IBM Lotus Domino Server and Application Performance in the Real World

Andrew Pollack, President
Northern Collaborative Technologies

andrewp@thenorth.com
http://www.thenorth.com

Who Am I?

Administrator & Developer since version 2.0

IBM Lotus Beacon Award Winner

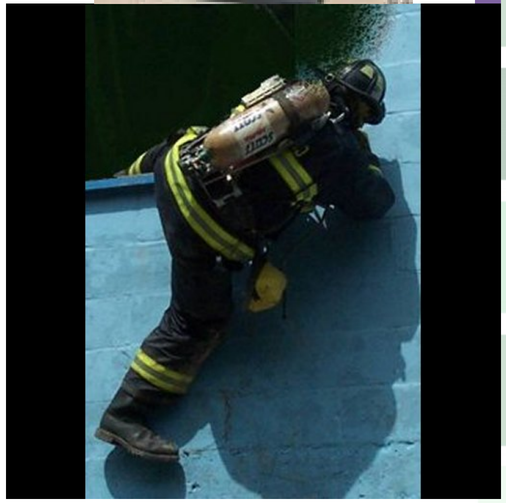
Services

- Site Performance Reviews
- Legal Case Consulting
- Application Development
- Administrative Overhaul
- Security Review & Penetration Testing

Products

- NCT Search
- NCT Compliance Search
- NCT Simple Sign On

Structural Firefighter



Key Focus Points

- Performance with a Big Picture approach
- Defining Performance In User Terms
- Key Performance Choke Points
- General Considerations
- Common General Tweaks
- Make Your Web Site Faster!
- Developers, Developers, Developers
- Servers and SANS and VMs – oh My!
- Virtually Perfection
- When good INI settings go Bad!
- Finding Your Own Choke Points
- Summary

Can Prevent Performance Problems

PERFORMANCE WITH A BIG PICTURE APPROACH

Big Picture : There Is No Magic

- No Single INI Variable -- #1 Server Fix
- Focus On The Basics!
- No Super Storage Network
- No Ultimate Network Switch
- No Omnipotent Third Party Application
- No Über-Consultant
  - Not Even Me!

Big Picture: Small Issues Stack Up

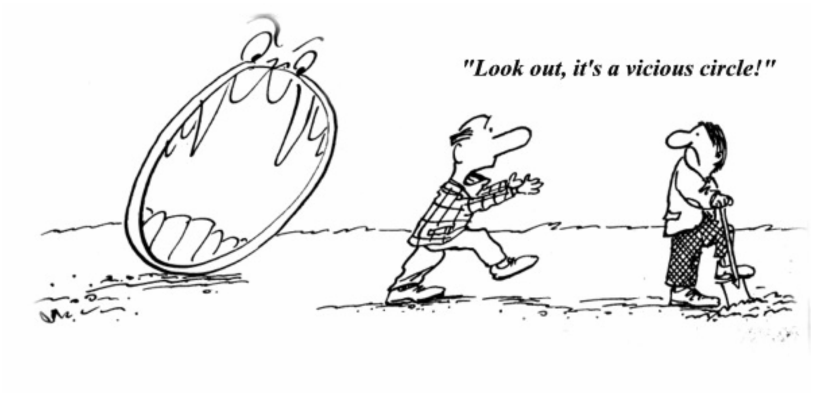
Performance Problems Are like snowflakes

- Individually, they don't matter much at all
- You notice them only once they stack up

For example:

- Poorly Performing Disk I/O
- + Agents Changing Many Documents
- + Many Views (or BAD views) to Update
- == Very Slow System

These kinds of problems create a feedback loop, which amplifies the problems



Beware of INI Changes

My Number One Server Crash Response

INI Changes Come From

- Well meaning tips
- Low level tech-support
- Too much time on public forums

How to fix most server crashes

- Clean out ALL non-default INI settings
  - Unless you can specifically document why it's critical
- Clean out ALL non-shipping Code
  - Get rid of those fix-packs that didn't fix the problem

Yes, there are some good changes to make to the INI file

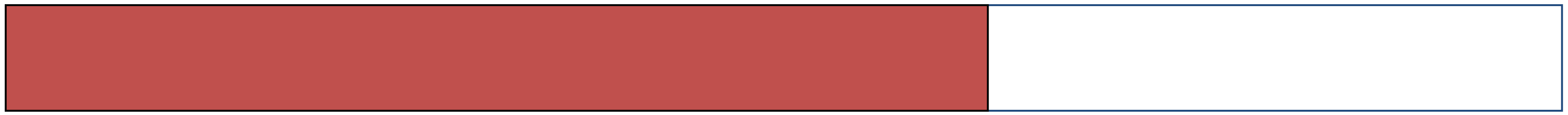
It’s not how you feel, its how you look.
Darling, you look marvelous!   -- Billy Crystal

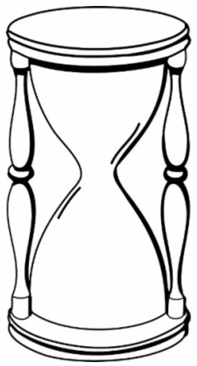
DEFINING PERFORMANCE IN USER TERMS

Performance in User Terms

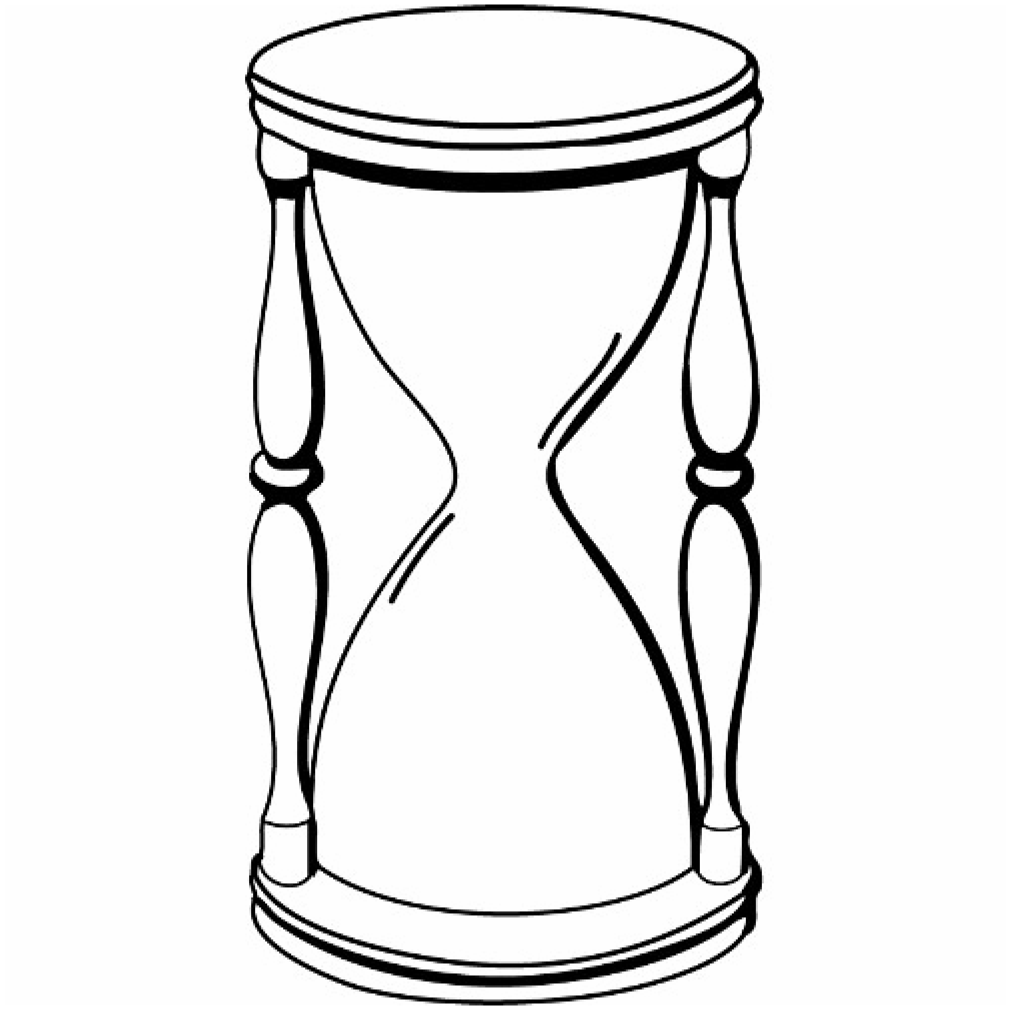
If the user must wait for something, it will always seem slow – no matter how fast you make it.

Nothing is worse than an hourglass cursor and a bar slowly moving across the screen





…Except NOT having the bar



Performance in User Terms: Tips

Move anything not immediately required by the user to a background process

- Batch process updates of data that users do not need instantly

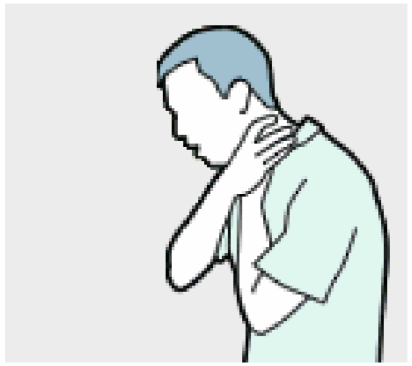
Cache Commonly Referenced Data

- How often do your common lookups change?
  - Country Names?
  - Escalation Levels?
  - Document Categories?
- Lookup once when the database is opened, and store the values as environment variables locally

Don’t pop-up modal dialog boxes with no choices!

We're going the wrong way, but we're making excellent time!

KEY PERFORMANCE CHOKE POINTS



Choke Points: The Network

Bandwidth vs. Latency

- Bandwidth
  - How big around is the pipe?
- Latency
  - How long is the pipe from end to end?

- Even light takes several minutes to reach us from the Sun.

- Latency impacts “Chatty” connections
  - Notes Database Open
  - Multiple View Lookups
  - AJAX on Web Applications

Where does Latency Come from?

Ping times larger than 100ms are “high” latency.

WAN links, Satellite links, Modems, and VPN’s are all prone to latency issues

Multi-Hop connections across buffered routers and firewalls can introduce latency

Encryption software can introduce latency

Dealing with High Latency

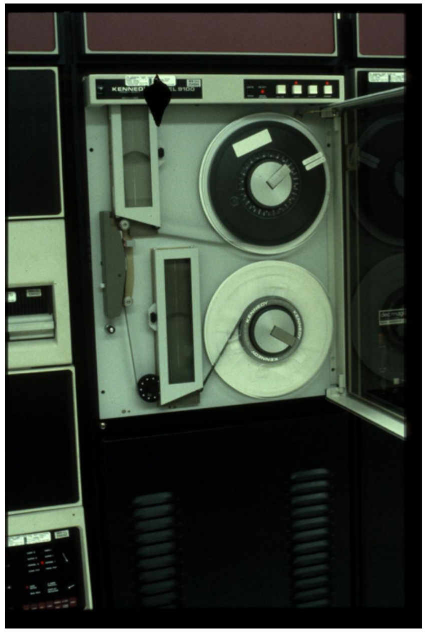
- Avoid opening and closing many documents
- Avoid DB Lookups by caching common values
  - Example: Use a db open script to write common lookup values to a local environment variable each time the user opens the database
- Use “RunOnServer” to move complex agent work to the server, the read the result from a profile document
- Consider JSON embedded on the document instead of AJAX lookups
- Stop using “NoCache” on your DBLookups

Choke Points: Disk I/O

This is the #1, #2, and #3 Root performance problem on Domino Server

Nearly any other performance problem is made many times worse if the Disk I/O is overwhelmed

Most Domino Servers are not well optimized for Disk I/O



Common Sources of Disk Performance Problems

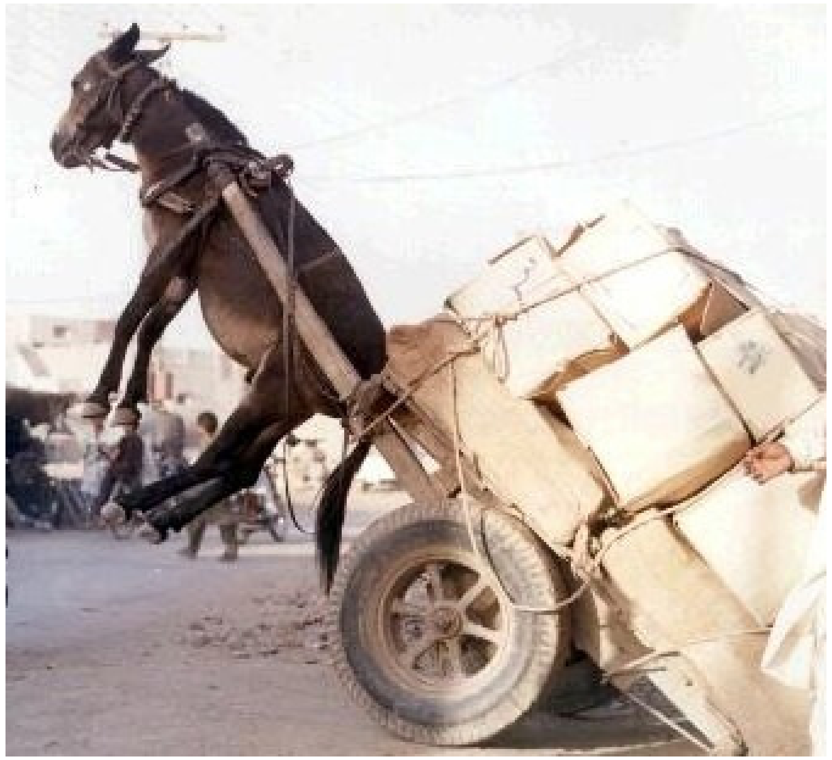
Failure to use DAOS!

One “Data” drive is used for too much
- databases, index rebuilds, temporary files, swap files, and even transaction logging

Poor SAN configuration for Domino volumes

Too heavy a reliance on Storage Area Networks

Poor choice of RAID configurations



For FSM Sake, Start Using DAOS!

- DAOS is safe.
- It will likely save you 50% or more of your storage space
  - Really - It is safe.
- That means 50% or more savings in DISK I/O as well
  - It's not like "Shared Mail" – I promise
- It also means 50% less space on every backup
  - And it's safe, too!
- In the newest server versions it will also save network traffic

Whenever Possible Use Multiple Drives

- Put your transaction logging files on a separate drive
- Move your view indexing temporary files to another drive
- Consider moving disk-intensive applications to their own drive
- If you must have memory swapping, give it its own drive
- Active Log Files for Web Servers, SMTP, etc. can also be offloaded to their own drives

Not Everything Needs Its own Drive

Things that load once and are not re-accessed frequently do not need to be on high performance resources

- The Operating System
- Application Program Files
- Archived Log Files

Unique Drive letters may not be different drives

In Virtualized environments and on a SAN, multiple virtual drives on the same physical volume do not help

One Disk may have multiple partitions

- Different partitions are NOT different spindles
  - All the partitions on the same drive, share the same read-write head and are impacted by data access as a single entity.

Multiple drives in a RAID array don’t count

- A RAID array is treated by the system as a single drive. By definition, data is written across the whole array

The “Best-Case” is multiple drives on different drive controllers

Too heavy a reliance on Storage Area Networks

The SAN is not Evil – But it isn't perfect either

- High Speed, but High Latency

A SAN is a Compromise

- Trades the speed and simplicity of local drives for enterprise manageability and flexibility

Good for Backup Data

Good for Big, Sequential Files

- Media Files
- Installation Kits
- Archival Data

Choke point for active database work

Domino with a SAN

Consider the benefits of a SAN

- Highly redundant storage
- Single backup point
- Consolidated free space
- Performance?
  - I have yet to see a SAN that truly outperforms local high speed disks

Not all Domino Data needs these features

- Transaction Logs – Consider local RAID if possible
- Indexing Scratch Space – Use Cheap, Local, Fast Drives

If you're already clustering Domino, only one of the clustered machines may need to be on the SAN

If you do use a SAN

- Work with the SAN team to configure your volumes
  - Dedicated LUN & Disks for each of these if possible:
    - Domino Data
    - Transaction Logs
    - Temp Space for view index rebuilding
      - Operating system “TEMP” variable
      - Notes.INI “View_Rebuild_Dir=“
  - Tell the SAN team to treat it like a relational database
    - Highly Read Intensive
- Where can you compromise?
  - Cheap local drives for low-risk use
    - Memory Swap File
    - Temporary Scratch Space for View Rebuilds
    - Web Server Cache Files
    - Log Files

Virtualization and Domino

Domino runs just fine in VMWARE

- Some of my best friends are virtual servers
- All my production & development servers are in VMs

Performance issues are VERY similar to SANs

- Disk I/O is again critical to Domino performance
- Virtual environments often share disk resources
- Virtual environments often utilize SANs

Follow the guidelines for using Domino on a SAN

- Local, dedicated storage spindles wherever possible
- Dedicated LUN & Disks wherever you can

Poor choice of RAID configurations

RAID is not ALWAYS the best performance choice

Some Common Types of RAID

- RAID 0
  - Increase performance, Decrease Reliability (x number of drives)
- RAID 1
  - Increase Reliability, No Performance Difference
- RAID 5 (The Most Common) – Uses 3 or more drives
  - Balance of redundancy plus some performance gain
- RAID 1+0 (aka RAID 10)
  - Two pairs of RAID1 read as a RAID0 (Hybrid)
- RAID 0+1
  - Two pairs of RAID0 written as RAID1 (Hybrid)

Why RAID isn't ALWAYS the best way?

Competition for Resources

- An overall performance gain with RAID 5 of 30% (typically) is spread across all the disk I/O on the server.

You are smarter than RAID

- You can put highly intensive resources on specific drives or arrays, balancing the load more effectively

Multiple RAID Arrays are not always possible

- Expensive
- Multiple Drive Bays
- Power Hungry Drives

Disk I/O: Rethinking RAID

Most RAID arrays are configured to improve redundancy, not necessarily speed.

Not all data requires redundancy

- Loss of some data is very low risk
  - Memory Swap Files
  - Indexing scratch space
  - Temporary files
  - Cache files

Inexpensive SATA drives can be used for a real performance gain

Solid State Drives – Very fast but often not idea

Solid State Drives (SSD)

Also known as “FLASH” drives

Getting more common on Laptops, Netbooks

Reliability Issues are Largely Resolved

VERY Fast READ Times

Write Performance Quickly Degrades

- This is changing quickly, but still the case for most uses
- Windows 7 & Windows Server 2008 R2 Support “TRIM”
    - http://en.wikipedia.org/wiki/TRIM

Good for Program Files, Java Libraries

Bad for NSF Databases, Indexing, Translogs

Choke Points: System Resources

These should be obvious

- More RAM is better – Up to what is supported
  - Depending on the OS, you may need to partition your server to take full advantage

- Drive Cache – If your OS lets you manage it, you should work to really optimize this

Most Anti-Virus Software is EVIL when it runs against Domino Databases

- Make sure your AV is Domino aware!
- Do you really need AV software running on a Domino Server
  - Hint: No, you usually don't

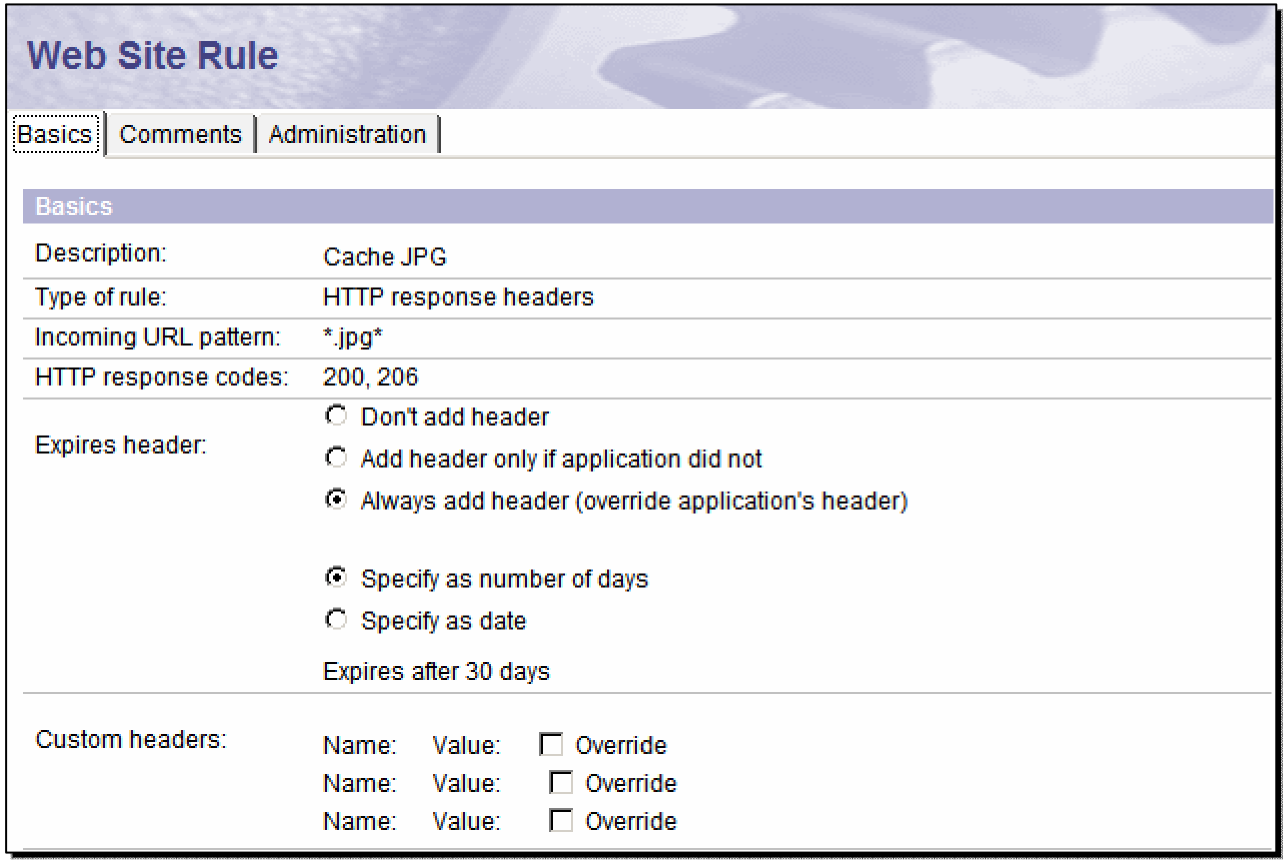
Faster faster! The lights are turning red…

MAKE YOUR WEBSITE FASTER!

Let the browser cache common items

Resources that don’t change frequently can be cached

- JPG
- PNG
- GIF
- MOV
- MP3
- MSI
- MPG
- ZIP
- EXE



Web Site Rule

Basics | Comments | Administration

Basics

Description:	Cache JPG
Type of rule:	HTTP response headers
Incoming URL pattern:	*.jpg*
HTTP response codes:	200, 206
Expires header:	○ Don't add header ○ Add header only if application did not [✓ Always add header (override application's header)] [✓ Specify as number of days] ○ Specify as date Expires after 30 days
Custom headers:	Name: Value: ☐ Override Name: Value: ☐ Override Name: Value: ☐ Override

Developers really LOVE when administrators give them feedback

APPLICATION DESIGN STRATEGIES

Choke Points: Views

For application performance tuning, views are the first, second, and third place to look

View indexing is very disk intensive – and can amplify disk I/O shortcommings

To update a view, a full database scan often needs to happen. That can be very very slow on large databases

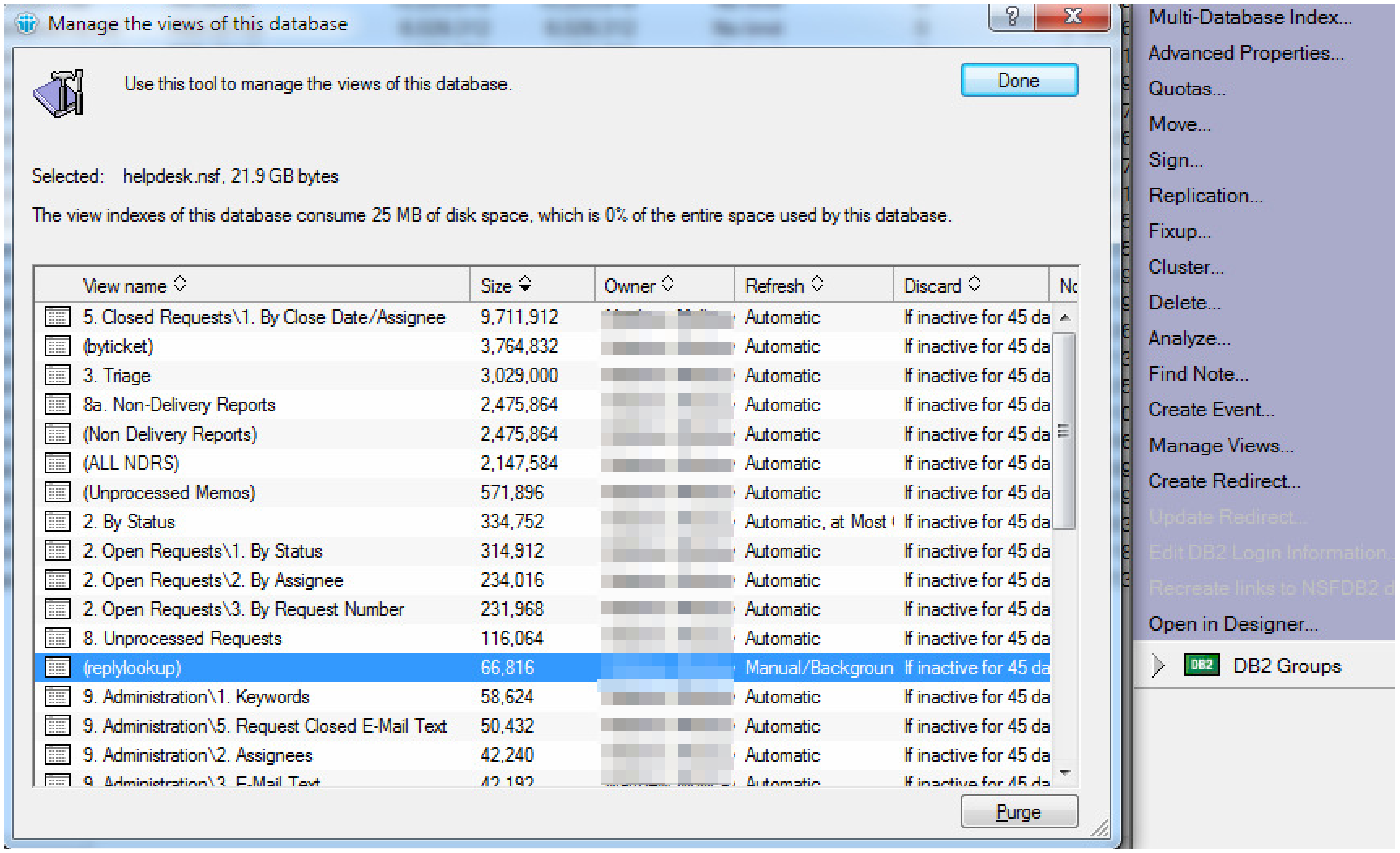
Any view performance problem grows exponentially with the volume of data

- These problems are often not caught in test

What Kills View Performance

WHEN GOOD VIEWS GO BAD

Use the “Manage Views” Admin Client Feature



Manage the views of this database

Use this tool to manage the views of this database.

Selected: helpdesk.nsf, 21.9 GB bytes

The view indexes of this database consume 25 MB of disk space, which is 0% of the entire space used by this database.

View name	Size	Owner	Refresh	Discard	Nc
5. Closed Requests\1. By Close Date/Assignee	9,711,912	[redacted]	Automatic	If inactive for 45 da	
(byticket)	3,764,832	[redacted]	Automatic	If inactive for 45 da	
3. Triage	3,029,000	[redacted]	Automatic	If inactive for 45 da	
8a. Non-Delivery Reports	2,475,864	[redacted]	Automatic	If inactive for 45 da	
(Non Delivery Reports)	2,475,864	[redacted]	Automatic	If inactive for 45 da	
(ALL NDRS)	2,147,584	[redacted]	Automatic	If inactive for 45 da	
(Unprocessed Memos)	571,896	[redacted]	Automatic	If inactive for 45 da	
2. By Status	334,752	[redacted]	Automatic, at Most	If inactive for 45 da	
2. Open Requests\1. By Status	314,912	[redacted]	Automatic	If inactive for 45 da	
2. Open Requests\2. By Assignee	234,016	[redacted]	Automatic	If inactive for 45 da	
2. Open Requests\3. By Request Number	231,968	[redacted]	Automatic	If inactive for 45 da	
8. Unprocessed Requests	116,064	[redacted]	Automatic	If inactive for 45 da	
(replylookup)	66,816	[redacted]	Manual/Backgroun	If inactive for 45 da	
9. Administration\1. Keywords	58,624	[redacted]	Automatic	If inactive for 45 da	
9. Administration\5. Request Closed E-Mail Text	50,432	[redacted]	Automatic	If inactive for 45 da	
9. Administration\2. Assignees	42,240	[redacted]	Automatic	If inactive for 45 da	
9. Administration\3. E-Mail Text	42,192	[redacted]	Automatic	If inactive for 45 da	

Bad View Design: Too Much Data

Switch @Responeses to @AllDescendants

- NO visibible difference to users
- Can reduce view sizes drastically

Can You Set a CUTOFF date?

- Form = “Request” & @Modified < [01/01/2008]
  - o Hardcode The Date
  - o Change it by AGENT, Warning in DB Script if out of date

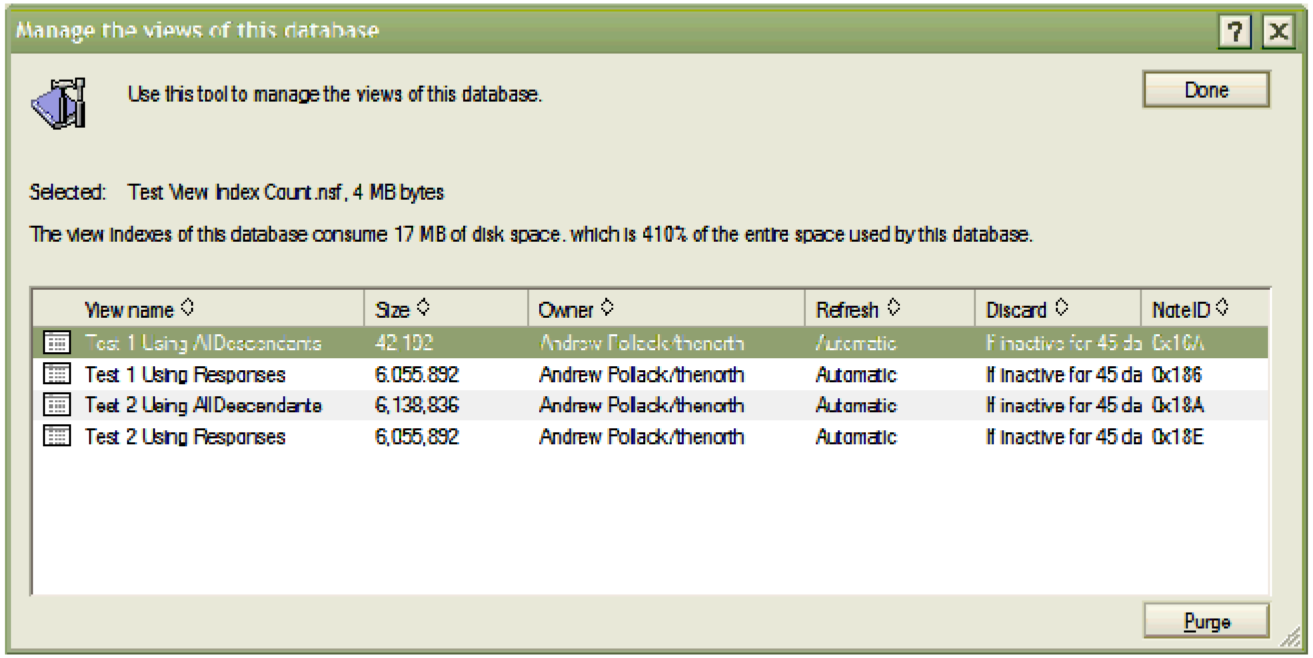
How about a CUTOFF data for MOST of the views, with just one or two for “Archival” data?

Switch @Responeses to @AllDescendants

NO visible difference to users

Can reduce view sizes drastically

- View #2 is 153 Times the Size of #1 and has the EXACT same content



Manage the views of this database

Use this tool to manage the views of this database.

Selected: Test View Index Count.nsf, 4 MB bytes

The view indexes of this database consume 17 MB of disk space, which is 410% of the entire space used by this database.

View name	Size	Owner	Refresh	Discard	NoteID
Test 1 Using AllDescendants	42,192	Andrew Pollack/thenorth	Automatic	If inactive for 45 da	0x10A
Test 1 Using Responses	6,055,892	Andrew Pollack/thenorth	Automatic	If inactive for 45 da	0x186
Test 2 Using AllDescendants	6,138,836	Andrew Pollack/thenorth	Automatic	If inactive for 45 da	0x18A
Test 2 Using Responses	6,055,892	Andrew Pollack/thenorth	Automatic	If inactive for 45 da	0x18E

Bad View Design: Too Much Sorting

EACH Sorted Column can as much as DOUBLE the size of the total view index

Many views have all the columns sorted

- Even the ones that end up sorted because of order

Multiple Column Click-To-Sort Views Can be WORSE that multiple views!

Many SHARED columns are sorted

- Developers Assume No Downside

Bad View Design: Too Much Data

Does the SUBJECT really need to be in every view?

Can you create one “Master” view with all the data, and several “Index” views with an Action Button to open the master view?

Bad View Design: Using TIME Values

If your view column (bad) or selection formula (worse) uses @Now, @Today, etc.. You're hurting performance

Time dependant views are "Always" considered out of date and must be re-indexed for every use

If you've got one, you've got more. Developers that do this tend to repeat the pattern

Alternatives to Time Specific Views

Use a FOLDER instead

- Run a agent to select the right documents for the folder on a periodic basis – Daily for “@Today” or Hourly for @Hour(@Now), etc.
  - This will still cause an update, but only once each time the update happens

Use Categories

- Categorize documents based on a stored date value, then use a “show single category” option on the view

If you MUST use a time specific view, set its update frequency to the absolute least frequent you can

- It will still update for each user access, however, unlike a folder which is static

Bad View Design: Highly Complex Formulas

Consider a column formula with 10 steps

Now consider 100,000 Documents in it

That column must execute 1 Million steps for each view index rebuild – just in that column

Many column formulas are much more complex, and serve many times that many documents

Alternatives to Complex view formulas

Create Hidden Fields on the Document

At “Save” time, compute the value that would be on the view column in the hidden fields

Display the value of the hidden field as the view column formula.

What was a complex formula executing hundreds of thousands of times is now a single field value

Bad View Design: Too Many Views

Consider a database with 100,000 documents

Consider that database having 10 views

Consider each view having 5 columns

Each time data in the database is updated, every selection formula has to be checked to see if the view is impacted

Every view has to be updated by the indexer

Alternative to Using Many Views

Embed The View on a Form or Frameset

- Categorize the view in the same way you would distinguish the different views
- Use Show Single Category to differentiate the data to the user
- Compute text values on the form to result in very different data in each category if needed

Use multi-column hidden views so that the same view can serve multiple lookup needs

- Make sure your developers coordinate so that duplicate lookup views are not created

Full Text Search: The Good, The Bad, and the Ugly

The Good

- You can use it in agents instead of db.search
- Db.ftsearch() has a rich syntax and can be much faster
- Its lets users find things – of course

The Bad

- Usually set to update “immedately”
- Agents that change many documents can cause a massive amount of disk I/O at the worst possible time

The Ugly

- Be careful using it as a way to gather documents in code, as it may not be up to date

Choke Points: @DBLookup

Why are you using “NoCache”?

- Cache times are very small, does your data really change on a second by second basis?

Can be very chatty – a killer on high latency networks, but not as bad for web apps

Requires more views to be up to date – big performance hit in databases that change a lot

Many lookups on the same form, to the same place for different values?

- Use it once to get the UNID, then use @GetDocValue

Use a profile document, or local environment variables updated in the dbopen script to store commonly looked up data

Agents are better with Hash!

Calculate and Store a HASH value

- A HASH is a short, nearly unique, value crated by applying a mathematical formula to a set of data. For example, you can hash an entire paragraph and get a short string as a hash value.   The same source will always produce the same hash, but any change to the source will produce a different hash.

- You can tell if a document has changed, simply by comparing the HASH value

Pre-Load The Documents You'll Need

- Traditional Code Pattern
  1. Read a value to be updated on many documents
  2. Load the first document, Make Changes, & Save
  3. Load the next document, Make Changes, & Save
     - Repeat until finished

- Once you start saving changes, view lookups will slow down radically and the next document will take longer to find

- Alternative Code Pattern
  1. Read a value to be updated on many documents
  2. Load all the documents to be updated into a collection or list
  3. Update all the documents at once
  4. Save all the updated documents

Other Agent Tricks

Read View Entries – Not Documents

** This is less critical in newer server versions **

Delay Updates until the agent is finished

- NotesDatabase.DelayUpdates=True

Turn off MIME conversion when working with mail documents

- NotesSession.ConvertMime=False

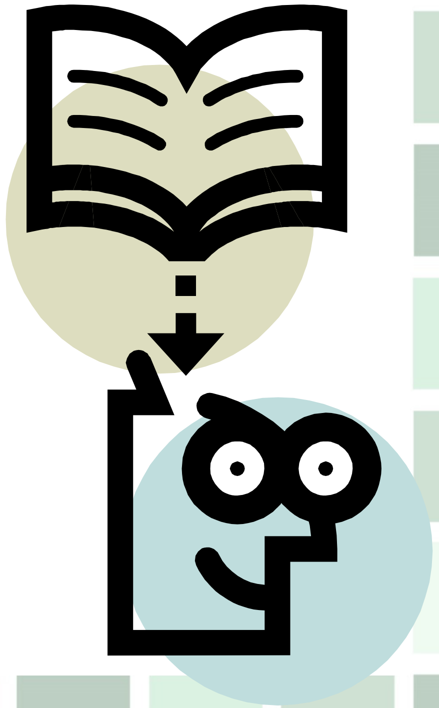
Run agents periodically, not “Before New Mail Arrives” – that slows down the router

Story Time!

Andrew’s Magic Hash story

If you fall asleep, please don’t drool on the table

C’mon, it’s a true story!



Where to place the blame

FINDING YOUR OWN CHOKE POINTS

Where to look for performance problems

Look for Disk Performance First

- Start Simple: Are the drive lights sitting on for long periods of time?
- Use the operating system's tools
  - Performance monitor & diskmon in Windows, "top" in Linux, etc.
  - Processes like "logasio" which is for transaction logging will show up

Check for network latency and bandwidth

- Start Simple: Use Ping to check latency

Domino Configuration Tuner

- Delivered as a database template (DCT.NTF)
  - Available for free – download from IBM
    http://www-01.ibm.com/support/docview.wss?uid=swg24019358

- Evaluates the server and compares to known best practices
- Makes recommendations for changes

- Recommendations are generally good – but not universal
  - Do not follow blindly – Understand the recommendation first
  - Document any changes you make so they can be undone

- If something should “always be set” a certain way, it would be the default.

You really came here looking for cool INI settings like DominoRunFaster=11

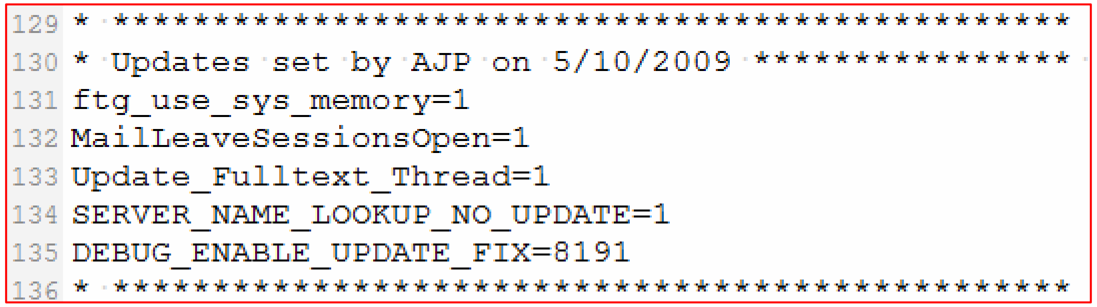
Remember – Many of these have become the default over time. You are usually Better off using the default settings.   Be especially careful of old ini settings after you have upgraded the server to a new version.

Obsolete INI settings HURT performance

INI SETTING TWEAKS

Some NOTES.INI tweaks

COMMENT NOTES.INI Changes!



```
* ********************************************************
* Updates set by AJP on 5/10/2009 *****************
ftg_use_sys_memory=1
MailLeaveSessionsOpen=1
Update_Fulltext_Thread=1
SERVER_NAME_LOOKUP_NO_UPDATE=1
DEBUG_ENABLE_UPDATE_FIX=8191
* ********************************************************
```

Here’s some that I use

- MailLeaveSessionsOpen=1
  - For busy mail servers, can speed up delivery
- Update_Fulltext_Thread=1
  - Move full text indexing to its own thread, distinct from the indexer – This is the closest to “runfaster” I have found
- Ftg_use_sys_memory=1
  - Use memory outside the Domino server
- HttpQueueMethod=2
  - Like having one line for multiple cash registers
  - Default in 8.5.1 and later

A few more notes.ini tweaks

Use These Together:

- SERVER_NAME_LOOKUP_NO_UPDATE=1
  - Tells the server to use the old index while the new one catches up
  - Starting with 8.0 this should be the default
- DEBUG_ENABLE_UPDATE_FIX=8191
  - Fine tunes when the directory indexes get refreshed
  - Starting with 8.0.1 this should be the default

- SERVER_MAX_CONCURRENT_TRANS=-1
  - Default as of 8.03 and 8.0 – used to be 20
- SERVER_POOL_TASKS=100
  - Default as of 7.03 an 8.0 is now 40
  - Used to be 2 times server_max_concurrent_trans
  - 100 is used in IBM Perferformance Testing
  - Set lower if processor is maxed

And of course… NSF_Buffer_Pool_Size_MB

NSF_Buffer_Pool_Size_MB=

- Very powerful, but very complex
- Check the Lotus Notes Knowledge base
- Starts at around 300

Not as critical as it used to be

- Documentation Says it is now set AUTOMATICALLY for non-partitioned Servers

- My Testing Says it is also now set AUTOMATICALLY even for partitioned Servers in 8.5.x

Check your success with this console command

- show stat database.database.b*

- Don't check too soon after a change, its only valid over time

Notes 8 Client Tweak

To make the Eclipse based client load faster

Open this folder:

{NotesProgramDirectory} \framework \rcp \deploy

Prior to 8.5.1 use this folder instead:

{NotesProgramDirectory} \framework \rcp \eclipse \plugins \com.ibm.rcp.j2se.{Version}

- Edit the file: jvm.properties
- Change the line: vmarg.Xmx=-Xmx256m
- So that it reads: vmarg.Xmx=-Xmx512m

Note: You can set it higher, but aim for no more than half of your available RAM

Readers on my blog overwhelmingly report fantastic results with this one

Summary

Repeat After me:

- There is No “RUN_FASTER=1”
- I will clean up my NOTES.INI
- I will COMMENT my NOTES.INI changes

Performance Isn’t Magic, its Planning

Save the Disk I/O, Save the Server

Latency is as critical as Bandwidth

When in doubt, Blame the developer

Questions?

- Ask now, don’t wait for the end and ask quietly at the podium

  - The most up to date copy of this presentation will be on my blog site: http://www.thenorth.com/apblog

  - Andrew Pollack – Northern Collaborative Technologies
    - andrewp@thenorth.com
    - http://www.TheNorth.com