# 

1.2.1         [Changing varchar(m) to varchar(n)](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#1.2.1)  
1.2.2         [Frequently asked questions on Table partitioning](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#1.2.2)   
1.2.3         [How do I manually drop a table?](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#1.2.3)   
1.2.4         [Why not create all my columns varchar(255)?](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#1.2.4)   
1.2.5         [What's a good example of a transaction?](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#1.2.5)   
1.2.6         [What's a natural key?](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#1.2.6)   
1.2.7         [Making a Stored Procedure invisible](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#1.2.7)   
1.2.8         [Saving space when inserting rows monotonically](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#1.2.8)   
1.2.9         [How to compute database fragmentation](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#1.2.9)   
1.2.10       [Tasks a DBA should do...](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#1.2.10)   
1.2.11       [How to implement database security](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#1.2.11)   
1.2.12       [How to shrink a database](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#1.2.12)   
1.2.13       [How do I turn on auditing of all SQL text sent to the server](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#1.2.13)  
1.2.14       [sp\_helpdb/sp\_helpsegment is returning negative numbers](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#1.2.14)

# 1.2.1: Changing varchar(m) to varchar(n)

Before you start:

select max(datalength(column\_name))

from *affected\_table*

In other words, **please** be sure you're going into this with your head on straight.

### How To Change System Catalogs

This information is **Critical To The Defense Of The Free World**, and you would be **Well Advised To Do It Exactly As Specified**:

use master

go

sp\_configure "allow updates", 1

go

reconfigure with override /\* System 10 and below \*/

go

use *victim\_database*

go

select name, colid

from syscolumns

where id = object\_id("*affected\_table*")

go

begin tran

go

update syscolumns

set length = *new\_value*

where id = object\_id("*affected\_table*")

and colid = *value\_from\_above*

go

update sysindexes

set maxlen = maxlen + *increase/decrease?*

where id=object\_id("*affected\_table*")

and indid = 0

go

/\* check results... cool? Continue... else **rollback tran** \*/

commit tran

go

use master

go

sp\_configure "allow updates", 0

go

reconfigure /\* System 10 and below \*/

go

[Return to top](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#UserDatabaseAdmin)

# 1.2.2: FAQ on partitioning

### Index of Sections

* [What Is Table Partitioning?](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#part1)
  + [Page Contention for Inserts](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#part1.1)
  + [I/O Contention](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#part1.2)
  + [Caveats Regarding I/O Contention](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#part1.3)
* [Can I Partition Any Table?](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#part2)
  + [How Do I Choose Which Tables To Partition?](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#part2.1)
* [Does Table Partitioning Require User-Defined Segments?](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#part3)
* [Can I Run Any Transact-SQL Command on a Partitioned Table?](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#part4)
* [How Does Partition Assignment Relate to Transactions?](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#part5)
* [Can Two Tasks Be Assigned to the Same Partition?](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#part6)
* [Must I Use Multiple Devices to Take Advantage of Partitions?](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#part7)
* [How Do I Create A Partitioned Table That Spans Multiple Devices?](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#part8)
* [How Do I Take Advantage of Table Partitioning with bcp in?](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#part9)
* [Getting More Information on Table Partitioning](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#part10)

#### What Is Table Partitioning?

Table partitioning is a procedure that creates multiple page chains for a single table.

The primary purpose of table partitioning is to improve the performance of concurrent inserts to a table by reducing contention for the last page of a page chain.

Partitioning can also potentially improve performance by making it possible to distribute a table's I/O over multiple database devices.

##### Page Contention for Inserts

By default, ASE stores a table's data in one double-linked set of pages called a page chain. If the table does **not** have a clustered index, ASE makes all inserts to the table in the last page of the page chain.

When a transaction inserts a row into a table, ASE holds an exclusive page lock on the last page while it inserts the row. If the current last page becomes full, ASE allocates and links a new last page.

As multiple transactions attempt to insert data into the table at the same time, performance problems can occur. Only one transaction at a time can obtain an exclusive lock on the last page, so other concurrent insert transactions block each other.

Partitioning a table creates multiple page chains (partitions) for the table and, therefore, multiple last pages for insert operations. A partitioned table has as many page chains and last pages as it has partitions.

##### I/O Contention

Partitioning a table can improve I/O contention when ASE writes information in the cache to disk. If a table's segment spans several physical disks, ASE distributes the table's partitions across fragments on those disks when you create the partitions.

A fragment is a piece of disk on which a particular database is assigned space. Multiple fragments can sit on one disk or be spread across multiple disks.

When ASE flushes pages to disk and your fragments are spread across different disks, I/Os assigned to different physical disks can occur in parallel.

To improve I/O performance for partitioned tables, you must ensure that the segment containing the partitioned table is composed of fragments spread across multiple physical devices.

##### Caveats Regarding I/O Contention

Be aware that when you use partitioning to balance I/O you run the risk of disrupting load balancing even as you are trying to achieve it. The following scenarios can keep you from gaining the load balancing benefits you want:

* You are partitioning an existing table. The existing data could be sitting on any fragment. Because partitions are randomly assigned, you run the risk of filling up a fragment. The partition will then steal space from other fragments, thereby disrupting load balancing.
* Your fragments differ in size.
* The segment maps are configured such that other objects are using the fragments to which the partitions are assigned.
* A very large bcp job inserts many rows within a single transaction. Because a partition is assigned for the lifetime of a transaction, a huge amount of data could go to one particular partition, thus filling up the fragment to which that partition is assigned.

#### Can I Partition Any Table?

No. You cannot partition the following kinds of tables:

1. Tables with clustered indexes (as of release 11.5 it is possible to have a clustered index on a partitioned table)
2. ASE system tables
3. Work tables
4. Temporary tables
5. Tables that are already partitioned. However, you can unpartition and then re-partition tables to change the number of partitions.

##### How Do I Choose Which Tables To Partition?

You should partition heap tables that have large amounts of concurrent insert activity. (A heap table is a table with no clustered index.) Here are some examples:

1. An "append-only" table to which every transaction must write
2. Tables that provide a history or audit list of activities
3. A new table into which you load data with bcp in. Once the data is loaded in, you can unpartition the table. This enables you to create a clustered index on the table, or issue other commands not permitted on a partition table.

#### Does Table Partitioning Require User-Defined Segments?

No. By design, each table is intrinsically assigned to one segment, called the default segment. When a table is partitioned, any partitions on that table are distributed among the devices assigned to the default segment.

In the example under "How Do I Create A Partitioned Table That Spans Multiple Devices?", the table sits on a user-defined segment that spans three devices.

#### Can I Run Any Transact-SQL Command on a Partitioned Table?

No. Once you have partitioned a table, you cannot use any of the following Transact-SQL commands on the table until you unpartition it:

1. drop table
2. sp\_placeobject
3. truncate table
4. alter table table\_name partition n

On releases of ASE prior to 11.5 it was not possible to create a clustered index on a partitioned table either.

#### How Does Partition Assignment Relate to Transactions?

A user is assigned to a partition for the duration of a transaction. Assignment of partitions resumes with the first insert in a new transaction. The user holds the lock, and therefore partition, until the transaction ends.

For this reason, if you are inserting a great deal of data, you should batch it into separate jobs, each within its own transaction. See "How Do I Take Advantage of Table Partitioning with bcp in?", for details.

#### Can Two Tasks Be Assigned to the Same Partition?

Yes. ASE randomly assigns partitions. This means there is always a chance that two users will vie for the same partition when attempting to insert and one would lock the other out.

The more partitions a table has, the lower the probability of users trying to write to the same partition at the same time.

#### Must I Use Multiple Devices to Take Advantage of Partitions?

It depends on which type of performance improvement you want.

Table partitioning improves performance in two ways: primarily, by decreasing page contention for inserts and, secondarily, by decreasing i/o contention. "What Is Table Partitioning?" explains each in detail.

If you want to decrease page contention you do not need multiple devices. If you want to decrease i/o contention, you must use multiple devices.

#### How Do I Create A Partitioned Table That Spans Multiple Devices?

Creating a partitioned table that spans multiple devices is a multi-step procedure. In this example, we assume the following:

* We want to create a new segment rather than using the default segment.
* We want to spread the partitioned table across three devices, data\_dev1, data\_dev2, and data\_dev3.

Here are the steps:

1. Define a segment:

sp\_addsegment newsegment, my\_database,data\_dev1

1. Extend the segment across all three devices:

sp\_extendsegment newsegment, my\_database, data\_dev2  
sp\_extendsegment newsegment, my\_database, data\_dev3

1. Create the table on the segment:

create table my\_table  
(names, varchar(80) not null)  
on newsegment

1. Partition the table:

alter table my\_table partition 30

#### How Do I Take Advantage of Table Partitioning with bcp in?

You can take advantage of table partitioning with bcp in by following these guidelines:

1. Break up the data file into multiple files and simultaneously run each of these files as a separate bcp job against one table.

Running simultaneous jobs increases throughput.

1. Choose a number of partitions greater than the number of bcp jobs.

Having more partitions than processes (jobs) decreases the probability of page lock contention.

1. Use the batch option of bcp in. For example, after every 100 rows, force a commit. Here is the syntax of this command:

bcp table\_name in filename -b100

Each time a transaction commits, ASE randomly assigns a new partition for the next insert. This, in turn, reduces the probability of page lock contention.

#### Getting More Information on Table Partitioning

For more information on table partitioning, see the chapter on controlling physical data placement in the ASE Performance and Tuning Guide.

[Return to top](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#UserDatabaseAdmin)

# 1.2.3: How to manually drop a table

Occasionally you may find that after issuing a *drop table* command that the ASE crashed and consequently the table didn't drop entirely. Sure you can't see it but that sucker is still floating around somewhere.

Here's a list of instructions to follow when trying to drop a corrupt table:

1. sp\_configure allow, 1
2. go
3. reconfigure with override
4. go
6. Write *db\_id* down.
7. use *db\_name*
8. go
9. select db\_id()

go

1. Write down the *id* of the **bad\_table**:
2. select id
3. from sysobjects
4. where name = *bad\_table\_name*

go

1. You will need these index IDs to run *dbcc extentzap*. Also, remember that if the table has a clustered index you will need to run *extentzap* on index "0", even though there is no sysindexes entry for that indid.
2. select indid
3. from sysindexes
4. where id = *table\_id*

go

1. This is not required but a good idea:
2. begin transaction

go

1. Type in this short script, this gets rid of all system catalog information for the object, including any object and procedure dependencies that may be present.

Some of the entries are unnecessary but better safe than sorry.

declare @obj int

select @obj = id from sysobjects where name =

delete syscolumns where id = @obj

delete sysindexes where id = @obj

delete sysobjects where id = @obj

delete sysprocedures where id in

(select id from sysdepends where depid = @obj)

delete sysdepends where depid = @obj

delete syskeys where id = @obj

delete syskeys where depid = @obj

delete sysprotects where id = @obj

delete sysconstraints where tableid = @obj

delete sysreferences where tableid = @obj

delete sysdepends where id = @obj

go

1. Just do it!
2. commit transaction

go

1. Gather information to run *dbcc extentzap*:
2. use master
3. go
4. sp\_dboption *db\_name*, read, true
5. go
6. use *db\_name*
7. go
8. checkpoint

go

1. Run *dbcc extentzap* once for **each** index (including index 0, the data level) that you got from above:
2. use master
3. go
4. dbcc traceon (3604)
5. go
6. dbcc extentzap (*db\_id*, *obj\_id*, *indx\_id*, 0)
7. go
8. dbcc extentzap (*db\_id*, *obj\_id*, *indx\_id*, 1)

go

Notice that extentzap runs **twice** for each index. This is because the last parameter (the *sort* bit) might be 0 or 1 for each index, and you want to be absolutely sure you clean them all out.

1. Clean up after yourself.
2. sp\_dboption *db\_name*, read, false
3. go
4. use *db\_name*
5. go
6. checkpoint
7. go
8. sp\_configure allow, 0
9. go
10. reconfigure with override

go

[Return to top](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#UserDatabaseAdmin)

# 1.2.4: Why not max out all my columns?

People occasionally ask the following valid question:

Suppose I have varying lengths of character strings none of which should exceed 50 characters.

**Is there any advantage of last\_name varchar(50) over this last\_name varchar(255)?**

That is, for simplicity, can I just define all my varying strings to be varchar(255) without even thinking about how long they may actually be? Is there any storage or performance penalty for this.

There is no performance penalty by doing this but as another netter pointed out:

If you want to define indexes on these fields, then you should specify the smallest size because the sum of the maximal lengths of the fields in the index can't be greater than 256 bytes.

and someone else wrote in saying:

Your data structures should match the business requirements. This way the data structure themselves becomes a data dictionary for others to model their applications (report generation and the like).

[Return to top](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#UserDatabaseAdmin)

# 1.2.5: What's a good example of a transaction?

This answer is geared for Online Transaction Processing (OTLP) applications.

To gain maximum throughput all your transactions should be in stored procedures - see [Q1.5.8](http://www.isug.com/Sybase_FAQ/ASE/section1.5.html#1.5.8). The transactions within each stored procedure should be short and simple. All validation should be done outside of the transaction and only the modification to the database should be done within the transaction. Also, don't forget to name the transaction for *sp\_whodo* - see [Q9.2](http://www.isug.com/Sybase_FAQ/ASE/section9.html#9.2).

The following is an example of a *good* transaction:

/\* perform validation \*/

select ...

if ... /\* error \*/

/\* give error message \*/

else /\* proceed \*/

begin

begin transaction acct\_addition

update ...

insert ...

commit transaction acct\_addition

end

The following is an example of a *bad* transaction:

begin transaction poor\_us

update X ...

select ...

if ... /\* error \*/

/\* give error message \*/

else /\* proceed \*/

begin

update ...

insert ...

end

commit transaction poor\_us

This is bad because:

* the first update on table X is held throughout the transaction. The idea with OLTP is to get in and out *fast*.
* If an error message is presented to the end user and we await their response, we'll maintain the lock on table X until the user presses return. If the user is out in the can we can wait for hours.

[Return to top](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#UserDatabaseAdmin)

# 1.2.6: What's a natural key?

Let me think back to my database class... okay, I can't think that far so I'll paraphrase... essentially, a *natural key* is a key for a given table that uniquely identifies the row. It's natural in the sense that it follows the business or real world need.

For example, assume that social security numbers are unique (I believe it is strived to be unique but it's not always the case), then if you had the following employee table:

employee:

ssn char(09)

f\_name char(20)

l\_name char(20)

title char(03)

Then a natural key would be *ssn*. If the combination of *\_name* and *l\_name* were unique at this company, then another *natural key* would be *f\_name, l\_name*. As a matter of fact, you can have many *natural keys* in a given table but in practice what one does is build a surrogate (or artificial) key.

The surrogate key is guaranteed to be unique because (wait, get back, here it goes again) it's typically a monotonically increasing value. Okay, my mathematician wife would be proud of me... really all it means is that the key is increasing linearly: i+1

The reason one uses a surrogate key is because your joins will be faster.

If we extended our employee table to have a surrogate key:

employee:

id identity

ssn char(09)

f\_name char(20)

l\_name char(20)

title char(03)

Then instead of doing the following:

where a.f\_name = b.f\_name

and a.l\_name = a.l\_name

we'd do this:

where a.id = b.id

We can build indexes on these keys and since Sybase's atomic storage unit is 2K, we can stash more values per 2K page with smaller indexes thus giving us better performance (imagine the key being 40 bytes versus being say 4 bytes... how many 40 byte values can you stash in a 2K page versus a 4 byte value? -- and how much wood could a wood chuck chuck, if a wood chuck could chuck wood?)

### Does it have anything to do with natural joins?

Um, not really... from "A Guide to Sybase..", McGovern and Date, p. 112:

The equi-join by definition must produce a result containing two identical columns. If one of those two columns is eliminated, what is left is called the natural join.

[Return to top](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#UserDatabaseAdmin)

# 1.2.7: Making a Stored Procedure invisible

### System 11.5 and above

It is now possible to encrypt your stored procedure code that is stored in the syscomments table. This is preferred than the old method of deleting the data as deleting will impact future upgrades. You can encrypt the text with the sp\_hidetext system procedure.

### Pre-System 11.5

Perhaps you are trying to prevent the buyer of your software from defncopy'ing all your stored procedures. It is perfectly safe to delete the syscomments entries of any stored procedures you'd like to protect:

sp\_configure "allow updates", 1

go

reconfigure with override /\* System 10 and below \*/

go

use *affected\_database*

go

delete syscomments where id = object\_id("*procedure\_name*")

go

use master

go

sp\_configure "allow updates", 0

go

I believe in future releases of Sybase we'll be able to *see* the SQL that is being executed. I don't know if that would be simply the stored procedure name or the SQL itself.

[Return to top](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#UserDatabaseAdmin)

# 1.2.8: Saving space when inserting rows monotonically

If the columns that comprise the clustered index are monotonically increasing (that is, new row key values are greater than those previously inserted) the following System 11 dbcc tune will not split the page when it's half way full. Rather it'll let the page fill and then allocate another page:

dbcc tune(ascinserts, 1, "*my\_table*")

By the way, SyBooks is wrong when it states that the above needs to be reset when ASE is rebooted. This is a permanent setting.

To undo it:

dbcc tune(ascinserts, 0, "*my\_table*")

[Return to top](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#UserDatabaseAdmin)

# 1.2.9: How to compute database fragmentation

## Command

dbcc traceon(3604)

go

dbcc tab(production, *my\_table*, 0)

go

## Interpretation

A delta of one means the next page is on the same track, two is a short seek, three is a long seek. You can play with these constants but they aren't that important.

A table I thought was unfragmented had L1 = 1.2 L2 = 1.8

A table I thought was fragmented had L1 = 2.4 L2 = 6.6

## How to Fix

You fix a fragmented table with clustered index by dropping and creating the index. This measurement isn't the correct one for tables without clustered indexes. If your table doesn't have a clustered index, create a dummy one and drop it.

[Return to top](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#UserDatabaseAdmin)

# 1.2.10: Tasks a DBA should do...

A good presentation of a DBA's duties has been made available by Jeff Garbus ([jeffg@soaringeagleltd.com](mailto:jeffg@soaringeagleltd.com)) of Soaring Eagle Consulting Ltd (<http://www.soaringeagleltd.com/>) and numerous books can be found [here](http://www.midsomer.org/presentations/JG/dba_tasks).  These are Powerpoint slides converted to web pages and so may be difficult to view with a text browser!

An alternative view is catalogued below.  (OK, so this list is crying out for a bit of a revamp since checkstorage came along  Ed!)

|  |  |  |
| --- | --- | --- |
| DBA Tasks | | |
| **Task** | **Reason** | **Period** |
| dbcc checkdb, checkcatalog, checkalloc | I consider these the minimal dbcc's to ensure the integrity of your database | If your ASE permits, daily before your database dumps. If this is not possible due to the size of your databases, then try the different options so that the end of, say, a week, you've run them all. |
| Disaster recovery scripts - scripts to rebuild your ASE in case of hardware failure | Always be prepared for the worst. Make sure to test them. |  |
| scripts to logically dump your master database, that is bcp the critical system tables: sysdatabases, sysdevices, syslogins, sysservers, sysusers, syssegments, sysremotelogins | You can selectively rebuild your database in case of hardware failure | Daily |
| %ls -la <disk\_devices> | A system upgrade is known to change the permissions. | After any change as well as daily |
| dump the user databases | CYA\* | Daily |
| dump the transaction logs | CYA | Daily |
| dump the master database | CYA | After any change as well as daily |
| System 11 and beyond - save the $DSQUERY.cfg to tape | This is the configuration that you've dialed in, why redo the work? | After any change as well as daily |
| update statistics on frequently changed tables and sp\_recompile | To ensure the performance of your ASE | Depending on how often your major tables change. Some tables are pretty much static (e.g. lookup tables) so they don't need an update statistics, other tables suffer severe trauma (e.g. massive updates/deletes/inserts) so an update stats needs to be run either nightly/weekly/monthly. This should be done using cronjobs. |
| create a dummy ASE and do bad things to it: delete devices, destroy permissions... | See disaster recovery! | When time permits |
| Talk to the application developers. | It's better to work with them than against them. | As time permits. |
| Learn new tools | So you can sleep! | As time permits. |
| Read comp.databases.sybase | Passes the time. | Priority One! |

\* Cover Your Ass

[Return to top](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#UserDatabaseAdmin)

# 1.2.11: How to implement database security

This is a brief run-down of the features and ideas you can use to implement database security:

## Logins, Roles, Users, Aliases and Groups

* sp\_addlogin - Creating a login adds a basic authorisation for an account - a username and password - to connect to the server. By default, no access is granted to any individual databases.
* sp\_adduser - A user is the addition of an account to a specific database.
* sp\_addalias - An alias is a method of allowing an account to use a specific database by impersonating an existing database user or owner.
* sp\_addgroup - Groups are collections of users at the database level. Users can be added to groups via the sp\_adduser command.

A user can belong to only one group - a serious limitation that Sybase might be addressing soon according to the ISUG enhancements requests. Permissions on objects can be granted or revoked to or from users or groups.

* sp\_role - A role is a high-level Sybase authorisation to act in a specific capacity for administration purposes. Refer to the Sybase documentation for details.

### Recommendations

Make sure there is a unique login account for each physical person and/or process that uses the server. Creating generic logins used by many people or processes is a *bad idea* - there is a loss of accountability and it makes it difficult to track which particular person is causing server problems when looking at the output of sp\_who. Note that the output of sp\_who gives a hostname - properly coded applications will set this value to something meaningful (ie. the machine name the client application is running from) so you can see where users are running their programs. Note also that if you look at master..sysprocesses rather than just sp\_who, there is also a program\_name. Again, properly coded applications will set this (eg. to 'isql') so you can see which application is running. If you're coding your own client applications, make sure you set hostname and program\_name via the appropriate Open Client calls. One imaginative use I've seen of the program\_name setting is to incorporate the connection time into the name, eg APPNAME-DDHHMM (you have 16 characters to play with), as there's no method of determining this otherwise.

Set up groups, and add your users to them. It is much easier to manage an object permissions system in this way. If all your permissions are set to groups, then adding a user to the group ensures that users automatically inherit the correct permissions - administration is \*much\* simpler.

## Objects and Permissions

Access to database objects is defined by granting and/or revoking various access rights to and from users or groups. Refer to the Sybase documentation for details.

### Recommendations

The ideal setup has all database objects being owned by the dbo, meaning no ordinary users have any default access at all. Specific permissions users require to access the database are granted explicitly. As mentioned above - set permissions for objects to a group and add users to that group. Any new user added to the database via the group then automatically obtains the correct set of permissions.

Preferably, no access is granted at all to data tables, and all read and write activity is accomplished through stored procedures that users have execute permission on. The benefit of this from a security point of view is that access can be rigidly controlled with reference to the data being manipulated, user clearance levels, time of day, and anything else that can be programmed via T-SQL. The other benefits of using stored procedures are well known (see [Q1.5.8](http://www.isug.com/Sybase_FAQ/ASE/section1.5.html#1.5.8)). Obviously whether you can implement this depends on the nature of your application, but the vast majority of in-house-developed applications can rely solely on stored procedures to carry out all the work necessary. The only server-side restriction on this method is the current inability of stored procedures to adequately handle text and image datatypes (see [Q1.5.12](http://www.isug.com/Sybase_FAQ/ASE/section1.5.html#1.5.12)). To get around this views can be created that expose only the necessary columns to direct read or write access.

## Views

Views can be a useful general security feature. Where stored procedures are inappropriate views can be used to control access to tables to a lesser extent. They also have a role in defining row-level security - eg. the underlying table can have a security status column joined to a user authorisation level table in the view so that users can only see data they are cleared for. Obviously they can also be used to implement column-level security by screening out sensitive columns from a table.

## Triggers

Triggers can be used to implement further levels of security - they could be viewed as a last line of defence in being able to rollback unauthorised write activity (they cannot be used to implement any read security). However, there is a strong argument that triggers should be restricted to doing what they were designed for - implementing referential integrity - rather being loaded up with application logic.

## Administrative Roles

With Sybase version 10 came the ability to grant certain administrative roles to user accounts. Accounts can have sa-level privilege, or be restricted to security or operator roles - see sp\_role.

### Recommendations

The use of any generic account is not a good idea. If more than one person requires access as sa to a server, then it is more accountable and traceable if they each have an individual account with sa\_role granted.

[Return to top](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#UserDatabaseAdmin)

# 1.2.12: How to Shrink a Database

**Warning: This document has not been reviewed. Treat it as alpha-test quality information and report any problems and suggestions to bret@sybase.com**

It has historically been difficult to shrink any database except tempdb (because it is created fresh every boot time). The two methods commonly used have been:

1. Ensure that you have scripts for all your objects (some tools like SA Companion, DB Artisan or [dbschema.pl](ftp://ftp.midsomer.org/pub/dbschema.tgz) from Sybperl can create scripts from an existing database), then bcp out your data, drop the database, recreate it smaller, run your scripts, and bcp in your data.
2. Use a third-party tool such as DataTool's SQL Backtrack, which in essence automates the first process.

This technote outlines a third possibility that can work in most cases.

### An Unsupported Method to Shrink a Database

This process is fairly trivial in some cases, such as removing a recently added fragment or trimming a database that has a log fragment as its final allocation, but can also be much more complicated or time consuming than the script and bcp method.

#### General Outline

The general outline of how to do it is:

1. Make a backup of the current database
2. Migrate data from sysusages fragments with high lstart values to fragments with low lstart values.
3. Edit sysusages to remove high lstart fragments that no longer have data allocations.
4. Reboot ASE.

#### Details

1. Dump your database. If anything goes wrong, you will need to recover from this backup!
2. Decide how many megabytes of space you wish to remove from your database.
3. Examine sysusages for the database. You will be shrinking the database by removing the fragments with the highest lstart values. If the current fragments are not of appropriate sizes, you may need to drop the database, recreate it so there are more fragments, and reload the dump.

A trivial case: An example of a time when you can easily shrink a database is if you have just altered it and are sure there has been no activity on the new fragment. In this case, you can directly delete the last row in sysusages for the db (this row was just added by alter db) and reboot the server and it should come up cleanly.

1. Change the segmaps of the fragments you plan to remove to 0. This will prevent future data allocations to these fragments.

Note: If any of the fragments you are using have user defined segments on them, drop those segments before doing this.

sp\_configure "allow updates", 1

go

reconfigure with override  -- not necessary in System 11

go

update sysusages

set segmap = 0

where dbid = <dbid>

and lstart = <lstart>

go

dbcc dbrepair(<dbname>, remap)

go

Ensure that there is at least one data (segmap 3) and one log (segmap 4) fragment, or one mixed (segmap 7) fragment.

If the server has been in use for some time, you can shrink it by deleting rows from sysusages for the db, last rows first, after making sure that no objects have any allocations on the usages.

1. Determine which objects are on the fragments you plan to remove.
2. traceon(3604)
3. go
4. dbcc usedextents( dbid,0,0,1)

go

Find the extent with the same value as the lstart of the first fragment you plan to drop. You need to migrate every object appearing from this point on in the output.

1. Migrate these objects onto earlier fragments in the database.

Objids other than 0 or 99 are objects that you must migrate or drop. You can migrate a user table by building a new clustered index on the table (since the segmap was changed, the new allocations will not go on this fragment).

You can migrate some system tables (but not all) using the sp\_fixindex command to rebuild its clustered index. However, there are a few system tables that cannot have their clustered indexes rebuilt, and if they have any allocations on the usage, you are out of luck.

If the objid is 8, then it is the log. You can migrate the log by ensuring that another usage has a log segment (segmap 4 or 7). Do enough activity on the database to fill an extents worth of log pages, then checkpoint and dump tran.

Once you have moved all the objects, delete the row from sysusages and reboot the server.

Run dbcc checkdb and dbcc checkalloc on the database to be sure you are ok, then dump the database again.

[Return to top](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#UserDatabaseAdmin)

# 1.2.13: How do I audit the SQL sent to the server?

This does not seem to be well documented, so here is a quick means of auditing the SQL text that is sent to the server.  Note that this simply audits the SQL sent to the server.  So, if your user process executes a big stored procedure, all you will see here is a call to the stored procedure.  None of the SQL that is executed as part of the stored procedure will be listed.

Firstly, you need to have installed Sybase security (which involves installing the sybsecurity database and loading it using the script $SYBASE/scripts/installsecurity).   Read the Sybase Security Administration Manual, you may want to enable a threshold procedure to toggle between a couple of audit tables.  Be warned, that the default configuration option "suspend auditing when device full" is set to 1.   This means that the server will suspend all normal SQL operations if the audit database becomes full and the sso logs in and gets rid of some data.  You might want to consider changing this to 0 unless yours is a particularly sensitive installation.

Once that is done, you need to enable auditing.  If you haven't already, you will need to restart ASE in order to start the audit subsystem.  Then comes the bit that does not seem well documented, you need to select an appropriate audit option, and the one for the SQL text is "cmdtext".  From the sybsecurity database, issue

sp\_audit "cmdtext",<username>,"all","on"

for each user on the system that wish to collect the SQL for.  sp\_audit seems to imply that you can replace "<username>" with all, but I get the error message "'all' is not a valid user name".  Finally, enable auditing for the system as a whole using

sp\_configure "auditing",1  
go

If someone knows where in the manuals this is well documented, I will add a link/reference.

Note: The stored procedure sp\_audit had a different name under previous releases.  I think that it was called sp\_auditoption.  Also, to get a full list of the options and their names, go into sybsecurity and simply run sp\_audit with no arguments.

[Return to top](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#UserDatabaseAdmin)

# 1.2.14: sp\_helpdb/sp\_helpsegment is returning negative numbers

A number of releases of ASE return negative numbers for sp\_helpdb. One solution given by Sybase is to restart the server. Hmm... not always possible. An alternative is to use the dbcc command 'usedextents'. Issue the following:

dbcc traceon(3604)

dbcc usedextents(, 0, 1, 1)

and the problem should disappear. This is actually a solved case, Sybase solved case no: 10454336, go to <http://info.sybase.com/resolution/detail.stm?id_number=10454336> to see more information.

[Return to top](http://www.isug.com/Sybase_FAQ/ASE/section1.2.html#UserDatabaseAdmin)