Microsoft® SQL Server™ performance tuning for Microsoft Dynamics™ NAV

‘TechNet Evening’
Introductions

Steven Renders
Microsoft Certified Trainer
Plataan
steven.renders@plataan.be
➤ Check Out: www.plataan.be
Overview

- Introduction
- Dynamics-NAV Architecture
- Differences Between the Server Options
- Recovery Model
- Security & Synchronisation process
- Sql Server Versions & Limitations
- How to conduct a Performance audit
- Hardware Recommendations
- The NDBCS Database Driver
- Using Find Statements
Overview

- Optimizing Dynamics NAV Indexes and SIFT Tables
- Useful Tools & Scripts & DMV's
- Index Hinting
- Form design & performance
- Locking & Deadlocks
- Maintenance
  - Dynamics-NAV Native DB
  - Sql Server
- Q & A
The **Client**:  
- basically responsible for the **user interface**.  
- also responsible for executing all the **business logic**.  
- **reads objects** from the database, **running** the objects, **controlling** their **behavior**.  
- **Most** of the Navision application runs on the clients.

The **Server**:  
- Controls the **number of users** that can connect.  
- Controls **access** to the **data** through **locking**.  
- Keeps track of all the read and write **transactions**.  
- **Sends data** to each client, as requests are made.  
- Performs all the **key-based filtering** & calculates the **SumIndexFields**.  
- **Caches** data that can be requested again.
Differences Between the Server Options

- The way you **create** a database
- The **backup** facilities that are available
- The ability to access the data in the database with **third party tools**
- The way that **SIFT™** works
- Performance **monitoring**
- **Scalability**
- Multi-processor support
Recovery Model

- Determines what information is written to the **transaction log** recovery model you want to use in database:
  - Bulk-Logged
  - Full - default option in Dynamics-NAV!
  - Simple

![Alter Database Window](image)
Whitepaper: Security Synchronization in Microsoft Business Solutions-Navision 4.0 SP3:

- When to Synchronize the Security System
- Standard Security
- Synchronizing the Standard Security Model
- Enhanced Security
- Synchronizing the Enhanced Security Model
- Selecting the Security Model
- After Changing the Security Model
- Converting the Database
- Attaching xp_ndo to SQL Server
## Security & Synchronisation

<table>
<thead>
<tr>
<th>Feature</th>
<th>Standard Security</th>
<th>Enhanced Security</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synchronization Performance</td>
<td>Fast</td>
<td>Slower If you have several companies and many users in the same database, the synchronization process will be slower with Enhanced Security.</td>
</tr>
<tr>
<td>Windows groups displayed</td>
<td>Local domain + forest of domains</td>
<td>Local domain only</td>
</tr>
<tr>
<td>Logins required in Navision</td>
<td>Windows groups and individual Windows users</td>
<td>Windows Groups + the members of each group and individual Windows users</td>
</tr>
<tr>
<td>Granularity of Synchronization</td>
<td>Entire security system</td>
<td>Entire security system and individual logins.</td>
</tr>
<tr>
<td>Automatic synchronization when you insert, modify or delete a Windows login or a database login in Navision.</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Required Extended Stored Procedure</td>
<td>xp_ndo_enumusersids</td>
<td>xp_ndo_enumusergroups</td>
</tr>
</tbody>
</table>
### Scalability and Performance

<table>
<thead>
<tr>
<th>Feature</th>
<th>Express</th>
<th>Workgroup</th>
<th>Standard</th>
<th>Enterprise</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of CPUs</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>No Limit</td>
<td>Includes support for multicore processors.</td>
</tr>
<tr>
<td>RAM</td>
<td>1 gigabyte (GB)</td>
<td>3 GB</td>
<td>Operating system maximum</td>
<td>Operating system maximum</td>
<td>Memory limited to maximum supported by operating system.</td>
</tr>
<tr>
<td>64-bit Support</td>
<td>Windows on Windows (WOW)</td>
<td>WOW</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Database Size</td>
<td>4 GB</td>
<td>No Limit</td>
<td>No Limit</td>
<td>No Limit</td>
<td></td>
</tr>
<tr>
<td>Partitioning</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>Support for large-scale databases</td>
</tr>
<tr>
<td>Parallel Index Operations</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>Parallel processing of indexing operations</td>
</tr>
<tr>
<td>Indexed Views</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>Indexed view creation is supported in all editions. Indexed view matching by the query processor is supported only in Enterprise Edition.</td>
</tr>
</tbody>
</table>
# Sql Server Versions & Limitations

<table>
<thead>
<tr>
<th>Feature</th>
<th>Express</th>
<th>Workgroup</th>
<th>Standard</th>
<th>Enterprise</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto Tuning</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Automatically tunes database for optimal performance.</td>
</tr>
<tr>
<td>Profiler</td>
<td></td>
<td>²</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>SQL Server Management Studio</td>
<td>See Comments</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Easy-to-use graphical management tool available as a separate download or included with the SQL Server Express w/Advanced Services download</td>
</tr>
<tr>
<td>SQL Server Management Studio</td>
<td></td>
<td>²</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Management Studio</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>Full management platform for SQL Server; includes Business Intelligence (BI) Development Studio.</td>
</tr>
<tr>
<td>Database Tuning Advisor</td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>Automatically suggests enhancements to your database architecture to improve performance.</td>
</tr>
<tr>
<td>Serviceability Enhancements</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Dynamic management views and reporting enhancements.</td>
</tr>
<tr>
<td>Full-text Search</td>
<td>See comments</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Available for SQL Server Express in the SQL Server Express w/ Advanced Services download</td>
</tr>
<tr>
<td>SQL Agent Job Scheduling Service</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>
How to conduct a Performance audit

- Use **checklists**:
  - Server Hardware Performance Checklist
  - Operating System Performance Checklist
  - SQL Server Configuration Performance Checklist
  - Database Configuration Settings Performance Checklist

- Use **Performance Monitor** to Identify **Hardware Bottlenecks**

- Use **Session Monitor** to locate **clients** that cause problems

- Use **Client Monitor and/or Sql Profiler** to Identify Poorest Performing **Queries**

- Use **DMV’s & DMF’s**!
## Interesting Counters

<table>
<thead>
<tr>
<th>Object Name</th>
<th>Counter Name</th>
<th>Instances</th>
<th>Best Values</th>
<th>Recommendation (Best Values not met)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory</td>
<td>Available Mbytes</td>
<td>SQL Server TS Servers</td>
<td>&gt;5MB</td>
<td>Add more memory; Reserve less memory for SQL Server</td>
</tr>
<tr>
<td></td>
<td>Pages/sec</td>
<td>SQL Server TS Servers</td>
<td>&lt;25</td>
<td>Add more memory; Reserve less memory for SQL Server</td>
</tr>
<tr>
<td>Physical Disk</td>
<td>Avg Disk Read Queue length</td>
<td>SQL Server Disks</td>
<td>&lt;2</td>
<td>Change disk system</td>
</tr>
<tr>
<td></td>
<td>Avg Disk Write Queue length</td>
<td>SQL Server Disks</td>
<td>&lt;2</td>
<td>Change disk system</td>
</tr>
<tr>
<td>Processor</td>
<td>% Processor time</td>
<td>SQL Server TS Servers</td>
<td>0-80</td>
<td>Add more CPUs</td>
</tr>
<tr>
<td>System</td>
<td>Processor Queue Length</td>
<td>SQL Server TS Servers</td>
<td>&lt;2</td>
<td>Add more CPUs</td>
</tr>
<tr>
<td></td>
<td>Context: Switches/sec</td>
<td>SQL Server TS Servers (Multi Processors)</td>
<td>&lt;5000</td>
<td>Set Affinity Mask</td>
</tr>
<tr>
<td>Network Interface</td>
<td>Output queue length</td>
<td>SQL Server TS Servers</td>
<td>&lt;2</td>
<td>Increase network capacity</td>
</tr>
<tr>
<td>SQL Server Access Methods</td>
<td>Full Scans/sec</td>
<td>SQL Server</td>
<td></td>
<td>Review Navision C/AL Code</td>
</tr>
<tr>
<td></td>
<td>Page Splits/sec</td>
<td>SQL Server</td>
<td>0</td>
<td>Defrag SQL Server; Indexes; Review Navision C/AL Keys</td>
</tr>
<tr>
<td>SQL Server Buffer Manager</td>
<td>Buffer Cache Hit Ratio</td>
<td>SQL Server</td>
<td>&gt;90</td>
<td>Add more memory</td>
</tr>
<tr>
<td>SQL Server Databases</td>
<td>Log Growths</td>
<td>SQL Server</td>
<td>0 (during peak times)</td>
<td>Increase and set the size of the transaction log</td>
</tr>
<tr>
<td>SQL Server General Statistics</td>
<td>User Connections</td>
<td>SQL Server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQL Server Locks</td>
<td>Lock Requests/sec</td>
<td>SQL Server</td>
<td></td>
<td>Review Navision C/AL Code</td>
</tr>
<tr>
<td></td>
<td>Lock Wait/sec</td>
<td>SQL Server</td>
<td></td>
<td>Review Navision C/AL Code</td>
</tr>
</tbody>
</table>
Client Monitor

- DEMO
Setup a TEST environment

- Test on a **separate** machine and not on production system!

- **Copy** the production database to the test server:
  - SQL Server: use backup/restore functions in Enterprise Manager / Mngt-Studio
  - Microsoft Dynamics NAV Database Server: use server-based HotCopy.

- **Warming up** the server to ensure that you get realistic measurements:
  - **execution plans** for most queries generated & ready for use.
  - most frequently used **data** is now available in **memory**.
Hardware Recommendations

- **CPU:**
  - #cpu’s, speed, cache
  - 64-bit

- **RAM MEMORY:**
  - As much as possible (64bit cpu)
  - SQL Server prefers to stay in RAM as much as it can

- **HARD DISK(S):**
  - **RAID:** Redundant Array of Independent Disks
    - Use RAID 1 or RAID 10, do not use RAID 5
  - You must create **several database files** & store them in **separate disks**
  - “When ever you double the amount of disks, you increase performance by 100%”
The NDBCS Database Driver

Dynamics-NAV Client

CAL

NDBC for SQL Server

Sql Server

Normal Tables + SIFT Tables

NDBC for NATIVE Server

Native DB

Normal Tables
Using Find Statements

- Microsoft Dynamics NAV DB versus SQL Server
  - Microsoft Dynamics NAV Server returns record by record
  - Indexed Sequential Access Method (ISAM)
  - SQL Server return Result Sets
  - SQL Server option for Microsoft Dynamics NAV is designed to detect if it is reading in a loop or reading single records
  - Loops → Result Sets
  - Single records → Singleton query (Select TOP 1 …)

\[
\begin{align*}
\text{FIND(‘-’)} & \quad \text{FINDFIRST} \\
\text{FIND(‘+’) } & \quad \text{FINDLAST} \\
\text{Remove ambiguity} & \quad \text{FINDSET}
\end{align*}
\]
FindSet

- **[OK]** := FINDSET([ForUpdate],[,UpdateKey]);
  - Use this function to find a set of records in a table based on the current key and filter.

- **[ForUpdate]**:
  - FALSE if you don't intend to modify any records in set.
  - TRUE if you want to modify some records in the set.
  - If you set this parameter to TRUE, a LOCKTABLE is immediately performed on the table before the records are read.

- **[UpdateKey]**:
  - This only applies if ForUpdate is set to TRUE.
  - If you are going to modify any field value within the current key, set this parameter to TRUE.
FindSet

- **Optimize** loops
- **Reuse** of cursors
- Database property **RecordSet**: Sets amount of records retrieved in default recordset (cashed)
In Microsoft Dynamics NAV, indexes are created for several purposes the most important of which are:

- **Data retrieval:**
  - To quickly retrieve a result set based on a filter.

- **Sorting:**
  - To display a result set in a specific order.

- **SIFT (Sum Index FlowField Technology):**
  - SIFT is used to maintain pre-calculated sums for various columns.
All the indexes in Microsoft Dynamics NAV are unique.

A primary key → unique clustered index on SQL Server

A secondary key → unique non-clustered index in SQL Server.

Microsoft Dynamics NAV Database Server supports SIFT effortlessly.

- In the SQL Server Option, when a SIFT field is defined on any index an extra table is created on SQL Server.
- This table is maintained by triggers that have been placed on the source data table.
Every time you **insert/delete/update** data in a base table ➔ all affected **SIFT tables** must also be **updated**.

The SIFT **trigger** manages this procedure automatically ➔ can take a long time ➔ performance decrease!

**Remember**: “Every single record that is inserted into a base table can cause hundreds of records to be updated in the SIFT tables !!!”
Sum Index Fields

- Have you designed your SIFT indexes optimally?
- Supporting too many SIFT indexes will affect performance.
- Unnecessary date fields in SIFT indexes of base table affects performance → create three times as many entries as an ordinary field.
- Too many fields in the SIFT indexes will also affect performance.
- The fields in the SIFT index that are used most regularly in queries must be positioned to the left in the SIFT index.
- Rule: the field that contains the greatest number of unique values must be placed on the left with the field that contains the second greatest number of unique values on its right and so on.
- Are there too many SIFT levels?
- If one of your SIFT tables becomes very large you might want to determine whether or not it should be optimized.
Recommendations

- **Eliminate** the **maintenance** of **indexes** that are only designed for **sorting** purposes.

- **Redesign** the indexes so that their **selectivity** becomes **higher** by putting **Boolean, Option** and **Date** fields towards the **end** of the index.

- Don’t maintain SIFT indexes on **small/temporary tables**:
  - Sales Line, Purchase Line, Warehouse Activity Line, ...

- **AVOID** **WHILE FIND(‘-’) or WHILE FIND(‘+’):**
  - automatically disables the **read ahead mechanism**.
  - use **REPEAT UNTIL NEXT** instead
Optimizing Microsoft Dynamics NAV Indexes and SIFT Tables

- **Minimize** the Number of Indexes
- Indexes on ‘Hot’ Tables
- **Redesign** Indexes for Better Selectivity
- **SQLIndex** Key Property
- **Clustered** Key Property
- Small/Temporary Tables **SIFT Maintenance**
- Minimize Number of **SIFT Buckets**
Keys & Properties
The dynamic management views (DMVs) in SQL Server 2005 ➔ what's going on inside SQL Server?

They are designed to be used instead of system tables and the various functions provided in SQL Server 2000.

The Dynamic Management Views are actually composed of both views and table-valued functions.

All are stored in the sys schema.

They all start with dm_ in the name.
Usefull Tools & Scripts & DMV’s

- 12 categories:
  - Common Language Runtime Related Dynamic Management Views
  - **I/O** Related Dynamic Management Views and Functions
  - Database Mirroring Related Dynamic Management Views
  - Query Notifications Related Dynamic Management Views
  - Database Related Dynamic Management Views
  - Replication Related Dynamic Management Views
  - **Execution** Related Dynamic Management Views and Functions
  - Service Broker Related Dynamic Management Views
  - Full-Text Search Related Dynamic Management Views
  - SQL Operating System Related Dynamic Management Views
  - **Index** Related Dynamic Management Views and Functions
  - Transaction Related Dynamic Management Views and Functions
Usefull Tools & Scripts & DMV’s

- **sys.dm_os_performance_counters**
  - all the counters that are pulled in PerfMon for the SQL Server category

- **sys.dm_db_index_physical_stats**
  - explain how much an index is used. It makes a great place for finding indexes that aren't needed.

- **sys.dm_db_missing_index_details**, **sys.dm_db_missing_index_group_stats**, **sys.dm_db_missing_index_groups**
  - identify indexes that are needed on tables.
  - index_group_stats is helpful because it identifies how many times the index could have been used

- **sys.dm_os_wait_stats**
As you can see the "sys.dm_os_performance_counters" DMV allows you an easy method to get at SQL Server performance counters using a simple SELECT statement.

You can use this DMV to develop a set of queries that you periodically run to monitor performance counters.

By routinely monitoring performance counters, and reviewing the counters to see if they are similar to your last counter snapshot, you can quickly identify if your server is having performance issues.
Usefull Tools & Scripts & DMV’s

- Database Resource Kit (PartnerSource)
  - Index Defrag Tool
  - Key Information Tool
  - Database Resource Kit.pdf
  - ...

- Tools_IndexQueries.sql
- SQL Server 2005 Performance Dashboard Reports
- ...

11/29/2007
The SQL Server 2005 Performance Dashboard Reports are Reporting Services report files designed to be used with the Custom Reports feature introduced in the SP2 release of SQL Server Management Studio.

Common performance problems that the dashboard reports may help to resolve include:

- CPU bottlenecks (and what queries are consuming the most CPU)
- IO bottlenecks (and what queries are performing the most IO).
- Index recommendations generated by the query optimizer (missing indexes)
- Blocking
The information captured in the reports is retrieved from SQL Server's dynamic management views.

There is no additional tracing or data capture required, which means the information is always available and this is a very inexpensive means of monitoring your server.

Reporting Services is not required to be installed to use the Performance Dashboard Reports.

(DEMO)
Index Hinting

- On SQL Server, you can use index hinting to force the server to use a particular index when executing queries for `FINDFIRST`, `FINDLAST`, `FINDSET`, `FIND('-')`, `FIND('+')`, `FIND('=')` and `GET` statements.

- Index hinting can help avoid situations where SQL Server’s Query Optimizer chooses an index access method that requires many page reads and generates long-running queries with response times that vary from seconds to several minutes.
  - Directing SQL Server to use a specific index can give instant 'correct' query executions with response times of milliseconds.
In Microsoft Dynamics NAV, index hinting is turned on by default and the application automatically uses this functionality to improve performance.

If you need to switch off or customize index hinting to fit your implementation, you must create a SQL Server table to store the configuration parameters.

- The parameters you enter into this table will determine some of the behavior of Microsoft Dynamics NAV when it is using this database.

In the database create a table, owned by dbo:

```sql
CREATE TABLE [$ndo$dbconfig] (config VARCHAR(512) NOT NULL)
GRANT SELECT ON [$ndo$dbconfig] TO public
```
Index Hinting

- The default value is IndexHint=Yes.

- You can disable index hinting at any level of granularity.

- There are two ways of using index hinting in your application:
  - You can leave index hinting turned on and disable it in specific places.
  - You can turn off index hinting and enable it in specific places.
Index Hinting Benefits

- Index hinting has been shown to optimize performance in the following scenarios:
  
  - Index hints **prevent** SQL Server from using an **out of date query** plan, such as a clustered index scan.
  
  - Index hints **prevent** SQL Server from **scanning** smaller tables and **escalating locks** to table locks.
In the following C/AL code, index hinting is turned on but SETCURRENTKEY has not been used:

```c-al
GLEntry.SETRANGE("G/L Account No.",'2910');
GLEntry.FINDSET;
```

This will generate the following SQL query:

```sql
SELECT TOP 500 * FROM "W1403"."dbo"."CRONUS International Ltd_$G_L Entry" WITH (READUNCOMMITTED, INDEX("CRONUS International Ltd_$G_L Entry$0")) WHERE ("G_L Account No_"=@P1)) ORDER BY "Entry No_" ','2910'
```

Note that **without** a SETCURRENTKEY, Microsoft Dynamics NAV will hint the SQL index which corresponds to the primary key in the G/L Account table. This is **not the best key** to use for this query.
Conversely, in the following C/AL code, hinting is turned on and **SETCURRENTKEY** has been used:

```c#
GLEntry.SETCURRENTKEY("G/L Account No.");
GLEntry.SETRANGE("G/L Account No.",'2910');
GLEntry.FINDSET;
```

This will generate the following SQL query:

```sql
SELECT TOP 500 * FROM "W1403"."dbo"."CRONUS International Ltd_$G_L Entry" WITH
(READUNCOMMITTED, **INDEX("$1")** WHERE (("G_L Account No_"=@P1)) ORDER BY "G_L Account No_","Posting Date","Entry No_" ',2910'
```

Now, because the **C/AL** code specifies which key to use, Microsoft Dynamics NAV hints the corresponding index from the code, which ensures that the right index is always used.
When you create an index, the data in the table is stored in the data pages of the database according to the order of the values in the indexed columns.

When new rows of data are inserted into the table or the values in the indexed columns are changed, SQL Server may have to reorganize the storage of the data in the table to make room for the new row and maintain the ordered storage of the data.

When a new row is added to a full index page, SQL Server moves approximately half the rows to a new page to make room for the new row.

This reorganization is known as a page split. Page splitting can impair performance and fragment the storage of the data in a table.
When **creating** an **index**, you can **specify** a **fill factor** to leave extra gaps and reserve a percentage of free space on each leaf level page of the index to **reduce** the potential for **page splits**.

- The fill factor value is a % from 0 to **100** that specifies how much to fill the data pages.

  - **100%** means the **pages** will be **full** and will take the least amount of storage space.
    - This setting should be used only when there will be no changes to the data, for example, on a **read-only** table.

  - **< 100%** leaves more empty space on the data pages, but requires more storage space.
    - This setting is more appropriate when there will be changes to the data in the table.
Form design & performance

- SIFT
  - Avoid FlowFields on List Forms calculated @runtime (even hidden ones)
  - Display on demand

- SourceTablePlacement property
  - Saved First or Last

- Find As You Type feature
Locking & Deadlocks

- Deadlocks
- LOCKTIMEOUT
- Minimizing the Duration of Locks
- Tools

- Tip: focus on **performance** before looking into locks because improving performance might minimize locking
Locking & Deadlocks

- A **deadlock** occurs when two or more transactions have a conflicting locking order and no deadlock can occur if the first lock the transactions place is on the same table.

- Always lock tables in the **same order**.

- Lock an agreed “**master resource**” first.
- You can also use the C/SIDE function \texttt{LOCKTIMEOUT} to temporarily enable or disable this property in the application.

- \textbf{Always rowlock}: ?
  - By default this property is not selected and SQL Server uses its default locking behavior.
  - This can improve \textit{performance} by allowing SQL Server to determine the best locking granularity.
RecordLevelLocking

Navision Database Server:

IF Rec.FIND('-') THEN
  REPEAT
    UNTIL Rec.NEXT = 0;
  Rec.LOCKTABLE(TRUE,TRUE);
  IF Rec.FIND('-') THEN
    REPEAT
      Rec.MODIFY;
    UNTIL Rec.NEXT = 0;

SQL Server:

Rec.LOCKTABLE;
IF Rec.FIND('-') THEN
  REPEAT
    UNTIL Rec.NEXT = 0;
  IF Rec.FIND('-') THEN
    REPEAT
      Rec.MODIFY;
    UNTIL Rec.NEXT = 0;
IF Rec.RECORDLEVELLOCKING THEN
    Rec.LOCKTABLE;
IF Rec.FIND(' - ') THEN
    REPEAT
    UNTIL Rec.NEXT = 0;
IF NOT Rec.RECORDLEVELLOCKING THEN
    Rec.LOCKTABLE(TRUE,TRUE);
IF Rec.FIND(' - ') THEN
    REPEAT
        Rec.MODIFY;
    UNTIL Rec.NEXT = 0;
Maintenance

- **Updating** SQL Server **Statistics**
- **Index Fragmentation**
- **Index Defrag Tool**
- **Maintenance Plan**
- **Optimalisation**

- (DEMO)
Performance Tuning

- Quick Wins
  - Upgrade to current version
  - Hardware upgrade
  - Operating System
  - SQL Server setup

- Time Consuming
  - Index & SIFT Alterations
  - Index Tuning
  - Code Changes
Resources available

- http://www.microsoft.com/sql
- http://www.sqlskills.com (Kimberly Tripp)
- http://www.sqljunkies.com
- http://www.sqlteam.com
- http://blogs.msdn.com/sqlblog
- http://sqlug.be

- http://plataan.typepad.com/microsoftdynamics

- ...

11/29/2007

'NAAM TRAINING'
Training available


- This two-day Dynamics-Nav course covers in detail the integration between Microsoft Dynamics-Nav and Microsoft SQL Server. The course is designed for technical consultants who are installing and implementing Dynamics-Nav with the Microsoft SQL Server option. This course deals with the integration, implementation, performance optimisation, maintenance, tools, locking and troubleshooting.
The End...

- Thank you for attendance and participation.

Steven Renders
Microsoft Certified Trainer
Plataan
steven.renders@plataan.be