created through DECLARE should be done with SELECT syntax instead of SET.
N'unicode data'	'unicode data'	The N prefix found before Unicode strings in MSSQL is not required in ASE.
OPENDATASOURCE() OPENROWSET() OPENXML()	No equivalent	If appropriate, replace OLE DB rowsets with CIS distributed queries using proxy tables.
DECLARE cursor name INSENSITIVE	SELECT ... INTO #temp DECLARE cursor name CURSOR FOR SELECT ... FROM #temp	All ASE cursors are sensitive; changes to data fetched by a cursor are reflected in the underlying table. Use a temporary table.
DECLARE cursor_name SCROLL  FETCH { NEXT   PRIOR   FIRST   LAST   ABSOLUTE   RELATIVE }	Remove SCROLL syntax.  Example to replace ABSOLUTE or RELATIVE: Create temporary table with identity column for the row number  SELECT * FROM #restab WHERE rownum = 123	ASE does not support scrolling capability and advanced fetching options.  For applications which require this scrolling and fetching capability, create temporary tables and use an identity column to make row selections, or use a cursor.
DEALLOCATE cursor_name	DEALLOCATE CURSOR cursor_name	Cursor deallocation syntax for ASE includes keyword CURSOR.
GLOBAL cursors	No equivalent	ASE cursors are local in scope.
Global Temporary Tables: CREATE TABLE ##global_table	No equivalent	ASE does not support global temporary tables. A temporary table may be created in tempdb to simulate behavior. The table can be dropped prior to session termination.
uniqueidentifier data type and the corresponding rowguidcol pseudo column	Identity column, a binary column or an integer counter	uniqueidentifier is not available in ASE.
RAISERROR ( (error number   error message) severity, state [ , argument list] ) [ WITH LOG   NOWAIT   SETERROR]	RAISERROR error number [ 'error message'] [ , argument list] [ WITH errordata restricted_select_list]	Adapt the RAISERROR statement: remove WITH clause options which are not used in ASE environment, replace printf()-style error_message with ASE's implicit conversion of @vars corresponding to numbered parameters in the error_message string.
FORMATMESSAGE()	To access error message text, select information from the system tables.	
sp_addmessage msgnum	sp_addmessage msgnum	User messages in ASE are stored in sysusermessages while MSSQL stores them in

msgnum > 50000	msgnum > 20000	sysmessages. ASE message numbers can be > 20,000
<b>Optimizer hints of:</b> INDEX NOLOCK UPDLOCK TABLOCK PAGLOCK TABLOCKX XLOCK HASH MAXDOP	Remove or replace with ASE's hints.	Optimizer hints for tables and queries should be removed to allow the ASE optimizer to choose the optimal plan.
PRINT 'I' + 'am'	PRINT 'I %1!', 'am'  DECLARE @s SELECT @s = 'I + 'am' PRINT @s	PRINT does not support concatenation of strings; use placeholders or a variable instead.
<b>DEFAULT keyword in the VALUES list of an INSERT statement or stored procedure</b>  INSERT..VALUES (DEFAULT)	Instead, use the ISNULL() function or define a default on the column and then do not even specify the column. For stored procedures, you can set a default for an parameters like:  <input_param <="" int="1" td=""> <td>Replace with appropriate syntax.</td> </input_param>	Replace with appropriate syntax.
<b>Derived tables and EXECUTE not available in VALUES list of INSERT</b>	Remove and replace with a temporary table.	You cannot use a derived table (subquery) or the EXECUTE statement in ASE's INSERT.
CAST()	CONVERT()	Converting expressions datatypes.
GETANSINULL()	SELECT COUNT(1) FROM master..sysdatabases WHERE status & 8192 = 8192 AND name = "db1"  This query will return 1 if the option is set and 0 otherwise.	IF the database option "allow nulls by default" is set to true, 1 is returned, otherwise 0 is returned.
VAR() VARP() STDEV() STDEVP() COUNT_BIT()	No equivalent	These aggregate functions not available in ASE.
CURSOR STATUS() APP_NAME() ISDATE() ISNUMERIC() PERMISSIONS() PARSENAME() CURRENT_TIMESTAMP() GETUTCDATE()	No equivalent	These system functions not available in ASE.
STATS_DATE()	Query system statistics tables directly	System function not necessary.
DATABASEPROPERTY() FILEPROPERTY() COLUMNPROPERTY() DATABASEPROPERTYEX() FILE_ID() FILE_NAME() FILEGROUP_ID() FILEGROUP_NAME() FILEGROUPPROPERTY() fn_listextendedproperty() FULLTEXTCATALOGPROPERTY() FULLTEXTSERVICEPROPERTY() INDEXKEY_PROPERTY() INDEXPROPERTY() OBJECTPROPERTY() SERVERPROPERTY() SESSIONPROPERTY() SQL_VARIANT_PROPERTY()	Query system catalog instead.	Meta-data functions not available.

msgnum > 50000	msgnum > 20000	sysmessages. ASE message numbers can be > 20,000
<b>Optimizer hints of:</b> INDEX NOLOCK UPDLOCK TABLOCK PAGLOCK TABLOCKX XLOCK HASH MAXDOP	Remove or replace with ASE's hints.	Optimizer hints for tables and queries should be removed to allow the ASE optimizer to choose the optimal plan.
PRINT 'I' + 'am'	PRINT 'I %!%', 'am'  DECLARE @s SELECT @s = 'I + 'am' PRINT @s	PRINT does not support concatenation of strings; use placeholders or a variable instead.
<b>DEFAULT keyword in the VALUES list of an INSERT statement or stored procedure</b> INSERT..VALUES (DEFAULT)	Instead, use the ISNULL () function or define a default on the column and then do not even specify the column. For stored procedures, you can set a default for an parameters like:  <input/> _param int = 1	Replace with appropriate syntax.
<b>Derived tables and EXECUTE not available in VALUES list of INSERT</b>	Remove and replace with a temporary table.	You cannot use a derived table (subquery) or the EXECUTE statement in ASE's INSERT.
CAST ()	CONVERT ()	Converting expressions datatypes.
GETANSINULL()	SELECT COUNT (1) FROM master..sysdatabases WHERE status & 8192 = 8192 AND name = "dbl"  This query will return 1 if the option is set and 0 otherwise.	IF the database option "allow nulls by default" is set to true, 1 is returned, otherwise 0 is returned.
VAR () VARP () STDEV () STDEVP () COUNT_BIT ()	No equivalent	These aggregate functions not available in ASE.
CURSOR_STATUS () APP_NAME () ISDATE () ISNUMERIC () PERMISSIONS () PARSENAME () CURRENT_TIMESTAMP () GETUTCDATE ()	No equivalent	These system functions not available in ASE.
STATS_DATE ()	Query system statistics tables directly	System function not necessary.
DATABASEPROPERTY() FILEPROPERTY() COLUMNPROPERTY () DATABASEPROPERTYEX () FILE_ID () FILE_NAME () FILEGROUP_ID () FILEGROUP_NAME () FILEGROUPPROPERTY () fn_listextendedproperty () FULLTEXTCATALOGPROPERTY () FULLTEXTSERVICEPROPERTY () INDEXKEY_PROPERTY () INDEXPROPERTY () OBJECTPROPERTY () SERVERPROPERTY () SESSIONPROPERTY () SQL_VARIANT_PROPERTY ()	Query system catalog instead.	Meta-data functions not available.

TYPEPROPERTY ()		
UNICODE ()	ASCII ()	String functions not available in ASE.
QUOTENAME ()	No equivalent	
REPLACE ()	STR_REPLACE()	
NCHAR ()	TO_UNICHAR()	
LIKE operator involves trailing space in search argument	LIKE "ABC[ ][ ] "	ASE LIKE ignores trailing space; use "[ ][ ]" to represent trailing spaces.
BEGIN DISTRIBUTED TRANSACTION	BEGIN TRANSACTION	CIS will automatically coordinate multi-server transactions.
BEGIN TRANSACTION...WITH MARK	BEGIN TRANSACTION tran_name	Recovery using named points of work is not supported in ASE. Remove the WITH MARK syntax and use ASE point-in-time recovery.
SET IMPLICIT TRANSACTION ON	SET chained ON	
IS_MEMBER () IS_SRVROLEMEMBER ()	MUT_EXCL_ROLES () PROC_ROLES () ROLE_CONTAIN () ROLE_ID () ROLE_NAME () SHOW_ROLE ()	Some ASE role functions not available in MSSQL.
Bracketed identifiers: SELECT * FROM [ TABLE]	SET quoted identifier ON and delimit identifiers using double quotes (""). SELECT * FROM "TABLE"	ASE does not support the use of square brackets for delimiting identifiers.
Identifiers > 30 characters	Identifiers <= 30 characters	Observe identifier length limits.
JOINS with > 64 tables	JOINS <= 64 tables	Observe the limit to the number of tables in a join.
Tables with > 254 columns	Tables <= 254 columns (allpages locking) Tables <= 1024 columns (datapages and datarows locking)	Observe limits to the number of columns in a table based on the desired locking scheme.
FOREIGN KEY... ON UPDATE   DELETE { CASCADE   NO ACTION}	Use triggers to cascade changes to other tables.	Cascading referential integrity does not exist in ASE.
ALTER TABLE...WITH NOCHECK	No equivalent	NOCHECK is default ASE behavior.
ALTER TABLE table name ADD DROP COLUMN column_name...	ALTER TABLE table name ADD DROP column_name...	Remove the keyword COLUMN when using ALTER TABLE to add or drop a column.
ALTER TABLE table name ADD COLUMN column name... WITH VALUES	ALTER TABLE table name ADD column_name... DEFAULT	To add values to a new column for existing rows, use DEFAULT keyword of ALTER TABLE.
ALTER TABLE table name ALTER COLUMN column_name...	ALTER TABLE table name MODIFY column_name...	To change a column's datatype, default or nullability in ASE, use MODIFY keyword of ALTER TABLE.
ALTER TABLE ADD DROP ROWGUIDCOL	No equivalent	Since ASE does not support the uniqueidentifier datatype, these references will have to be removed and substituted with a different identifying attribute.
CREATE TRIGGER...INSTEAD OF   AFTER	By default, ASE triggers are AFTER triggers: the trigger fires after the firing event.	ASE allows one trigger for each insert, update, and delete on a table. To achieve the effect of multiple triggers, place each of the trigger bodies in stored procedures and call these stored procedures from a single table trigger.
sp_settriggerorder	No equivalent	

CREATE PROCEDURE/TRIGGER/VIEW... WITH ENCRYPTION	sp_hidetext	To conceal query batch for the database object in the syscomments table, use the sp_hidetext procedure.
CREATE PROCEDURE... FOR REPLICATION	Use ASE Replicator or Sybase Replication Server for replication	Replication in ASE is handled through ASE Replicator or Sybase Replication Server for replication. Consult the product manuals for specific commands and usage.
CREATE TABLE ...col_name AS computed_col_expression	Add a column to the table and maintain through a trigger or the application.	Computed columns are not supported in ASE.
CREATE VIEW ... federated table ... UNION ALL federated_table	No equivalent	Partitioned views as used by member tables in Federated Servers are not available.
CREATE VIEW... WITH SCHEMABINDING	No equivalent	Indexed Views are not available in ASE.
IDENTITY(seed, increment) datatype	datatype IDENTITY	For ASE, remove the seed and increment when defining identity columns.
IDENT CURRENT(table_name) IDENT SEED() IDENT INCR()	NEXT_IDENTITY(table_name)	For ASE, the last identity value inserted for any table is retrieved with @@IDENTITY. To find the next available identity value that will be inserted for a table, use NEXT_IDENTITY().
@@IDENTITY	@@IDENTITY	Last identity value assigned in the session for any table in all scopes.
SCOPE_IDENTITY()	No equivalent	Last identity for any table in current scope.
IDENTITYCOL	SYB_IDENTITY	To refer to an identity column in a table.

### ANSI System Value Functions

MSSQL	ASE Equivalent	Comment
CURRENT_TIMESTAMP CURRENT_USER SESSION_USER SYSTEM_USER USER	USER_NAME()  SUSER_NAME() USER	The MSSQL 'CREATE   ALTER TABLE' syntax allows five ANSI functions to return a system supplied value.

### MSSQL-Specific Functions and ASE Equivalents:

MSSQL	ASE Equivalent	Comment
SOME()	IN()	Subquery function to compare a single value with a list of values.
CHECKSUM() CHECKSUM_AGG()	No equivalent	Function returns checksum over values in a row or expression.
COLLATE()	SORTKEY()	Retrieve an alternative collation value for sorting.
COLLATIONPROPERTY()	No equivalent	Obtains properties of a collation sequence
COUNT_BIG()	COUNT()	Similar to COUNT(), but returns bigint data type.
GETUTCDATE()	No equivalent	Retrieves Greenwich Mean Time.
OPENQUERY()	No equivalent	Passes query to a linked server. Create and use proxy tables instead.
OPENROWSET()		Creates a rowset from a remote OLE DB data source. Create and use proxy tables instead.
ROWCOUNT_BIG()	ROWCOUNT()	Similar to ROWCOUNT(), but with bigint return type.
SUSER_SID()	SUSER_ID()	Returns the login identifier.
SUSER_SNAME()	SUSER_NAME()	Returns the return login name.

## Additional ASE functions not available in MSSQL

ASE Function	Comment
COL_LENGTH ()	The defined length of column.
COL_NAME ()	The name of column for specified table and column id.
COMPARE ()	Compares two character expressions using the specified collation and sorting rules.
DATA_PGS ()	The number of pages used by specified table or index.
LICENSE_ENABLED ()	A positive or negative value indicating whether specified licensed components (e.g., ASE_SERVER, ASE_HA, ASE_DTM, etc.) are enabled on the server.
LOCKSCHEME ()	The locking scheme(allpages, datapages, datarows) as a string.
PTN_DATA_PAGES ()	Number of data pages used by a partition.
SORTKEY ()	Provides values for sorting using different character collations.
SYB_SENDMSG ()	Sends a text message via UDP to the specified IP address and port.
UHIGHSURR () UOWSURR ()	Detects high and low location of surrogate pair in a Unicode value.
USCALAR ()	Returns scalar value for first Unicode character in an expression.

## Session Option Setting Differences

When migrating T-SQL to ASE from MSSQL the following points must be addressed:

MSSQL SET Option	ASE Equivalent	Comment
ANSI_NULL_DFLT_ON ANSI_NULL_DFLT_OFF	No equivalent	Not available as a session level setting. Use the ASE database option "allow nulls by default".
ANSI_DEFAULTS ANSI_PADDING	No equivalent	Trailing spaces on variable length columns are always truncated, while fixed-length values are always padded with spaces.
ANSI_NULLS	ansinull	
CURSOR_CLOSE_ON_COMMIT	close on endtran	
DEADLOCK_PRIORITY	No equivalent	
FIPS_FLAGGER	fipsflagger	The ASE option does not take a FIPS level parameter.
IMPLICIT_TRANSACTIONS	chained	
LOCK_TIMEOUT	lock wait	
NUMERIC_ROUNDABORT	arithabort numeric_truncation	
QUERY_GOVERNOR_COST_LIMITS	Impose resource limits on users instead of through a session.	Use administrative resource limits on users via Resource Governor feature.
REMOTE_PROC_TRANSACTION	transactional_rpc	
SHOWPLAN_TEXT	showplan	There is no equivalent to MSSQL's SHOWPLAN_ALL which produces a result set rather than a human readable plan.
XACT_ABORT	Not available	Use T-SQL transactional control statements to define transaction boundaries.
STATISTICS_PROFILE	No equivalent	Use the ASE <i>optdiag</i> utility to show statistics on objects and data.

The ANSI\_WARNINGS, CONCAT\_NULL\_YIELDS\_NULL, ARITHABORT, ARITHIGNORE, NUMERIC\_ROUNDABORT session options are described as database options in Appendix B, Database Option Differences.

### Additional ASE-Specific T-SQL Session Options

ASE SET Option	Comment
ansi_permissions	Forces checking for ANSI-92 permissions requirements for delete and update statements.
proxy	Assume permissions of another login name.
session authorization	
role	Toggle role on or off for a session.
prefetch	Enables/disables large I/Os to the data cache.
process_limit_action	Warns when insufficient number of worker processes available to process a query.
cursor_rows	Sets number of rows returned by each cursor FETCH statement.
table count	Sets number of tables considered during join optimization.
char_convert	Enables/disables character set conversion between the ASE and client.
statistics subquerycache	Displays hits, misses, and rows in subquery cache for each subquery.
string_truncation	Exception raised when data truncated through insert/update.

### Global Variable Differences

When migrating T-SQL to ASE from MSSQL, the following global variable differences must be addressed:

MSSQL Global Variable	ASE Equivalent	Comment		
@@CURSOR_ROWS	No equivalent. Use: <code>SELECT COUNT(*) FROM ...'</code> to find out how many rows would be returned by cursor query.	ASE @@ROWCOUNT will give you the number of rows fetched up to the last fetch statement.		
@@fetch_status	@@sqlstatus	Cursor FETCH status values are slightly different between servers.		
Status	Description		Status	Description
0	Successful FETCH.		0	Successful FETCH.
-1	Cursor reached end of dataset or row does not exist.		2	Cursor reached end of data set.
-2	Row deleted or key updated after cursor opened (scrollable option).	1	FETCH resulted in error.	
@@DBTS	No equivalent	The ASE server automatically inserts timestamp column values and does not provide a way to query any current value for this type.		
@@MAX_PRECISION	@@MAX_PRECISION	Note: ASE's max precision is 38 digits.		
@@REMSERVER	No equivalent			
@@SERVICENAME	@@SERVERNAME	The name of the Windows service for an ASE server is always "Sybase SQLServer _NAME" and the internal service name is always "SYBSQL_NAME" where "NAME" is the name of the ASE server.		
@@PROCID	@@PROCID	If a stored procedure causes a trigger to fire, the value of @@PROCID in the trigger is the ID of the stored procedure (so allowing the trigger to find out what caused it to fire). The MSSQL version of @@PROCID will be set to the ID of the trigger in this case.		

## Reserved Keyword Differences

Reserved keywords in the T-SQL language should not be used as object identifiers (e.g. table or column names). Any database objects created with names matching any of these keywords will need to be renamed during migration (for example using an application prefix). Optionally, ASE provides a session option, `quoted_identifier`, which may be turned on to allow access to object names with reserved words.

<code>arith_overflow</code>	<code>identity_gap</code>	<code>once</code>	<code>replace</code>
<code>char_convert</code>	<code>identity_start</code>	<code>online</code>	<code>reservepagegap</code>
<code>confirm</code>	<code>inout</code>	<code>out</code>	<code>returns</code>
<code>controlrow</code>	<code>install</code>	<code>partition</code>	<code>role</code>
<code>deterministic</code>	<code>jar</code>	<code>perm</code>	<code>shared</code>
<code>disk</code>	<code>lock</code>	<code>permanent</code>	<code>stringsize</code>
<code>endtran</code>	<code>max_rows_per_page</code>	<code>processexit</code>	<code>stripe</code>
<code>errordata</code>	<code>mirror</code>	<code>proxy</code>	<code>syb_identity</code>
<code>errorexit</code>	<code>mirrorexit</code>	<code>proxy_table</code>	<code>syb_restree</code>
<code>exclusive</code>	<code>modify</code>	<code>quiesce</code>	<code>syb_terminate</code>
<code>exp_row_size</code>	<code>new</code>	<code>readpast</code>	<code>temp</code>
<code>external</code>	<code>noholdlock</code>	<code>remove</code>	<code>unpartition</code>
<code>func</code>	<code>numeric_truncation</code>	<code>reorg</code>	<code>user_option</code>

MSSQL's T-SQL language reserved keyword list includes the following keywords that are not in ASE's list.

<code>absolute</code>	<code>describe</code>	<code>lower</code>	<code>session</code>
<code>action</code>	<code>descriptor</code>	<code>match</code>	<code>session_user</code>
<code>ada</code>	<code>diagnostics</code>	<code>minute</code>	<code>size</code>
<code>allocate</code>	<code>disconnect</code>	<code>module</code>	<code>smallint</code>
<code>are</code>	<code>distributed</code>	<code>month</code>	<code>space</code>
<code>assertion</code>	<code>domain</code>	<code>names</code>	<code>sql</code>
<code>backup</code>	<code>end-exec</code>	<code>natural</code>	<code>sqlca</code>
<code>bit</code>	<code>exception</code>	<code>nchar</code>	<code>sqlcode</code>
<code>bit_length</code>	<code>extract</code>	<code>next</code>	<code>sqlerror</code>
<code>both</code>	<code>false</code>	<code>no</code>	<code>sqlstate</code>
<code>cascaded</code>	<code>file</code>	<code>nocheck</code>	<code>sqlwarning</code>
<code>cast</code>	<code>first</code>	<code>none</code>	<code>substring</code>
<code>catalog</code>	<code>float</code>	<code>numeric</code>	<code>system_user</code>
<code>char</code>	<code>fortran</code>	<code>octet_length</code>	<code>then</code>
<code>char_length</code>	<code>found</code>	<code>opendatasource</code>	<code>time</code>
<code>character</code>	<code>freetext</code>	<code>openquery</code>	<code>timestamp</code>
<code>character_length</code>	<code>freetexttable</code>	<code>openrowset</code>	<code>timezone_hour</code>
<code>collate</code>	<code>full</code>	<code>openxml</code>	<code>timezone_minute</code>
<code>collation</code>	<code>get</code>	<code>outer</code>	<code>top</code>
<code>column</code>	<code>go</code>	<code>overlaps</code>	<code>trailing</code>
<code>connection</code>	<code>hour</code>	<code>pad</code>	<code>translate</code>
<code>constraints</code>	<code>identitycol</code>	<code>partial</code>	<code>translation</code>
<code>contains</code>	<code>immediate</code>	<code>pascal</code>	<code>trim</code>
<code>containstable</code>	<code>include</code>	<code>percent</code>	<code>true</code>
<code>corresponding</code>	<code>indicator</code>	<code>position</code>	<code>unknown</code>
<code>cross</code>	<code>initially</code>	<code>preserve</code>	<code>updatetext</code>
<code>current_date</code>	<code>inner</code>	<code>prior</code>	<code>upper</code>
<code>current_time</code>	<code>input</code>	<code>real</code>	<code>usage</code>
<code>current_timestamp</code>	<code>insensitive</code>	<code>global</code>	<code>value</code>
<code>current_user</code>	<code>int</code>	<code>relative</code>	<code>varchar</code>
<code>date</code>	<code>integer</code>	<code>restore</code>	<code>whenever</code>
<code>day</code>	<code>interval</code>	<code>restrict</code>	<code>write</code>
<code>dec</code>	<code>language</code>	<code>right</code>	<code>year</code>
<code>decimal</code>	<code>last</code>	<code>rowguidcol</code>	<code>zone</code>
<code>deferrable</code>	<code>leading</code>	<code>scroll</code>	
<code>deferred</code>	<code>left</code>	<code>second</code>	
<code>deny</code>	<code>local</code>	<code>section</code>	

## ASE OLE DB Interfaces and Types Support

ASE provides an OLE DB provider in support of the OLE DB and ADO specifications. For more information on the ASE OLE DB provider, please see the Sybase ASE OLE DB Provider Help file.

### ASE OLE DB Objects and Supported Interfaces

OLE DB Object	CoType	Interface
Command	TCommand	IAccessor IColumnsInfo IColumnsRowset ICommand ICommandPrepare ICommandProperties ICommandText ICommandWithParameters IConvertType ISupportErrorInfo
DataSource	TDataSource	IDBCreateSession IDBInitialize IDBProperties IDBInfo IPersist IPersistFile ISupportErrorInfo
Enumerator	TEnumerator	IParseDisplayName ISourcesRowset ISupportErrorInfo
Multiple Results	TMultipleResults	IMultipleResults ISupportErrorInfo
Rowset	TRowset	IAccessor IColumnsInfo IColumnsRowset IConnectionPointContainer IConvertType IRowset IRowsetChange IRowsetIdentity IRowsetInfo IRowsetLocate IRowsetRefresh IRowsetScroll IRowsetUpdate ISupportErrorInfo
Session	TSession	IDBSchemaRowset IDBCreateCommand IGetDataSource IOpenRowset ISessionProperties ISupportErrorInfo ITransaction ITransactionJoin ITransactionLocal
Transaction	TTransaction	ITransaction ISupportErrorInfo
Transaction Options	TTransactionOptions	ITransactionOptions ISupportErrorInfo
Error Lookup		IErrorLookup

## ASE to OLE DB Data Type Mapping

The ASE OLE DB driver provides support for the following OLE DB Data Types.

ASE Data Type	OLE DB Data Type
binary	DBTYPE_BYTES
bit	DBTYPE_BOOL
char	DBTYPE_STR
datetime	DBTYPE_DBTIMESTAMP
decimal	DBTYPE_NUMERIC
float	DBTYPE_R8
image	DBTYPE_BYTES
int	DBTYPE_I4
money	DBTYPE_CY
numeric	DBTYPE_NUMERIC
real	DBTYPE_R4
smalldatetime	DBTYPE_DBTIMESTAMP
smallint	DBTYPE_I2
smallmoney	DBTYPE_CY
sysname	DBTYPE_STR
text	DBTYPE_STR
tinyint	DBTYPE_UI1
varbinary	DBTYPE_BYTES
varchar	DBTYPE_STR
No equivalent	DBTYPE_VARNUMERIC
No equivalent	DBTYPE_FILETIME

## ASE OLE DB Connection Properties

The following properties can be specified within the connection string for an ASE ODBC connection.

Attribute Name	Description
ApplicationName	Name used by ASE to identify your application.
ArraySize	Number of rows retrieved from server for each fetch. Default is 50 rows.
Charset	Character set corresponding to a subdirectory in \$SYBASE/charsets.
Database	Name of the database to which you want to connect.
DefaultLengthForLongData	Maximum length of TEXT or IMAGE data fetched in 1024-byte multiples. Default is 1024.
EnableQuotedIdentifiers	Enable quoted identifiers.
ExtendedErrorInfo	Amount of error information to be returned.
InitializationString	Run these ASE SET commands at connect time.
Language	National language from subdirectory \$SYBASE/locales. Default is English.
LogonID	Default logon ID to connect to ASE, case-sensitive.
NetworkProtocol	ASE Network Library to use for application communication: Winsock or Named Pipes. Default is Winsock.
OptimizePrepare	Determines whether stored procedures are created on server for every call to ICommand::Prepare.
PacketSize	Network packet size for data transfer.
PrintStatementBehavior	Whether print statements sent back to user as separate result sets.
RaiseErrorPositionBehavior	Specifies when the error is returned and where the cursor is positioned when raiserror is encountered.
SelectMethod	Determines whether database cursors are used for Select statements.
SybaseServerName	The name of the server.
ServerPortAddress	Address of the ASE server.
StoredProcRowCount	Row count for SQL statements in the stored procedure.
SybaseLDAPURL	The LDAP URL for the search operation used to retrieve information from an LDAP directory.
SybaseServerName	The name of the ASE server in the LDAP entry.
TightlyCoupledDistributedTransactions	Determines whether the driver uses tightly coupled distributed transactions when connected to a Sybase version 12 or higher server. Default is 1, multiple connections are treated as one transaction.
UseLDAPHAFailoverServer	Determines whether the driver uses the High Availability (HA) Failover server specified in the LDAP directory. Default is 0, or false.
UseSybaseLDAPSchema	Determines whether the driver uses an LDAP directory entry to retrieve connection information. Default is 0, or false.
UseSSL	Determines whether the driver uses a secure sockets layer (SSL) connection. Default is 0, or false.
WorkstationID	Workstation ID used by the client.

## ASE ODBC Function Compliance and Types Support

Sybase provides ASE ODBC drivers that are level 1, 2, and 3-compliant. For more information on the ASE OLE DB provider, please see the Sybase ASE ODBC Driver Help files.

### Level 2-Compliant Functions

SQLBrowseConnect SQLDataSources SQLExtendedFetch (forward scrolling only)	SQLMoreResults SQLNativeSql SQLNumParams	SQLParamOptions SQLSetScrollOptions
---	--	--

### Level 3-Compliant Functions

SQLAllocHandle SQLBindCol SQLBindParameter SQLBrowseConnect SQLBulkOperations SQLCancel SQLCloseCursor SQLColAttribute SQLColumns SQLConnect SQLCopyDesc SQLDataSources SQLDescribeCol SQLDisconnect SQLDriverConnect SQLDrivers SQLEndTran SQLError SQLExecDirect	SQLExecute SQLExtendedFetch SQLFetch SQLFetchScroll (forward scrolling only) SQLFreeHandle SQLFreeStmt SQLGetConnectAttr SQLGetCursorName SQLGetData SQLGetDescField SQLGetDescRec SQLGetDiagField SQLGetDiagRec SQLGetEnvAttr SQLGetFunctions SQLGetInfo SQLGetStmtAttr SQLGetTypeInfo SQLMoreResults	SQLNativeSql SQLNumParams SQLNumResultCols SQLParamData SQLPrepare SQLPutData SQLRowCount SQLSetConnectAttr SQLSetCursorName SQLSetDescField SQLSetDescRec SQLSetEnvAttr SQLSetStmtAttr SQLSpecialColumns SQLStatistics SQLTables SQLTransact
--	--	---

## ASE to ODBC Data Type Mapping

The ASE ODBC driver provides support for the following ODBC Data Types.

ASE Data Type	ODBC Data Type
binary	SQL_BINARY
bit	SQL_BIT
char	SQL_CHAR
datetime	SQL_TYPE_TIMESTAMP
decimal	SQL_DECIMAL
float	SQL_FLOAT
image	SQL_LONGVARIABLE
int	SQL_INTEGER
money	SQL_DECIMAL
numeric	SQL_NUMERIC
real	SQL_REAL
smalldatetime	SQL_TYPE_TIMESTAMP
smallint	SQL_SMALLINT
smallmoney	SQL_DECIMAL
sysname	SQL_VARCHAR
text	SQL_LONGVARIABLE
timestamp	SQL_VARBINARY
tinyint	SQL_TINYINT
varbinary	SQL_VARBINARY
varchar	SQL_VARCHAR

## ASE ODBC Connection Properties

The following properties can be specified within the connection string for an ASE ODBC connection.

Attribute Name (Short Name)	Description
ApplicationName (APP)	Name used by ASE to identify your application.
ApplicationUsingThreads (AUT)	Ensures that the driver works with multi-threaded applications. Default is 1 or thread-safe.
ArraySize (AS)	Number of rows retrieved from server for each fetch. Default is 50 rows.
Charset (CS)	Character set corresponding to a subdirectory in \$\$SYBASE/charsets.
CursorCacheSize (CCS)	The number of connections that the connection cache can hold. Default is 1 cursor.
Database (DB)	Name of the database to which you want to connect.
DataSourceName (DSN)	A string that identifies a single connection to an ASE database
DefaultLongDataBufLen (DLDBL)	Maximum length of TEXT or IMAGE data fetched in 1024-byte multiples. Default is 1024.
DistributedTransactionModel (DTM)	DistributedTransactionModel={XA Protocol   Native OLE}, default is XA Protocol
EnableDescribeParam (EDP)	Determines whether the ODBC API function SQLDescribeParam is enabled. Default is 0, or disabled.
EnableQuotedIdentifiers (EQI)	Enable quoted identifiers, default is 0, or disabled.
FailoverNetworkAddress (FNA)	Specifies the address of the High Availability (HA) Failover server
InitializationString (IS)	Run these ASE SET commands at connect time.
Language (LANG)	National language from subdirectory \$\$SYBASE/locales, default is English.
LogonID (UID)	Default logon ID to connect to ASE, case-sensitive.
NetworkAddress (NA)	The network address depends on which network protocol is chosen under NetworkLibraryName and on the ASE server
NetworkLibraryName (NLN)	ASE Network Library to use for application communication: Winsock or Named Pipes. Default is Winsock.
OptimizePrepare (OP)	Determines whether stored procedures are created on server for every call to SQLPrepare. Default is 1, the drivers creates procedures only if the statements contains parameters.
Password (PWD)	A case-sensitive password.
PacketSize (PS)	Network packet size for data transfer. Default is 0, the ASE server default.
RaiseErrorPositionBehavior (REPB)	Specifies when the error is returned and where the cursor is positioned when raiserror is encountered. Default is 0, raiserror is handled separately.
SelectMethod (SM)	Determines whether database cursors are used for Select statements. Default is 0, cursors are used.
SybaseLDAPURL (SLURL)	The LDAP URL for the search operation used to retrieve information from an LDAP directory.
SybaseServerName (SSN)	The name of the ASE server in the LDAP entry.
TightlyCoupledDistributedTransactions (TCDT)	Determines whether the driver uses tightly coupled distributed transactions when connected to a Sybase version 12 or higher server. Default is 1, multiple connections are treated as one transaction.
TrustedRootsFileName (TRFN)	Specifies the location and name of the trusted roots file used for the SSL connection.
UseLDAPHAFailoverServer (ULDAPHA)	Determines whether the driver uses the High Availability (HA) Failover server specified in the LDAP directory. Default is 0, or false.
UseSybaseLDAPSchema (USLDAP)	Determines whether the driver uses an LDAP directory entry to retrieve connection information. Default is 0, or false.
UseSSL (USSL)	Determines whether the driver uses a secure sockets layer (SSL) connection. Default is 0, or false.
WorkstationID (WKID)	Workstation ID used by the client.

## Appendix B Portability List: Administrative Features

In this appendix, several lists are presented describing administrative feature differences that are likely to be encountered when migrating applications from Microsoft SQL Server to Sybase Adaptive Server Enterprise.

### Server Configuration Option Differences

MSSQL Server Option	ASE Equivalent	Comment
affinity mask	DBCC TUNE (cpuaffinity, start_cpu, on)	DBCC TUNE allows the DBA to decide which starting CPU has affinity to Engine 0.
cost threshold for parallelism	No equivalent	Handled automatically by ASE's query processor.
cursor threshold	No equivalent	Cursor result sets in ASE are materialized before they are fetched.
index create memory	number of sort buffers	Index creation for ASE can be done serially or in parallel, and performance can be affected by raising the "number of sort buffers".
lightweight pooling	No equivalent	ASE will use fibers for user connections automatically.
max degree or parallelism	max parallel degree	ASE also provides the setting "max scan parallel degree" for further DBA control.
max text repl size	No equivalent	Use <i>ASE Replicator</i> or <i>Sybase Replication Server</i> for replication.
max worker threads	number of worker processes	
media retention	tape retention in days	
min memory per query	No equivalent	While memory cannot be specified at the query-level, ASE provides the server configuration option "stack size" to handle memory issues.
open objects	number of open objects	
priority boost	No equivalent	
query governor cost limit	allow resource limits	The ASE resource governor allow fine-tuning of resource limits for users. See "Limiting Access to Server Resources" in the <i>ASE System Administration Guide</i> for more details.
remote login timeout remote query timeout	No equivalent	ASE uses standard timeout values.
remote proc trans	SET transactional_rpc	When CIS is enabled, use SET transactional_rpc and SET cis_rpc_handling to use transactional RPCs. The ASE global variable @@cis_rpc_handling can be checked to see if cis_rpc_handling is set. Returns 0 if cis_rpc_handling is off. Returns 1 if cis_rpc_handling is on. The ASE global variable @@transactional_rpc can be checked to see if transactional_rpc is set. Returns 0 if RPCs to remote servers are transactional. Returns 1 if RPCs to remote servers are not transactional.
min server memory	No equivalent	
max server memory	max memory	The maximum amount of physical memory which can be allocated.
set working set size (default, 0)	dynamic allocation on demand	Allocate memory as needed.
set working set size (=1)	allocate max shared memory	Reserve max memory at server start.
show advanced option	sp_displaylevel	To control which configuration parameters are displayed using sp_configure.
user options	No equivalent	

## Database Configuration Differences

MSSQL Database Option	ASE Equivalent	Comment
ANSI_NULL_DEFAULT	allow nulls by default	Create table column declarations do not allow nulls by default.
AUTO_CLOSE AUTO_SHRINK	No equivalent	
AUTO_CREATE_STATISTICS AUTO_UPDATE_STATISTICS	No equivalent	Use UPDATE STATISTICS in conjunction with a scheduling utility to maintain statistics.
ANSI_NULLS	No equivalent	Not necessary in ASE because of its transparent handling of UNICODE as a character set.
ANSI_WARNINGS	SET arithabort ON SET arithignore ON SET ansinull ON	Can be emulated using ASE's session-level settings.
CONCAT_NULL_YIELDS_NULL	No equivalent	By default, concatenation of NULL to a string will yield the string, not NULL.
CURSOR_CLOSE_ON_COMMIT	SET close on endtran ON	Can be emulated using an ASE session-level setting.
CURSOR_DEFAULT LOCAL   GLOBAL	No equivalent	By default, all cursors in ASE are local.
ONLINE   OFFLINE	ONLINE DATABASE	Use ASE database option "dbo use only" to emulate OFFLINE.
TORN_PAGE_DETECTION	No equivalent	Use of disks with battery-backed caches alleviates incomplete writes of data in the event of a power failure.
RECOVERY FULL  BULK_LOGGED  SIMPLE	Default behavior  select into/ bulkcopy/plsort  trunc log on chkpt	By default, ASE fully logs all operations. For minimal logging of the WRITETEXT utility, the SELECT INTO command, and fast BCP ( <i>bcp</i> into a table w/o a unique index), the "select into option" is set. To simulate SIMPLE recovery model, set the database option trunc log on chkpt.
RECURSIVE_TRIGGERS	SET self recursion ON	The allow nested triggers database option must be enabled for ASE just as the nested triggers server option had to be enabled for MSSQL to allow recursion in triggers.
READ_ONLY  READ_WRITE	read only  No equivalent	Read, write modes for a database. By default ASE databases allow both read and write.
WITH { ROLLBACK AFTER   ROLLBACK IMMEDIATE   NOWAIT }	No equivalent	

## Additional ASE-specific Database Options

ASE Database Option	Comment
abort tran on log full	Normally, a transaction is suspended when free space is at a premium. When this option is set, abort the transaction.
no free space acctg	For non-log segments, free space accounting is disabled, resulting in quicker recovery time.
auto identity	For each table that is created with <code>CREATE TABLE</code> without a primary key, unique constraint, or identity column, a hidden identity column is created for that table, referred to as <code>SYB_IDENTITY</code> .
identity in nonunique index	Makes a non-unique index unique for cursors and isolation level 0 reads.
no chkpt on recovery	Used for keeping a standby copy of database or for bypassing recovery during troubleshooting.

## DBA Command Differences

The more important differences in DBA commands or system procedures between the products are:

MSSQL DBA Command or Procedure	ASE Equivalent	Comment
sp_configure RECONFIGURE [ WITH OVERRIDE]	sp_configure	ASE does not require RECONFIGURE to activate parameters.
sp_serveroption	sp_serveroption	MSSQL uses parameter to set options for replication, collation compatibility, and connection and query timeouts.  ASE uses parameter to set inter-server network related options (for example whether inter-server password exchanges should be encrypted or not).
ALTER DATABASE sp_dboption	sp_dboption	Slightly different use between products.
sp_tableoption	sp_cacheconfig sp_bindcache	To pin an object in memory, create a cache and bind the object.
table lock on bulk load	No equivalent	ASE does not use row or table locks during the bulk load process, except briefly on the last row or page.
sp_indexoption	ALTER TABLE...LOCK allpages  datapages  datarows	ASE's default locking scheme is allpages (page-locking for data and indexes).
sp_setapprole sp_addrole sp_srvrolemember	CREATE ROLE / DROP ROLE GRANT ROLE / REVOKE ROLE	Security Command differences for roles.
xp_grantlogin xp_loginconfig xp_logininfo xp_revokelogin	sp_grantlogin sp_loginconfig sp_logininfo sp_revokelogin	Security Command differences for integrated Windows authentication.
sp_addarticle sp_dropdistpublisher sp_helpreplicationdboption sp_addsubscription	Use ASE Replicator or Sybase Replication Server.	Replication system stored procedures not in ASE. Please see <i>ASE Replicator User's Guide</i> for native ASE replication capabilities or the Replication Server manuals set for large-scale replication.
sp_msx_enlist sp_post_msx_operation	No equivalent	"target server" related system stored procedures.
sp_addlinkedserver	sp_addserver sp_addexternlogin	Linked server related stored procedures. Use ASE CIS

	create proxy_table	related stored procedures and commands.
sp_add_alert sp_help_jobstep sp_apply_job_to_targets	No equivalent	Agent/alert/job stored procedures not available. Use operating system scheduler.
sp_procoption	No equivalent	Start-up processing related system stored procedures. Use operating system to script events run during start-up.
sp_create_removable sp_attach_db sp_attach_single_file_db sp_detach_db sp_db_remove sp_certify_removeable	No equivalent	Removable and detachable database related system stored procedures.
sp_makewebtask sp_runwebtask sp_dropwebtask	No equivalent	Web task related system stored procedures (Jobs done by SQL Server Agent).
sp_OACreate sp_OAGetProperty sp_OASetProperty sp_OAMethod sp_OADestroy	No equivalent	OLE automation object stored procedures.
sp_fulltext_table sp_fulltext_service sp_fulltext_database sp_fulltext_column sp_fulltext_catalog	sp_create_text_index sp_text_online sp_refresh_text_index sp_drop_text_index sp_text_configure sp_text_dump_database sp_text_load_index sp_optimize_text_index	Substitute with ASE full-text stored procedures. Full-Text feature conversion information and stored procedures can be found in the <i>Full-Text Search Specialty Data Store User's Guide</i> .
fn_trace_geteventinfo fn_trace_getfilterinfo fn_trace_getinfo fn_trace_gettable fn_get_sql sp_trace_create sp_trace_generateevent sp_trace_setevent sp_trace_setfilter sp_trace_setstatus	sp_sysmon  Monitor Server (with Sybase Central's graphical monitor applets)  Windows NT Performance Monitor or System Monitor	It is not necessary to use complex system stored procedures for monitoring of ASE.
sp_change_users_login	No equivalent	Could be simulated in ASE using a SQL batch.
sp_helplogins	sp_displaylogin	Not available, but could be written by DBA or installed from a public stored procedure library.
sp_setnetname	No equivalent	Server network names are maintained in the interfaces file.
sp_createstats sp_autostats  CREATE STATISTICS statistics name ON table/view name (column_list) [ WITH [ FULLSCAN   SAMPLE number { PERCENT ROWS } [ , ] ] [ NORECOMPUTE ] ]  DROP STATISTICS	UPDATE/DELETE STATISTICS commands  UPDATE STATISTICS table_name [ [ index_name ]   [ ( column_list ) ] ] [ using step values ] [ with [ consumers = consumers ] [ , ] [ SAMPLING = X PERCENT ] ]	Statistics related commands. ASE does not have automatic update of statistics provided through sp_autostats. Note that both servers allow full or sampled scans for gathering statistics information. Refer to the ASE <i>Reference Manual</i> for additional syntax.

RESTORE HEADERONLY	LOAD...{WITH HEADERONLY   LISTONLY}	
RESTORE LOG...WITH STOPATMARK STOPBEFOREMARK = 'mark_name'	LOAD TRAN...WITH UNTIL TIME = "datetime_string"	ASE provides point-in-time recovery, only.
CREATE/ALTER DATABASE	DISK INIT CREATE/ALTER DATABASE	ASE initializes devices then adds them to a database.
Trace Flags (e.g., 302, 310, 3604 and 3605, etc)	302, 310, 3604 and 3605 are officially supported by Sybase	Contact Sybase Customer Service and Support for details of other trace flags similar to those supported by MSSQL.
xp_msver xp_sprintf xp_sscanf xp_gettable_dblib xp_gettable_odbc xp_hello xp_sqlmaint xp_srv_paraminfo_sample	No equivalent.	sscanf extended stored procedures. Write custom ASE stored procedures to handle this functionality.

The MSSQL BACKUP syntax is slightly different from the ASE DUMP syntax. Below is the MSSQL BACKUP command with the non-portable keywords in **bold**.

```

BACKUP DATABASE { database_name | @database_name_var }
TO < backup_device > [ ,...n ]
[ WITH
[ BLOCKSIZE = { blocksize | @blocksize_variable } ]
[ [ , ] DESCRIPTION = { 'text' | @text_variable } ]
[ [ , ] DIFFERENTIAL ]
[ [ , ] EXPIREDATE = { date | @date_var }
| RETAINDAYS = { days | @days_var } ]
[ [ , ] PASSWORD = { password | @password_variable } ]
[ [ , ] FORMAT | NOFORMAT ]
[ [ , ] { INIT | NOINIT } ]
[ [ , ] MEDIADESCRIPTION = { 'text' | @text_variable } ]
[ [ , ] MEDIANAME = { media_name | @media_name_variable } ]
[ [ , ] MEDIAPASSWORD = { mediapassword | @mediapassword_variable } ]
[ [ , ] NAME = { backup_set_name | @backup_set_name_var } ]
[ [ , ] { NOSKIP | SKIP } ]
[ [ , ] { NOREWIND | REWIND } ]
[ [ , ] { NOUNLOAD | UNLOAD } ]
[ [ , ] RESTART ]
[ [ , ] STATS [ = percentage ] ]
]

< backup_device > ::=
{
  logical_backup_device_name | @logical_backup_device_name_var }
|
{ DISK | TAPE } =
{ 'physical_backup_device_name' | @physical_backup_device_name_var }
}

```

Although the **MEDIA**NAME qualifier is not available in ASE, the **DUMPVOLUME** qualifier may be substituted to identify a volume label. **DUMPVOLUME** is restricted to 6 characters. Also note, the “disk”, “tape”, and “pipe” qualifiers to the backup device are not necessary in ASE.

The following table lists the MSSQL DBCC subcommands that have either no equivalent in ASE or a limited equivalent.

MSSQL DBCC Command	ASE Equivalent
CHECKFILEGROUP CHECKIDENT INPUTBUFFER OUTPUTBUFFER	No equivalent
PHYSICAL_ONLY SHOWCONTIG SHRINKDATABASE SHRINKFILE UPDATEUSAGE USEROPTIONS	
dllname (FREE)	sp_freedll
OPENTRAN	Query master..systransactions or master..syslogshold
PINTABLE UNPINTABLE	sp_cacheconfig, with sp_bindcache
PROCCACHE	PROCBUF
SHOW_STATISTICS	optdiag utility

## System Table Differences

This section displays the system tables in common and different between MSSQL and ASE. Though they share many of the same tables in master and other databases, the details of each table are slightly different. For more information on each, see SQL Server Books Online for MSSQL and the Reference Manual for ASE.

### Master Database

System Table	Supported By	Use
sysaltfiles	MSSQL	Special-circumstance files; ASE equivalent is sysdevices
syscacheobjects	MSSQL	Composition of cache
syscharsets	MSSQL, ASE	Character sets or sort orders
sysconfigures	MSSQL, ASE	Configuration parameters set by users
syscurconfigs	MSSQL, ASE	Configuration parameters in effect
sysdatabases	MSSQL, ASE	Databases within the server
sysdevices	MSSQL, ASE	Database disk devices (files or raw partitions) and tape and disk dump devices and for ASE. Database file information for MSSQL
sysengines	ASE	Information about each ASE engine process
syslanguages	MSSQL, ASE	Languages known to server
syslisteners	ASE	Active network connection listeners
syslockinfo	MSSQL	Active locks; similar to ASE syslocks
syslocks	ASE	Active locks
sysloginroles	ASE	Roles assigned to logins
syslogins	MSSQL, ASE	Login, password and authentication information
syslogshold	ASE	Oldest open transactions and Replication Server truncation points
sysmessages	MSSQL, ASE	System error or warning messages
sysmonitors	ASE	Monitor counters used by sp_sysmon and Monitor Server
sysoledbusers	MSSQL	User and password information for linked servers
sysperfinfo	MSSQL	Performance counters for Windows Perfmon
sysprocesses	MSSQL, ASE	Connected user and background system processes
sysremotelogins	MSSQL, ASE	Remote logins and passwords
sysresourcelimits	ASE	Resource limits

syssecmechs	ASE	Security services for each security mechanism
sys.servers	MSSQL, ASE	Local and remote servers
sys.sessions	ASE	User connections for ASE High Availability option
sys.srvrole	ASE	Server-wide system and user-defined roles
sys.timeranges	ASE	Defined time ranges for the resource limits
sys.transactions	ASE	Active transactions
sys.usages	ASE	Disk pieces (portions of a device allocated to a database)

### All Databases

System Table	Supported By	Use
sys.alternates	ASE	ASE logins aliased to existing database users
sys.attributes	ASE	Object attributes
sys.columns	MSSQL, ASE	Columns in tables and views, and parameters in procedures.
sys.comments	MSSQL, ASE	SQL text for views, rules, defaults, triggers, and procedures
sys.constraints	MSSQL, ASE	Referential and check constraints for tables and columns
sys.depends	MSSQL, ASE	Procedure, view, or table referenced by procedure, view, or trigger
sys.filegroups	MSSQL	Filegroups in database. ASE equivalent is in <code>sys.segments</code> , <code>master..sys.devices</code> and <code>master..sys.usages</code>
sys.files	MSSQL	Files in a database. ASE equivalent is in <code>master..sys.devices</code> and <code>master..sys.usages</code>
sys.foreignkeys	MSSQL	Foreign key constraints. ASE equivalent table is <code>syskeys</code> .
sys.fulltextcatalogs	MSSQL	Full-text catalogs for database
sys.gams	ASE	Bitmaps for full and available Allocation Units in a database
sys.indexes	MSSQL, ASE	Clustered or nonclustered indexes, tables without clustered indexes, tables containing text or image data
sys.indexkeys	MSSQL	Columns in index; ASE equivalent is stored in <code>sys.indexes</code>
sys.jars	ASE	Java archive (JAR) files
sys.keys	ASE	Primary, foreign, or common key set by user
sys.logs	ASE	Transaction log
sys.members	MSSQL	Members of database roles
sys.objects	MSSQL, ASE	Tables, views, procedures, rules, triggers, defaults, and (in tempdb only) temporary objects
sys.partitions	ASE	Partition (page chain) of a partitioned table
sys.permissions	MSSQL	Permissions granted to users, groups, and roles. ASE equivalent table is <code>sys.protects</code>
sys.procedures	ASE	Query tree (binary compilation) for views, rules, defaults, triggers, and procedures
sys.protects	MSSQL, ASE	User permissions information through GRANT and REVOKE
sys.queryplans	ASE	Abstract query plans and SQL text
sys.references	MSSQL, ASE	Referential integrity constraints declared on tables and columns
sys.roles	ASE	Server role to local database group associations
sys.segments	ASE	Segments (named collection of disk pieces)
sys.statistics	ASE	Statistics for columns
sys.tabstats	ASE	Statistics for tables and indexes
sys.thresholds	ASE	Thresholds defined for the database's segments
sys.types	MSSQL, ASE	System-supplied and user-defined datatypes
sys.usermessages	ASE	User-defined messages
sys.users	MSSQL, ASE	Users allowed in the database
sys.types	ASE	Extended SQL and Java class-based datatypes



## International Contacts

<b>Argentina</b> +5411 4313 4488	<b>Korea</b> +82 2 3451 5200
<b>Australia</b> +612 9936 8800	<b>Malaysia</b> +603 2142 4218
<b>Austria</b> +43 1 504 8510	<b>Mexico</b> +5255 5093 8500
<b>Belgium</b> +32 2 713 15 03	<b>Netherlands</b> +31 20 346 9290
<b>Brazil</b> +5511 3046 7388	<b>New Zealand</b> +64 4473 3661
<b>Bulgaria</b> +359 2 986 1287	<b>Nigeria</b> +234 12 62 5120
<b>Canada</b> +905 273 8500	<b>Norway</b> +47 231 621 45
<b>Central America</b> +506 204 7151	<b>Panama</b> +507 263 4349
<b>Chile</b> +56 2 330 6700	<b>Peru</b> +51 1 221 4190
<b>China</b> +8610 6856 8488	<b>Philippines</b> +632 750 2550
<b>Colombia</b> +57 1 218 8266	<b>Poland</b> +48 22 844 55 55
<b>Croatia</b> +385 42 33 1812	<b>Portugal</b> +351 21 424 6710
<b>Czech Republic</b> +420 2 24 31 08 08	<b>Puerto Rico</b> +787 289 7895
<b>Denmark</b> +45 3927 7913	<b>Romania</b> +40 1 231 08 70
<b>Ecuador</b> +59 322 508 593	<b>Russian Federation</b> +7 095 797 4774
<b>El Salvador</b> +503 245 1128	<b>Slovak Republic</b> +421 26 478 2281
<b>Finland</b> +358 9 7250 200	<b>Slovenia</b> +385 42 33 1812
<b>France</b> +33 1 41 91 96 80	<b>South Africa</b> +27 11 804 3740
<b>Germany</b> +49 69 9508 6182	<b>South Korea</b> +82 2 3451 5200
<b>Greece</b> +30 1 98 89 300	<b>Spain</b> +34 91 749 7605
<b>Guatemala</b> +502 366 4348	<b>Sweden</b> +46 8 568 512 00
<b>Honduras</b> +504 239 5483	<b>Switzerland</b> +41 1 800 9220
<b>Hong Kong</b> +852 2506 6000	<b>Taiwan</b> +886 2 2715 6000
<b>Hungary</b> +36 1 248 2919	<b>Thailand</b> +662 618 8638
<b>India</b> +91 22 655 0258	<b>Turkey</b> +90 212 325 4114
<b>Indonesia</b> +62 21 526 7690	<b>Ukraine</b> +380 44 227 3230
<b>Israel</b> +972 3 548 3555	<b>United Arab Emirates</b> +971 2 627 5911
<b>Italy</b> +39 02 696 820 64	<b>United Kingdom</b> +44 870 240 2255
<b>Ivory Coast</b> +225 22 43 73 73	<b>Venezuela</b> +58 212 267 5670
<b>Japan</b> +81 3 5210 6000	<b>Asian Solutions Center</b> +852 2506 8700
<b>Kazakstan</b> +7 3272 64 1566	

For other Europe, Middle East, or Africa inquiries:  
+33 1 41 90 41 64 (Sybase Europe)

For other Asia Pacific inquiries:  
+852 2506 8700 (Hong Kong)

For other Latin America inquiries:  
+925 236 6820



### Sybase, Inc. Worldwide Headquarters

One Sybase Drive  
Dublin, CA 94568-7902 USA  
Tel: +800 8 SYBASE  
www.sybase.com

Copyright ©2003 by Sybase, Inc. All rights reserved. This White Paper is for informational purposes only. Sybase makes no warranties, express or implied in this document. The information in this document is subject to change without notice. Sybase, Adaptive Server, Backup Server, Replication Server, PowerDesigner, PowerTransfer and Sybase Central are trademarks of Sybase, Inc. 6/97.

All other company and product names used herein may be the trademarks or registered trademarks of their respective companies. Printed in Canada.

This Technical White Paper was prepared by Sybase, Inc. L02339 ML5569