Company X  
SQL Server Technical Documentation

Quick Reference for Support Questions 1

Our SQL Servers & Their Service Levels 3

Differences Between Tiers 3

Database Server Restrictions 6

Things That Require DBA Signoff 6

Things Not Available on Production Servers 7

Things Not Available On Any Server 8

Load Testing 10

Backups & Restores 11

Network Backups 11

Restoring Databases 12

High Availability & Disaster Recovery 13

Clustering 13

Database Mirroring 13

Change Control 17

Deployment Practices 17

Maintenance Windows & Patches 19

QA & Development Servers 19

Mission-Critical & Production Servers 19

When to Get Help from the DBA 20

SQL Coding Standards 21

Design for Database Mirroring 21

Table Design 21

Stored Procedures & Functions 22

SQL Server Setup Standards 23

SQL Server 2005 Setup Checklist 23

# Quick Reference for Support Questions

If you need to do any of these things:

* Create a database
* Add or change security permissions
* Install software on a database server
* Back up or restore a database

Then create a help desk ticket for Windows Systems.

In an emergency, contact the following people in this order:

* Lead DBA – Jane Doe – desk 281-555-1213, cell 281-555-9357
* Junior DBA – John Smith – desk 281-555-1215, cell 281-555-7725
* IT Manager – Fred Adams – desk 281-555-1209, cell 281-555-0395
* Company Help Desk – 1-800-555-1000. Ask to open a Severity 1 help desk ticket, and they can reach the next available member of the IT team.

### What’s In This Document?

This document is for technical staff (developers & network admins) that need info about our SQL Server 2005 database servers.

# Our SQL Servers & Their Service Levels

We have four sets of SQL Servers for different uses. The differences between the categories are shown below:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Mission- Critical Production** | **Production** | **QA/Testing** | **Dev** |
| **High-Availability Features** |  |  |  |  |
| Backed up to outside location (DR site) | Yes | No | No | No |
| Backed up to Local (Live site) | Weekly Fulls, nightly diffs, logs every 5 minutes | Weekly fulls, nightly diffs, hourly logs | No | Daily fulls |
| Days of backups avail on disk | 3 weeks | 2 weeks | 0 (no backups) | 3 days |
| Scheduled maintenance downtime | 3rd Saturday of the month, 11pm – 2am Eastern | Every Saturday, 11pm-9am Eastern | Every weekend with 4 hours notice | Every weekend with 4 hours notice |
| RPO Goal | 1 hour | 4 hours | n/a | n/a |
| RTO Goal | 1 hour | 1 day | n/a | n/a |
| Backup Location | \\backups\sql\_p1 | \\backups\sql\_p2 | n/a | Local drive |
| SQL Server Edition | Enterprise | Standard | Developer | Developer |
| DBA Support | 24/7 | 24/7 | Business Hrs | Business Hrs |
| **Restrictions** |  |  |  |  |
| Users can change tables/views | Yes | Yes | Yes | Yes |
| Users can restore dbs, kill procs | No | No\* | Yes | Yes |
| Users can be sysadmins | No | No\* | No\* | No\* |
| Users can be dbowners | Yes | Yes | Yes | Yes |
| Users can remote desktop into server | No | No\* | No\* | No\* |
| **Performance** |  |  |  |  |
| Performance level | High | High | Medium | Low |
| Performance troubleshooting | Detailed | Detailed | Limited | Limited |
| \* Note – these features are available if the database is installed in its own dedicated virtual machine (VM). | | | | |
| **Servers** |  |  |  |  |
| General Shared Use – (SQL 2012/2014) | PRODSQL04, 05 | PRODSQL01, 02, 03 | SQLQA4 | SQLDEV1, 2 |
| Data Warehouse (BI) | WAREHOUSE\_14 | n/a | (Uses shared one) | (Uses shared one) |
| Data Marts (Reporting) | n/a | n/a | n/a | n/a |
| Sharepoint | n/a | n/a | n/a | n/a |
| Point of Sale Systems | n/a | n/a | n/a | n/a |

## 

## Differences Between Tiers

### What’s the Difference Between Mission-Critical and Production Servers?

Mission-critical databases can be failed over to our Disaster Recovery site with a minimum of human intervention and without data loss. We use SQL Server 2008’s Database Mirroring feature to keep realtime copies of the databases in Memphis.

These features mean that mission-critical servers cost more because of the redundancy and network bandwidth required for the failover process.

Keep in mind that the applications that query these databases may still need human intervention in order to use the Memphis standby server. Applications compiled with the .NET Framework v2 can fail over automatically as long as they configure their connection string with the mirror server’s name.

### What’s The Difference Between QA servers and Dev servers?

The QA/Testing servers are for end user testing and for load testing. Before an application goes live, the developers need to be able to know with confidence that it will support the full load of real-life usage.

Developers can continue to work on the development servers while a static set of code is tested on the QA servers.

QA servers are only backed up weekly, not daily. They usually have the full set of production data, so backing up the entire server is expensive. No labor should be lost in the event of a machine crash because development work is stored in the development servers, and any changes necessary to go live in production should be scripted anyway and saved in TFS or Subversion.

The QA servers are not for short-term testing. They are for user testing that lasts a week or more. Think of it as the last step before real-world deployment, not the first step after the developer compiles.

The QA servers are also used for testing the deployment scripts for new versions. All changes to production servers must be scripted and handed off to the DBA for deployment. We can test these deployment scripts by restoring the production database onto QA, and then applying the deployment script. If everything works as expected, then we can go live with the script on the actual production servers.

### What’s the Difference Between Enterprise and Developer Editions?

On our QA & development servers, we use the Developer Edition because it offers all of the functionality of the Enterprise version, but at a fraction of the price. The only difference between Developer and Enterprise versions is that the Developer version cannot be used in production.

There is no speed difference between SQL Server 2005 Developer and SQL Server 2005 Enterprise.

If something runs slow on the development servers, it will probably run slow on the production servers as well. Yes, the production servers are faster, but remember that there are other people using the production servers simultaneously. Things that run slow with only one user on development will definitely run slow with lots of users on production.

**Why 4 hours of downtime per week on QA and dev servers?**

We install all operating system, SQL Server, and hardware patches first on the development and QA servers. These patches don’t always work perfectly: sometimes they produce unintended side effects, and we have to back the changes back out. In order to do that, we may have to restore the OS using techniques like VMware snapshots or SAN snapshots, which may take an hour or more to revert and test.

Ideally, we won’t need this time – but on occasion, we do, and on the development servers, we’ll actually do it during business hours.

This practice means that we don’t have to incur as much downtime on the production servers. We can say with confidence that installing a particular patch will only take X minutes, and that it won’t have any negative effects on the production environment.

### Why is CXSQLDV1 32-bit?

Since 2008, all new production SQL Servers have been 64-bit. Some development & QA servers will continue to be 32-bit because we’re taking advantage of leftover available hardware. We have a lot of 32-bit hardware we’re phasing out, and as we phase out older production hardware, we reuse the old production stuff as development servers.

# Database Server Restrictions

We share database servers across several departments in the company, and we want to make sure the servers are as reliable as possible and as fast as possible. In order to do that, we’ve put a few restrictions into place.

These restrictions are only designed to keep the servers up longer and running faster. If any of these restrictions will prevent you from doing what you need to do, by all means, talk to the database administrators. They want to provide you with the best platform possible to deploy your applications.

## Things That Require DBA Signoff

### Functions & Stored Procs with .NET Code

We’re a .NET shop and we definitely want to embrace the idea of .NET code, but we just need to make sure that we don’t turn the database servers into application servers.

Before coding a function or a stored procedure in a .NET language like VB.NET or C#, meet with the DBA, and be prepared to explain:

* What the code will do
* How many times a day the code will be called
* The maximum amount of time the code will run each time
* Possible alternatives for putting the code inside SQL, including the pros and cons of each. Show why putting it in SQL is the best choice.

We’re completely open to putting .NET code in the database tier, but we just want to make sure we’re taking the right approach first. We like to separate out CPU-intensive code to the application tier whenever possible, because it lets us scale easier. SQL Server Enterprise licensing costs $30k per CPU, so it’s important to make sure that CPU-intensive applications really do need to be run on the server. Otherwise, it’s more cost-effective to buy an entirely separate application server rather than burning up one CPU of SQL Server licensing.

### Reporting Services, Integration Services, Analysis Services, Notification Services

Again, we’re completely open to utilizing 2005/2008’s .NET-friendly features, but we just want to make sure we’re doing it the right way in the right places. These advanced services are pretty darned cool, but we have to plan for the additional load that they’ll cause.

If you want to use any of these features, meet with the DBA. You don’t have to be prepared for anything, just have an exploratory discussion and we’ll wing it.

With Integration Services, if we need to get data out of the AS400, we prefer to use IBM DataStage jobs for our ETL. It’s the system used by the BI team’s data warehouse on CXDWPROD1. They have a standard process they’ve put into place with extensive monitoring and testing, and we can reuse a lot of their work easily. For questions about it, meet with Jim Smith.

### SQL Agent Jobs

As with .NET code in stored procs and functions, we want to minimize the number of “processing” tasks that take place on the server. The database server is designed to store data, and we like to keep scheduled tasks on a separate application server.

We also monitor the servers for failed jobs around the clock. If any job fails, the DBA will be paged. We want to ensure that all jobs will be as reliable as possible.

Jobs are not automatically copied from one server to another, which makes things a little more challenging in a disaster-recovery scenario.

If something is more efficiently processed on the server itself, then meet with the DBA and be prepared to talk about:

* What the job will do
* How often it will run
* The maximum amount of time the job will run each time
* Who will be on call when the job fails
* What should happen during a disaster recovery scenario (meaning, do we still need to run the job, is it still critical, and do any changes need to be made)

## Things Not Available on Production Servers

### System-Wide Privileges (Windows RDP, etc)

The database servers are shared resources across multiple departments. It’s the DBA’s job to tune the servers to properly support everybody’s applications, as well as support the applications when they don’t perform. In order to do that, we have to keep some controls in place to make sure system settings don’t change.

Accounts can still be database owners (dbo) on individual databases, though.

### Tuning Advisor & Index Tuning Wizard

Your humble database administrators rely on these every day to make life easier, but we do this on development and QA servers, not production. These tools make minor temporary changes to the database schema and also add a lot of load, so to avoid affecting our end users, we don’t do this on production.

## Things Not Available On Any Server

### Applications or Software Installed on the Server

The SQL Servers are strictly for data storage. No applications or processing is run on these machines. The memory is tuned to use as much as possible for SQL Server itself, and every CPU cycle is precious on these boxes.

We have several application servers available where programs can be installed, and if we need additional app servers, that’s definitely an option.

### DTS Packages

We do not use DTS packages on the centralized database servers. DTS packages are deprecated with SQL Server 2005. If an application still needs DTS packages, we can install them on an existing SQL Server 2000 machine, or put the standalone Express Edition on the application server. Express is limited to small amounts of data, but for an installation that strictly does DTS packages, that shouldn’t be a problem since we’re not storing data locally anyway.

### ODBC Connections & Drivers

Since no applications and DTS packages are installed on the SQL Servers, this probably goes without saying, but we don’t need ODBC connections or drivers.

### Stored Procs & Functions in the System Databases

The master, model, msdb and tempdb databases are for system and DBA use only. Some utility stored procedures like sp\_WhoIsActive have been installed in master, but no other objects are allowed in these databases.

### Linked Server Queries

This is a bit of a trick – we do allow linked server connections, but only with a read-only account that is denied access to write any data on the remote servers. For example, on the development server, we can have a linked server set up to query the production server, but users on the development server will not be allowed to write any data to the production server.

This is for security purposes, because we want to keep these things locked down tighter than a duck’s butt. Linked servers CAN be set up securely, but it’s a maintenance hassle, and we don’t want to build apps that rely on this stuff. This is where ETL tools like DataStage and Integration Services comes in.

Keep in mind that linked server queries will fail in DRP mode. When we role swap with Memphis, queries that try to hit linked servers in New York will fail, because those servers will be shut off.

### Remote Desktop Connections

Keep your grubby hands off the database server. Okay, no, seriously. We mean it. Put them back in your pockets.

If there’s anything you think you need remote desktop access for, just check with the DBA and we’ll help you do it from your local machine instead. For example, if you need to do performance monitoring on a server to see if your spiffy new code brings the database server to its knees, we can show you how to use Perfmon on your local machine to monitor the server.

# Load Testing

### Email Your Friendly Database Administrator

When scheduling load tests, send the database administrators an email to let them know that you’ll be doing load testing. They can run a SQL profile and a Windows performance monitor during that time range to let you know about any particular bottlenecks.

They can also tell you if anyone else has scheduled testing runs during that same time span that might slow your testing down. The nightly backup jobs might make your load testing look artificially slow, or other developers may also be load testing their own applications on these servers as well.

# Backups & Restores

## Network Backups

The databases are backed up with either Quest Litespeed or native SQL Server backup plans. The flat file backups are saved in \\cxbackups\sql\ shares by server name, and then by database name.

Network backups allow us to recover the databases in the event of a machine failure. Otherwise, if the machines backed up to local drives, the drives may be unavailable when the OS crashes.

The Windows team then backs up the flat files to tape every day, and the tapes are sent offsite for safekeeping. For access to these older backups, contact the help desk. Be aware that tape restores are not a quick process – the tapes have to be retrieved, loaded, and then copied onto the network, a process which may take several days.

If you know that you need a particular day’s backup – for example, if you’re planning a system upgrade or migration – contact the DBA ahead of time, and the backup can be saved online for a few days.

### Backing Up With Quest Litespeed

Most of our servers use Litespeed to do their backups. It offers about an 80% compression ratio, which lets us store a 1tb database in 200gb of space.

While these backups run, they take up more CPU power on the database server. It still pays off by letting us get the backups done faster, since it takes less time to copy the databases over the network.

Litespeed has its own management user interface. For access, check with the database administrators.

It schedules backup jobs using standard SQL Server Agent jobs, so you can check backup status in the agent job list.

Litespeed backs up each server, and then deletes the previous day’s backups. Unfortunately, that won’t work with our large servers:

* CXDMSQLPR1
* CXDWPROD1
* CXSQLDV1
* CXSQLQA1

These servers have a separate SQL Agent job called “Database Maintenance – Delete Old Backups – Core DBA.” That calls Litespeed’s extended stored proc xp\_slssqlmaint to delete the old backup files in each server’s backup folder. If the backup jobs on these servers are ever changed to point to a new backup location, then these jobs also have to be updated.

## Restoring Databases

### With Litespeed

Remote desktop into CXSQLDV1 (a SQL utility machine, not an actual database server) and click Start, Programs, Imceda, LiteSpeed, Enterprise Console.

If Litespeed asks for the central repository name, choose CXSQLPRVS1. If it shows a “Welcome to LiteSpeed” introductory window, click Close. Click Connections, Registered Servers to see the list of servers, and add any server that you need to work with.

Drill into the server where you want to restore, then right-click and click Restore Database Wizard.

Pay attention to the file names dialogue box because you may be restoring the data files to other drive locations if you’re restoring onto another server.

Click Restore, and you can watch the status as each job goes through.

### Without Litespeed

Servers without Litespeed use normal SQL Server backup/restore methods. It’s a pain to restore the transaction logs, though.

# High Availability & Disaster Recovery

## Clustering

### CXSQLPRVS1 – Production SQL 2008

This cluster serves our web site’s sales systems and other mission-critical OLTP databases. It consists of CXSQLPR1 and CXSQLPR2.

If the cluster fails over, database mirroring may go offline saying that the principal can’t reach the secondary (remote) server, CXSQLDR1. Reboot the remote server and the principal will be able to connect again.

As of 3/2010, CXSQLPR1 has some kind of hardware problem that’s extremely difficult to diagnose. When it’s the primary node, SalesApp experiences slow inserts to specific tables, and SQL will sometimes stop responding under very heavy load (like two simultaneous backups).

As a result, CXSQLPR1 should only be used as a standby node. If CXSQLPR2 fails, work on getting that node back up as fast as possible to act as the primary again.

## Database Mirroring

We use SQL Server’s built-in Database Mirroring to keep a live copy in Memphis of some critical production databases.

We use Asynchronous (High-Performance) mirroring with manual failover. That means when we do a failover from the primary server to the mirrored server, we have to manually make sure that all transactions on the primary server finish before we fail over.

We don’t use Synchronous (High-Safety) mirroring because our WAN connection between Memphis and New York isn’t fast enough. Before deciding to change to Synchronous, read through the Database Mirroring Best Practices, which suggest that SQL transaction response times in Synchronous mode degrade pretty badly over 20 milliseconds:

<http://www.microsoft.com/technet/prodtechnol/sql/2005/technologies/dbm_best_pract.mspx>

In the examples below, I’m going to focus on DeliveryApp. SalesApp is basically the same, but because it relies on multiple databases instead of just one, it’s more complicated to role swap. All of the databases must be manually managed to make sure they stay in sync during the role swap – we can’t have some databases live in Memphis and others in New York, because the applications will fail.

### Planned Failovers

This can be tested on the DeliveryApp servers using the MirrorTraining database. That database is mirrored between New York & Memphis just like the DeliveryApp database (DeliveryDB) and can be failed over back & forth between sites at any time. No change control is required.

To fail over from one server to another:

1. **Make sure all of the users are out of DeliveryApp.** When testing this with the MirrorTraining database, you can skip this step – it doesn’t affect DeliveryApp at all.
2. **Connect to the principal server with SSMS.** Open SQL Server Management Studio (on your machine or remote desktop into a SQL Server – but never production) and connect to the server currently acting as the principal (primary) server. Drill into Databases, and your screen should look like the below. Notice that the databases say “Principal, Synchronized.” If they say anything else, then you connected to the mirrored server by mistake. Connect to the principal one and pray your luck gets better, because it’s early and you’ve already goofed this up.
3. **Change the database to Synchronous mirroring.** Right-click on the database you want to fail over and click Tasks, Mirror. Change the Operating Mode from high performance to high safety. This forces it to get caught up, and makes sure that any transactions are committed at both servers simultaneously. Click OK.
4. **Fail over to the mirror.** Go back into Tasks, Mirror and click the Failover button. You will get a warning that basically means if any DeliveryApp users are active, they’re going to lose their lunch. Now is your chance to double-check Citrix, and if everybody’s out, click Yes.
5. **Verify that the database failed over.** Right-click anywhere in the right side window (Databases) and click Refresh. The database should now say “Mirror, Synchronized / Restoring . . . )” If it says anything else, the failover did not succeed. Contact a database administrator.
6. **Disable the complete backup jobs on the old primary server.** Drill into SQL Server Agent, Jobs, and right-click on the “Complete Backup Job for DB Maintenance Plan ‘Core DBA – User DBs - Mirrored’” and click Disable. There may also be a job for Core DBA – System DBs, but do not disable that job. Do the same for any other jobs like Transaction Log or Optimizations, as long as they end with “Core DBA – User DBs - Mirrored”. There may also be jobs for “Core DBA – System DBs” or “Core DBA – User DBs – NonMirrored”, but do not disable those jobs.
7. **Enable those same jobs on the new principal server**. Connect to the new principal server with SQL Server Management Studio by clicking Connect, Database Engine and type in the server name. Drill into SQL Server Agent, Jobs, and then right-click on each of the “Core DBA – User DBs - Mirrored” jobs and click Enable.
8. **Take the new principal out of synchronous mode**. On the new principal, right-click on the database name and click Tasks, Mirror. Change the operating mode from High Safety to High Performance, and click OK.
9. **Reconfigure the applications and start them again.** Change the database connection string in DeliveryApp to point to the new principal server. This can be done ahead of time by setting up both servers in the connection string, with CXSQLDR1 as the mirror.

To fail back, just repeat the same process after connecting to the new principal server.

### Unplanned Failovers

If the principal server has suffered catastrophic failure and must be brought online, you can force the mirror server online for service.

1. **Connect to the mirror server with SSMS.** Open SQL Server Management Studio (on your machine or remote desktop into a SQL Server – but never production) and connect to the server currently acting as the mirror (backup) server.
2. **Force the mirror database into service.** Click the New Query button at the top left, which will open a query window on the right side. Copy/paste this into the query window and click the Execute button:  
     
   ALTER DATABASE DELIVERYDB SET PARTNER FORCE\_SERVICE\_ALLOW\_DATA\_LOSS  
     
   That command will force the DeliveryDB database into principal mode. If any errors come up, consult with a DBA. That command will only work when the principal server is completely down. If it’s powered on, especially if SQL Server is running, SQL Server will not allow this command to work. If you really want to force it but the principal SQL Server is still up and running, you’ll have to power down the principal SQL Server, then run the command on the mirror.
3. **Verify that the database failed over.** Right-click anywhere in the right side window (Databases) and click Refresh. The database should now say “Principal, Synchronized)” If it says anything else, the failover did not succeed. Contact a database administrator.
4. **Enable the complete backup jobs.** Drill into SQL Server Agent, Jobs, and right-click on the “Complete Backup Job for DB Maintenance Plan ‘Core DBA – User DBs - Mirrored’” and click Enable. There may also be a job for Core DBA – System DBs, but do not disable that job. Do the same for any other jobs like Transaction Log or Optimizations, as long as they end with “Core DBA – User DBs - Mirrored”. There may also be jobs for “Core DBA – System DBs” or “Core DBA – User DBs – NonMirrored”, but do not disable those jobs.
5. **Reconfigure the applications and start them again.** Change the database connection string in Roadnet to point to the new principal server. This can be done ahead of time by setting up both servers in the connection string, with CXSQLDR1 as the mirror.

If the old principal server’s outage lasts too long, the new principal’s log files may fill up, causing SQL Server to crash. Within a few hours of the outage, the team needs to make a decision about whether to break mirroring. There’s no hard and fast rule – the DBA should be involved to gauge the rate at which the log files are filling up.

To break mirroring, right-click on the database name and click Tasks, Mirroring and click the Remove Mirroring button. After the old principal server is repaired, database mirroring will need to be re-established. This should only be done by a database administrator, but the short version is:

* Disable all backup jobs for this database on the principal server
* Do one complete backup on the principal
* Do one transaction log backup on the principal
* Restore both the complete and transaction log on the mirror server, but leave the database in recovery mode
* Enable database mirroring on the principal
* Alter the database partner timeout to 90 seconds instead of 10 (because our WAN link is slow and we get constant false alarms)
* Fail back over to New York if necessary
* Enable/disable backup jobs on the appropriate servers

# Change Control

## Deployment Practices

For a stable production environment, we follow this deployment practice:

* Developers script every change on the development servers first
* The manager tests the script on the QA box and signs off that it works
* The manager puts in a change control request
* The DBA either watches that test take place, or tests it again on QA
* The DBA applies the script on the production boxes

### Developers Script Every Change on Development First

Developers can make any changes they want on the development servers – they can modify tables with Management Studio, write any alter queries they want, and there are no controls on that.

However, in order to promote these changes into production, they have to be scripted out into text files.

### Managers Test Scripts on the QA Box

The product manager tests the change script on a fresh restore on QA. The restore must be absolutely fresh, restored straight from the production server, and applied once and only once.

If any errors come up during the script, the script has to go back to the developer and another restore must be done. We do not test a script, modify it, and then run it again because running a script twice can have other side effects.

If there’s only one developer involved (without a technical manager), the developer can test their own scripts on the QA box.

### Manager Puts In a Change Control Request

We can put in a tentative change control request to the Change Control Board meeting, but be considerate – don’t put in a change control request if the change script hasn’t been at least tested once on QA.

The preferred company-wide maintenance window is Saturday at 10pm.

At the time of the change, the responsible developer should plan to be available and have internet access to fix any problems that may arise. The manager should also be available via phone.

### DBA Signs Off on the QA Test

The DBA either has to watch the manager running the test, or run the test independently after the manager turns in the change.

### DBA Applies the Script in Production

The DBA calls both the developer and the manager to verify that they’re available and ready for the change. If both are available, the DBA applies the change script, and then contacts both staff members again to let them know that the change has been processed.

If the change is significant, the DBA will run a database backup or disconnect the mirrored pair (to keep the backup mirror out of sync) prior to applying the change. That way we’ll be able to go back to the prior version easier.

# Maintenance Windows & Patches

## QA & Development Servers

These servers can be taken down for maintenance every Saturday evening from 10pm until Sunday 2am. No load testing should be scheduled for this maintenance window.

Patches & updates for Windows, SQL Server, and backup software will be applied to these boxes as soon as the patches come out. The patch effects will be watched for a period of one to two weeks, and then the patches will be applied to the production servers.

## Mission-Critical & Production Servers

All reboots and outages must be scheduled through our Change Control process.

# When to Get Help from the DBA

### When a Query Takes More Than 30 Seconds

If any query regularly takes over 30 seconds to run, call the DBA. Within a matter of minutes, we can tell if there’s a quick fix on the database side to make it run faster. It really is that quick for us – most of the time, we can either make a really quick improvement, or we can pinpoint the problem and make a note of it for long-term improvements.

### When Buying (or Choosing) Software

Get us involved from the beginning, and we can help you budget how much storage and database processing power the app will need.

We can also help by identifying other people in the company who have a similar need, or who are using similar pieces of software that you might be able to reuse.

### When Tables Have Over Ten Million Rows

If you’re working with large sets of data, or if you’re designing a program that you anticipate will have over 10 million rows in it, let us know. We can help you design it to scale.

With SQL Server 2005/2008, we can also partition the table on the database side, invisible to your application, so that SQL is able to respond faster to insert/update/delete transactions.

# SQL Coding Standards

## Design for Database Mirroring

SQL 2005 includes a powerful new feature, Database Mirroring, that allows us to keep a realtime database copies in multiple locations. We can fail over from one server to another across geographic locations, like from New York to Memphis, without advanced notice or human intervention.

To take advantage of this feature, your applications must be compiled with the .NET Framework 2.0 or higher, or use ODBC. (SQL Server 2005 ODBC drivers include mirroring in the connection string.)

## Table Design

### All tables should have a primary key.

In any table, we should be able to identify a set of fields that make each row unique. They may not even be a natural set of keys – they might be a surrogate key field like an autonumbered identity field – but there does have to be a unique thing on each row.

Ideally, it should not be a GUID (uniqueidentifier) field. Those fields are rather large, and they take a lot more to store. This has negative impacts on index speed. Before using a GUID as your unique key, check with your database administrator to see what other options are available.

### All objects should be owned by DBO.

During development, explicitly create all tables, views, stored procedures, etc with a “dbo” prefix so that their name is dbo.whatever.

All objects should be owned by DBO because non-standard owners (like MyApplicationUser) have to have their permissions set differently, and they are viewed differently by different logins.

SQL Server will allow you to create multiple objects with the same name as long as they’re owned by different people. For example, we could end up with dbo.Customers, and Brent.Customers. Consider the following query:

SELECT \* FROM Customers

When Brent logs in, he will be querying the table Brent.Customers. When any other user logs in, they’ll query the table dbo.Customers. The two tables will have different data, and that will quickly become a troubleshooting nightmare.

## Stored Procedures & Functions

### All functions and stored procedures should include a description block.

Use the below as an example:

CREATE PROCEDURE dbo.USP\_MyStoredProc AS

/\*

Summary: Updates user history records to reflect recent sales.

Contact: Brent Ozar

Description: Called by the SalesApp Creative Statistics Generator once per hour.

Queries the OrderHistory tables and updates customer statistics.

ChangeLog:

Date Coder Description

2005-06-28 Brent Ozar Created.

2005-11-14 Brent Ozar Now uses the partitioned OrderHistory tables.

2006-03-28 Bill Gates Redesigned to use the new getOrderHistory function.

\*/

A programmer or DBA should be able to open the stored procedure and see right away what it does, who calls it, and who’s been changing it lately.

Notice that the description block is BELOW the “create procedure” statement. That way, the code stays intact inside the stored proc.

### All data definition statements should go first.

Create temp tables and declare table variables first, before anything else is done in the stored procedure or function. Otherwise, the SQL Server engine will recompile the stored proc multiple times – once at the beginning of the stored proc, and then again during execution when it encounters the create-table command.

# SQL Server Setup Standards

## SQL Server 2005 Setup Checklist

1. Create AD accounts for SQL Server and SQL Agent.
2. If SAN-connected:
   1. Upgrade HBA firmware.
   2. Set up multipathing drivers (RDAC) on HBAs.
   3. Set up SAN fabric zoning.
   4. Test SAN access by pulling one HBA cable, testing drives, and repeating the process with the other HBA cable.
3. If iSCSI-connected:
   1. Set up teaming software on NICs.
   2. Test iSCSI access by pulling one network cable, testing drives, and repeating the process with the other network cable.
4. Use DISKPART to create aligned partitions per KB article #929491:  
   <http://support.microsoft.com/default.aspx?scid=kb;en-us;929491&sd=rss&spid=3198>
5. Install the SMS client.
6. Install SNMP and WMI. Configure SNMP for a read-only community named snmpn0w, send traps to flmirwhatsup1.companyx.com, and accept SNMP packets from any host. (This is for WhatsUp.)
7. For 32-bit servers, enable the /PAE switch in boot.ini.
8. Set the page file on the C drive to 1.5x physical memory.
9. Go into Local Security Settings, Local Policies, User Rights Assignment.
   1. In the permission “Perform Volume Maintenance Tasks”, add the domain group “SQL Service – Service Admins”. This enables SQL 2005’s fast file initialization, so multi-gig file increases can be done instantly.
   2. Set up the “Lock pages in memory” permission, which enables better memory management.
10. Install SQL Server 2005.
    1. The services should start under the domain accounts set up earlier. This lets us back up to network shares.
    2. Do not install Reporting Services or Notification Services on production servers – we’ll implement separate servers for those if necessary.
11. If doing a cluster setup, install the client tools on the second server.
12. If the server uses iSCSI, set up the cluster & SQL services as dependent on iSCSI. Do this at the command prompt:  
    sc config clussvc depend= clusnet/netman/Rpcss/w32time/msiscsi  
    sc config mssqlserver depend= msiscsi
13. Install SQL Service Pack 2 and latest cumulative hotfixes.
14. Move the TempDB to a non-system drive.
    1. Configure one TempDB file per core. The data file creation should take a few seconds or less – if it takes longer, then instant file initialization isn’t configured correctly.
15. Configure SQL Server to use the appropriate amount of memory.
16. Run the SQL Server 2005 Surface Area Configuration for Services & Connections wizard and enable remote connections over TCP/IP.
17. Run the SQL Server 2005 Surface Area Configuration for Features wizard and enable database mail.
18. Go into Management Studio and configure Database Mail for mail.companyx.com.
19. After mail setup, right-click on SQL Agent, configure the alerting system, and restart SQL Agent.
20. Go into SQL Server Agent and configure operators for each sysadmin.
21. Create alerts for each severity level 16 through 25, and send responses to the operators.
22. Set the default SQL data & log file paths.
23. Run the SQL Server 2005 Performance Dashboard Reports setup.sql script.
24. Install Quest Litespeed under the service account CXDOMAIN\svcQuestLitespeed and point the central repository at CXSQLPRVS1.
    1. Set the default backup directory to \\cxbackups\sql\(servername).
25. Set up backups:
    1. For production servers - using Lightspeed, set up complete backups daily and transaction log backups every 15 minutes, both to the designated network backup share.
    2. For QA servers – full backups once a week. We rarely have to restore these, and we actually tell the users these aren’t backed up at all. We don’t really have the backup window to do fulls on these since they’re the same size as production.
    3. For development servers – full backups daily.
26. Configure WhatsUp to monitor the server including performance monitors.
27. Set up database mirroring.
    1. If the principal server is a cluster, alter the database timeout so that the database doesn’t fail over when the cluster fails over. This should also be done if the mirroring is being transferred over a slow WAN link, which can time out quickly.  
       ALTER DATABASE (mydatabasename) SET PARTNER TIMEOUT 90