STUDIA INFORMATICA 2007
Volume 28 Number 3A (72)

Marcin WOCH
Politechnika Śląska, Instytut Informatyki

ADMINISTRATOR'S TOOLS IN MICROSOFT SQL SERVER OPTIMIZATION FOR MICROSOFT DYNAMICS NAV

Summary. This article presents tools designed to help administrators and developers to optimize and monitor performances of SQL Server 2000 or SQL Server 2005 for Microsoft Dynamics NAV.

Keywords: SQL Server 2000, SQL Server 2005, Microsoft Dynamics NAV, Navision Attain, SQL Server Profiler, Client Monitor, Session Monitor, Index Tuning Wizard, Database Tuning Advisor.

NARZĘDZIA ADMINISTRATORA PRZY OPTYMALIZACJI SQL SERVER DLA MICROSOFT DYNAMICS NAV

Streszczenie. Artykuł prezentuje narzędzia dla projektantów systemu oraz administratorów, które usprawniają monitorowanie pracy serwera oraz optymalizację wydajności.

Słowa kluczowe: SQL Server 2000, SQL Server 2005, Microsoft Dynamics NAV, Navision Attain, SQL Server Profiler, Client Monitor, Session Monitor, Index Tuning Wizard, Database Tuning Advisor.

1. Introduction

Microsoft Dynamics Nav is an ERP (Enterprise Resource Planning) system which covers all areas of company's activities: accountancy, sales, receivables, purchase, payables, CRM, manufacturing, inventory, etc [12].

Navision system can run on two different servers: Microsoft Navision Database Server and Microsoft SQL Server [12]. To an end user there is no difference but SQL Server option

for Navision offers more scalability and tools that help improve system performances then native database. In this article SQL Server option for Navision is described.

2. Architecture Overview

Microsoft Dynamics NAV can run on Microsoft SQL Server [2], which is integrated with C/SIDE (Client/Server Integrated Development Engine). New versions of Navision support both SQL 2000 and SQL 2005 server. However, much better performances are achieved on 2005 version [12].

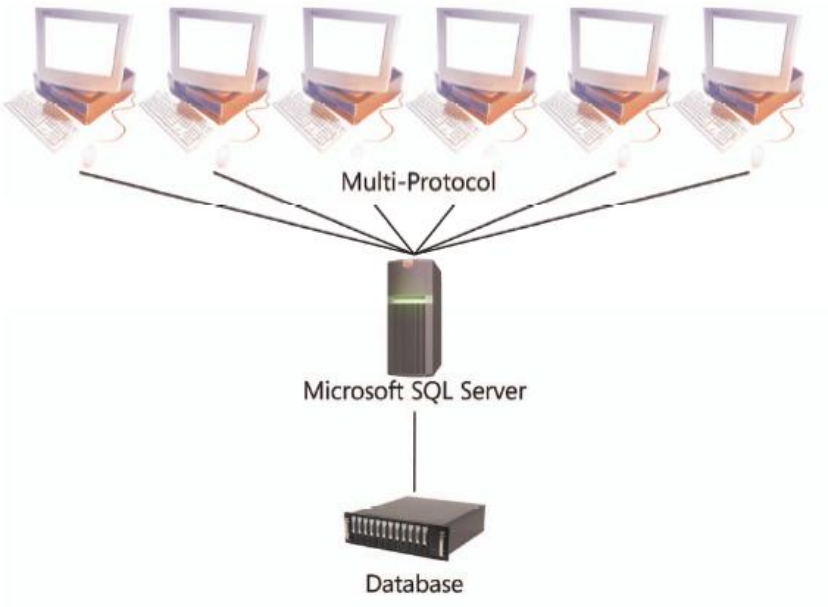


Fig. 1. MS SQL Server Option for MS Dynamics NAV architecture
Rys. 1. Architektura opcji MS SQL Server dla MS Dynamics NAV

3. Administration Tools

This chapter contains guidelines on tools that are used to optimize performances. Windows applications like Event Viewer or System Monitor, system commands and WMI language are very useful but because of many publications available they are not described in this article. For the same reason SQL Server tools are presented only in respect of their cooperation with Navision.

3.1. SQL Profiler

SQL Profiler is a graphical tool that allows system administrators monitor events in an instance of SQL Server. Data about each event can be captured and saved to a file for a later analysis. SQL Profiler should be used to monitor only events in which an administrator is interested with. Too many events being monitored add overhead to the server and slow down all processes. SQL Profiler can be used to [1]:

- monitor the performance of a server instance,
- debug T-SQL statements and stored procedures,
- identify slow-executing queries,
- test statements in a development phase,
- troubleshoot problems by capturing events on a live system and replaying them on a test one,
- audit and review an activity that occurred on a server instance.

Researches of new FIND functions (introduced in version 4.0 SP1) performed in SQL Profiler show:

- FINDSET used to modify data does not improve performances in comparison to FIND function,
- FINDSET used to modify data results with table locking,
- FINDLAST and FINDFIRST should be used only to find exactly one record. In combination with NEXT statement it deteriorates server performances when compared to FIND/NEXT functions. These functions should not be used with combination with REPEAT … UNTIL loop.

3.2. Trace logs

Trace logs contain detailed information about report server operations. They contain redundant information recorded in other log files plus additional one not otherwise available. Logs are located at \Microsoft Navision SQL Server\<SQL Server Instance>\Reporting Services\LogFiles directory. There are three types of log files:

- ReportServerService_<timestamp>.log – trace log for the Report Server Windows service and Web service,
- ReportServerWebApp_<timestamp>.log – log for Report Manager,
- ReportServer_<timestamp>.log – log for server engine.

Trace logs can be viewed in any text editor and contain:

- system information – OS version, number of processors, RAM,
- Reporting Services component and version information,

- events logged,
- exceptions,
- low resource warnings,
- inbound and summarized outbound SOAP envelopes,
- HTTP header, stack trace and debug trace information.

3.3. ODBC

Performance information can also be collected by SQL Server ODBC driver. ODBC driver can profile two types of performance data [1]:

- long-running queries,
- driver-performance data – these statistics can be written to a file or retrieved programmatically by a pre-defined structure SQLPERF.

Each application process gets its own copy of ODBC driver. After a driver opened a profiler log it does not close it until the driver is unloaded by Driver Manager. It usually happens when an application closes or tracing is stopped by a user, all the environment handles related to the driver instance are released.

3.4. Index Tuning Wizard and Database Tuning Advisor

Index Tuning Wizard and Database Tuning Advisor help the administrators select and create optimal indices, views, indexed views and partitions [5, 7]. Index Tuning Wizard is a SQL Server 2000 tool when Database Tuning Advisor has been introduced in SQL Server 2005.

They both analyze physical implementations of databases and a workload which is a set of T-SQL command executing against databases to be tuned. They use T-SQL scripts, trace logs and trace tables as in input during tuning process.

3.5. Navision Debugger

Debugger is integral part of Navision system. It helps check, correct or modify the code to improve an application.

The general term in debugging concept is a breakpoint which is a mark set on a statement [11]. When the program reaches that mark the debugger intervenes and suspends execution until user's instruction. When in a code there is no breakpoint activated then the application will run as normally even if the debugger is active.

Navision debugger provides two types of active breakpoints:

- run-time error – active debugger will stop execution when encounters an error,

- user breakpoint – a user can set up a breakpoint which will stop execution when encountered.

```
IF (SNRequired OR LotRequired) AND ("Quantity (Base)" <> 0) AND
  ("Value Entry Type" = "Value Entry Type"::"Direct Cost") AND
  NOT DisableItemTracking AND NOT Adjustment AND NOT IsServUndoConsumption
  AND NOT Subcontracting
THEN
  CheckItemTracking;

IF Correction THEN
  UndoQuantityPosting;
```

Fig. 2. Example of active and disabled breakpoint
Rys. 2. Przykładowy breakpoint aktywny i pasywny

Navision Debugger can be used to find a logical error. In that situation it allows execute one C/AL command at a time. To run the debugger in that mode Tools, Debugger, Breakpoint on Triggers option must be active.

To activate the debugger a user should choose an option Tools, Debugger, Active. It can also be started from the command line running the command: fin.exe debug.

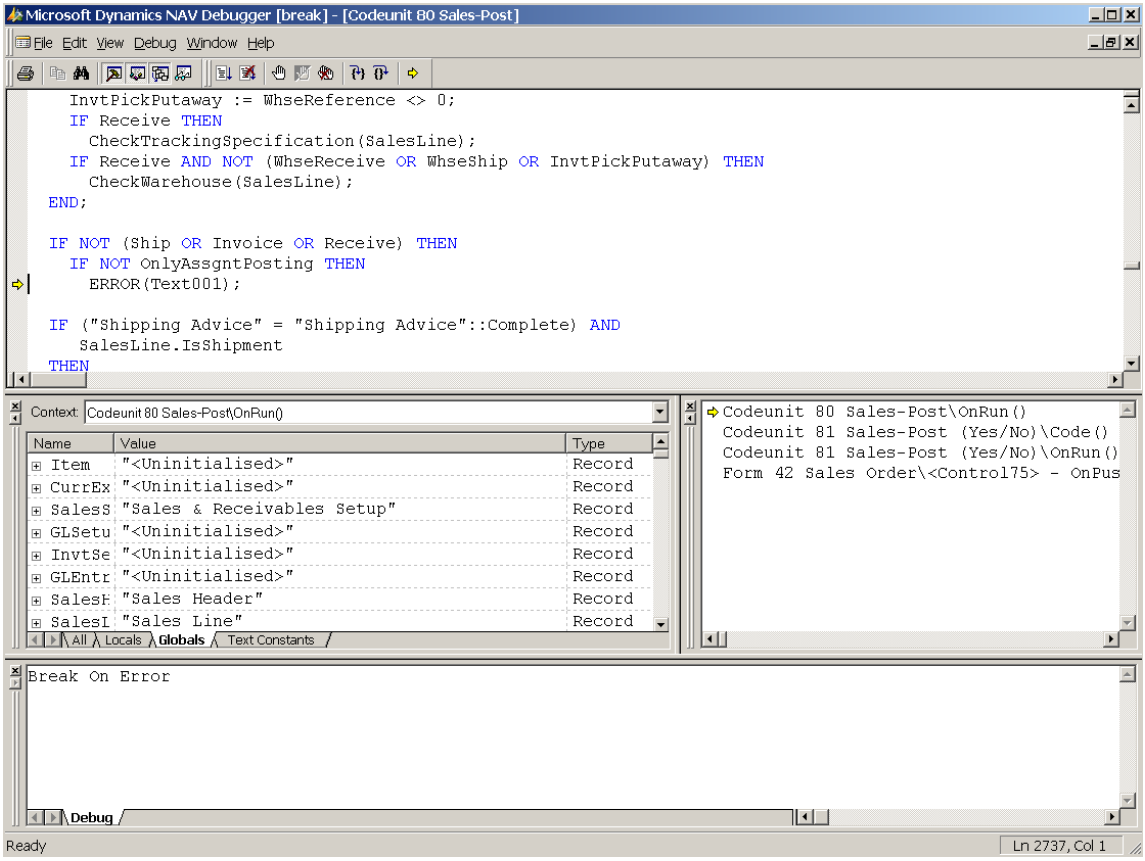


Fig. 3. Debugger Interface
Rys. 3. Interfejs debuggera

The debugger interface provides special menus, windows and dialog boxes [11]. The Call Stack box displays the stack of active functions calls. The Variables window shows all variables used in the current and previous statements grouped as local and global ones. Watch window (not presented in Fig. 3) is used to monitor variables of special interest. Variables from Variables window can be dragged and dropped into Watch window for more thorough review. Output form displays information related to debugging.

During debugging all the information about breakpoints is stored in virtual table Breakpoints (2000000059). This table keeps records about an object used, trigger line, and code no.

Breakpoints from table 2000000059 are also stored in NaviBP.xml file located in user's Application Data. This can help run Navision with certain information about stored breakpoints: fin.exe breakpoints=<path to NaviBP.xml>.

3.6. Code Coverage

Code Coverage tool is useful to test the application. Code Coverage can be started from menu Tools, Debugger, Code Coverage and then button Start or by use of C/AL function CODECOVERAGELOG. The functionality shows the objects for which the code has been executed and logged during the process. It also displays the code that has been logged during the process for a selected object.

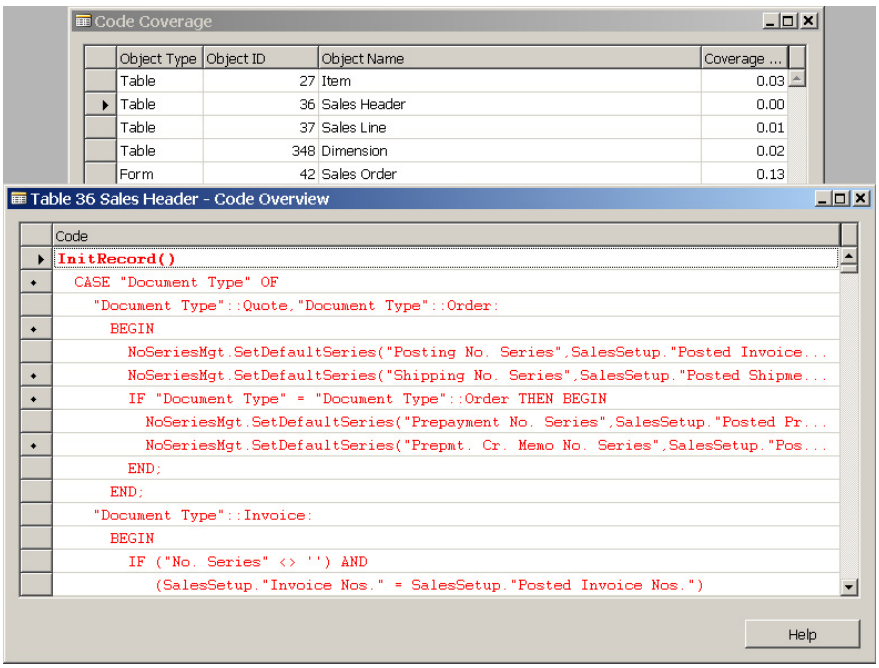


Fig. 4. Code Coverage and Code Overview windows
Rys. 4. Code Coverage oraz Code Overview

Red lines were not executed during transaction, black lines were executed. Only the executable code is marked on the left side of the window.

Code Coverage is widely used in finding so called 'dead code', which is a code never used in the transactions.

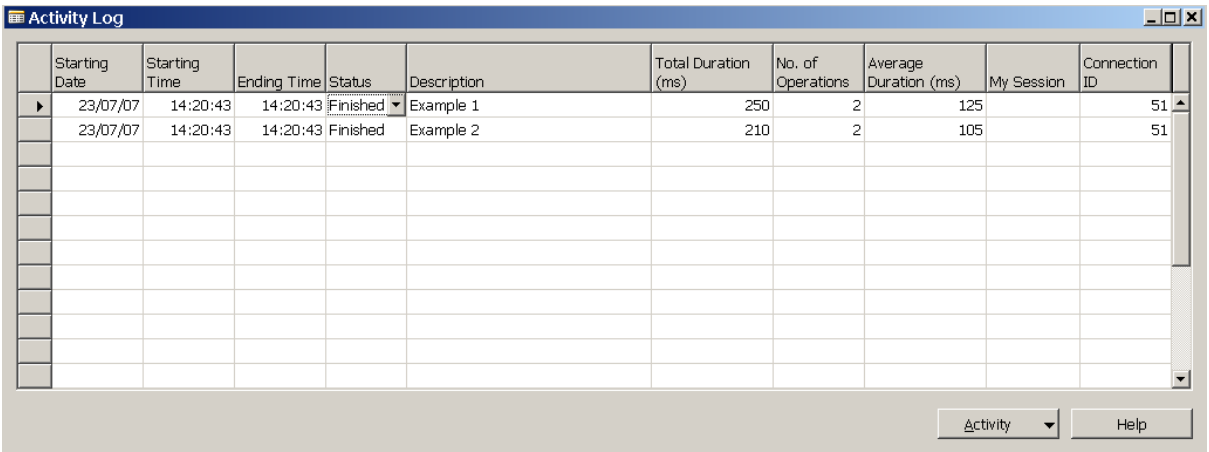
3.7. Session Monitor

This tool is used to locate clients causing performance problems. The Session Monitor window is refreshed automatically. Refreshment time is set up by a user.

The functionality tells which clients are connected to the server and the current load of the server. The most active sessions in terms of physical I/O should be investigated first. A user can also list the sessions by a memory usage. Session Monitor also gives information about CPU usage, clients waiting for locks held by other users, and clients who placed the locks. The information is given in fields Blocked, Blocked by Connection ID, Blocked by User ID, and Blocked by Host Name.

3.8. Activity Log

This tool provides information about time used by certain operations. Time can be measured by using a stopwatch or by some helper objects [1].



Starting Date	Starting Time	Ending Time	Status	Description	Total Duration (ms)	No. of Operations	Average Duration (ms)	My Session	Connection ID
23/07/07	14:20:43	14:20:43	Finished	Example 1	250	2	125		51
23/07/07	14:20:43	14:20:43	Finished	Example 2	210	2	105		51

Fig. 5. Activity Log
Rys. 5. Activity Log example

The window contains:

- activity status,
- date and time of start and finish of the activity,
- total time of the activity,
- number of operations within the activity,
- average time of each operation,
- session ID.

Menu Activity, Table Size provides information about which tables were involved in activity and how many records were added or deleted from each table.

3.9. Client Monitor

The Client Monitor is the most important tool used to monitor Navision performance and troubleshoot locking problems. It can also be used to identify incorrect server calls or indices and filters used. Client Monitor is available from version 3.10 onwards.

Client Monitor for SQL Server option provides many options to be monitored like SQL statements, execution plan, server statistics, etc.

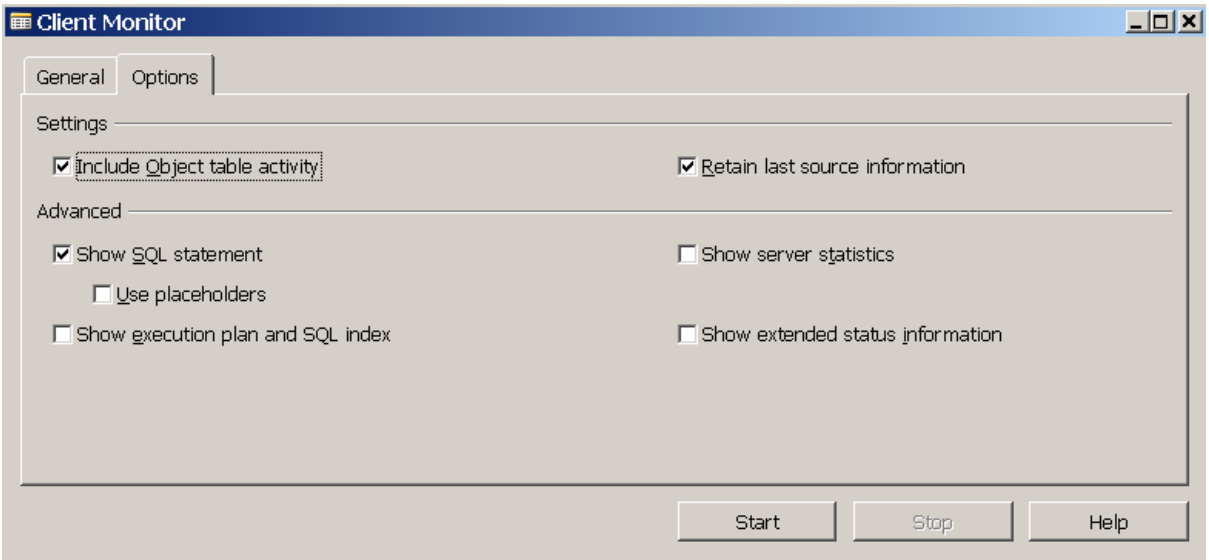
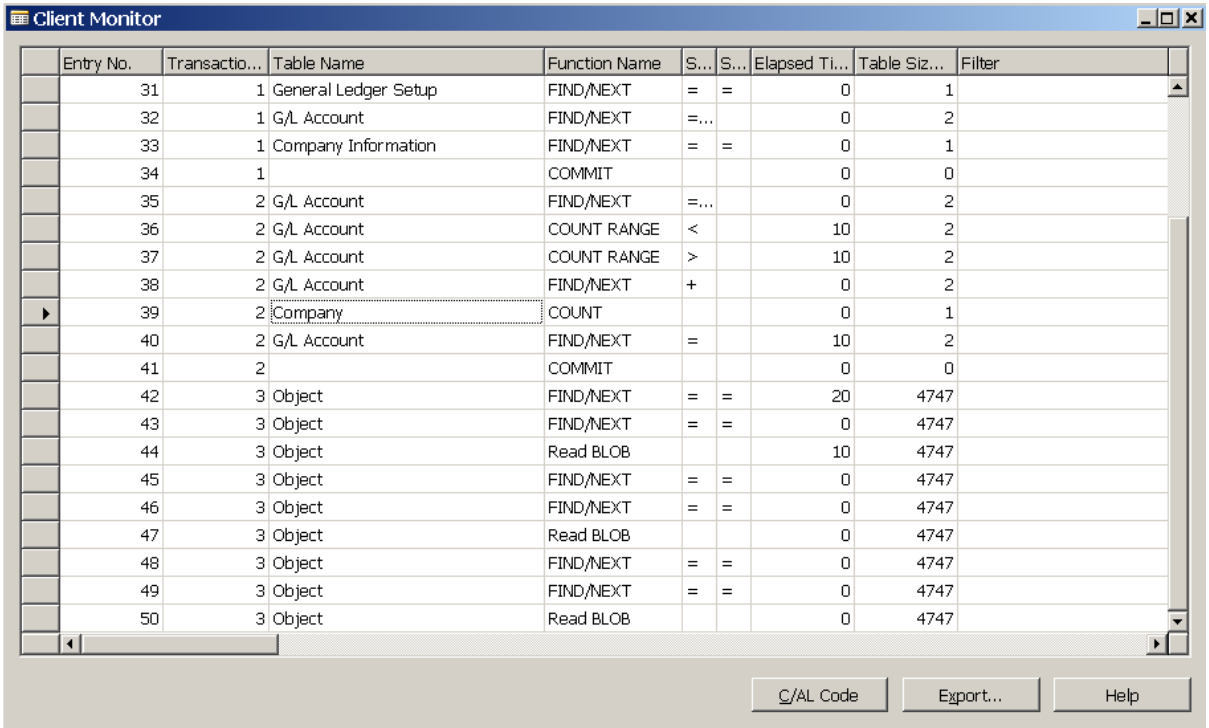


Fig. 6. Client Monitor Setup
Rys. 6. Ustawienia Client Monitor

Client Monitor retrieves data about operations performed for monitored activity. All activities are logged after pressing Start button and before the Client Monitor was stopped. There is huge number of information retrieved so it is good practice to reduce monitored activities.



Client Monitor

Entry No.	Transactio...	Table Name	Function Name	S...	S...	Elapsed Ti...	Table Siz...	Filter
31	1	General Ledger Setup	FIND/NEXT	=	=	0	1	
32	1	G/L Account	FIND/NEXT	=...		0	2	
33	1	Company Information	FIND/NEXT	=	=	0	1	
34	1		COMMIT			0	0	
35	2	G/L Account	FIND/NEXT	=...		0	2	
36	2	G/L Account	COUNT RANGE	<		10	2	
37	2	G/L Account	COUNT RANGE	>		10	2	
38	2	G/L Account	FIND/NEXT	+		0	2	
39	2	Company	COUNT			0	1	
40	2	G/L Account	FIND/NEXT	=		10	2	
41	2		COMMIT			0	0	
42	3	Object	FIND/NEXT	=	=	20	4747	
43	3	Object	FIND/NEXT	=	=	0	4747	
44	3	Object	Read BLOB			10	4747	
45	3	Object	FIND/NEXT	=	=	0	4747	
46	3	Object	FIND/NEXT	=	=	0	4747	
47	3	Object	Read BLOB			0	4747	
48	3	Object	FIND/NEXT	=	=	0	4747	
49	3	Object	FIND/NEXT	=	=	0	4747	
50	3	Object	Read BLOB			0	4747	

C/AL Code Export... Help

Fig. 7. Client Monitor
Rys. 7. Client Monitor

The window displays transaction number, tables affected by an operation, time of operation, number of records in table before Client Monitor was started, filter used and a function. Each function used by the Client Monitor [11] has its equivalent in C/AL code according to Table 1.

Table 1

Functions Calls in the Client Monitor

C/AL Function	Function in Client Monitor
GET	FIND/NEXT('=')
FIND('-')	FIND/NEXT('-')
NEXT	FIND/NEXT('>')
ISEMPTY	ISEMPTY (no MARKEDONLY filter)
CALCSUMS	CALCSUMS
CALCFIELDS	FIND/NEXT('-') – lookup FlowField CALCSUMS – sum FlowField
LOCKTABLE	LOCKTABLE
INSERT	LOCKTABLE, INSERT
MODIFY	LOCKTABLE FIND/NEXT('='), INSERT
DELETE	LOCKTABLE, DELETE
MODIFYALL	LOCKTABLE, MODIFYALL
DELETEALL	LOCKTABLE, DELETEALL

Client Monitor cannot be used to monitor temporary tables because they do not involve server calls. Is Code Coverage was started then C/AL Code button displays code for stored operations.

Client Monitor also displays information if a server call has locked the data. Data can be exported to Microsoft Excel and analyzed for example in pivot table.

Count of Table Name			
Search Result	Function Name	Search Method	Total
-	FIND/NEXT	-	1
	FIND/NEXT Total		1
- Total			1
<	FIND/NEXT	<	1
	FIND/NEXT Total		1
< Total			1
=	FIND/NEXT	=	70
		=><	3
	FIND/NEXT Total		73
= Total			73
>	FIND/NEXT	>	1
	FIND/NEXT Total		1
> Total			1
(blank)	CALCSUMS	(blank)	10
	CALCSUMS Total		10
	COMMIT	(blank)	
	COMMIT Total		
	COUNT	(blank)	1
	COUNT Total		1
	COUNT RANGE	<	3
		>	3
	COUNT RANGE Total		6
	FIND/NEXT	+	1

Fig. 8. Example of Pivot Table
Rys. 8. Przykład tabeli przestawnej

Client Monitor can be used to see if there are any locks which prevent concurrent tasks from being executed at the same time. The Client Monitor has to be run on all the computers involved in the test.

The window Client Monitor (Multi-User) contains information about the transactions potentially blocked by other users. COMMIT statement means that a server call is waiting to be completed. If any deadlock has occurred then SQL Error field would not be blank. These lines are marked red and bold.

Client Monitor (Multi-User)

Entry No.	Connectio...	Table Name	Function Name	S...	S...	Elapsed Ti...	Table Siz...	Lockir
1	51	Customer	FIND/NEXT	=...	=	0	4754	
2	51	Customer	FIND/NEXT	=	=	0	2616	

C/AL Code | Export... | Help

Fig. 9. Client Monitor (Multi-User)
Rys. 9. Client Monitor (Multi-User)

In the example two concurrent users try to read the same record from Customer table and the sessions were deadlocked. When the deadlock occurs one transaction is continued while other is aborted.

The Client Monitor gives some hints which keys and filters used should be checked and replaced in order to speed up a server query.

Client Monitor (Key Usage)

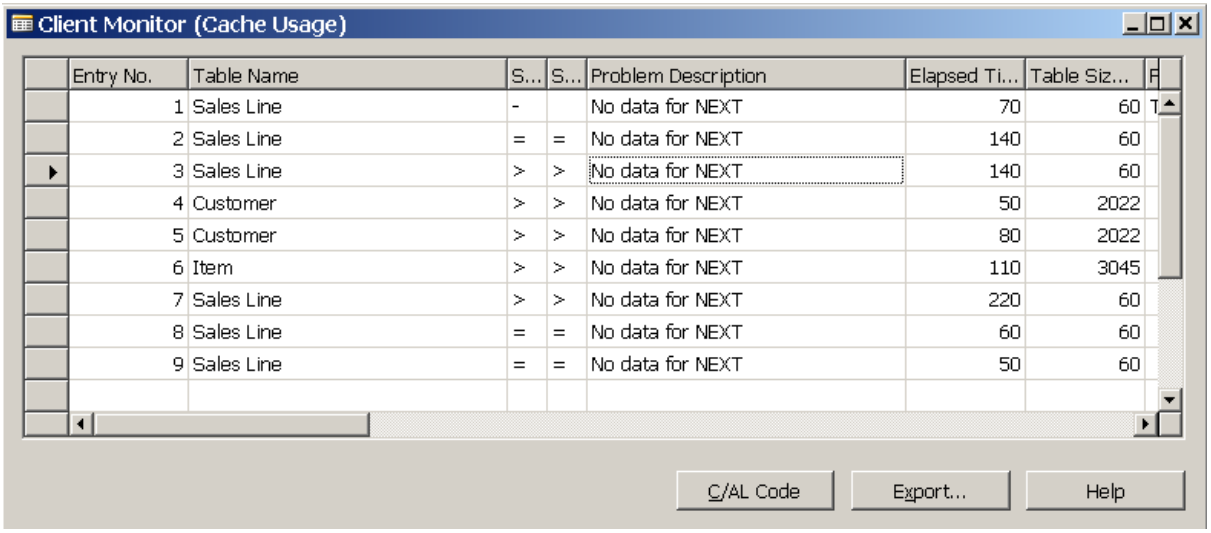
Entry No.	Table Name	Function Name	S...	S...	Good Filtered Start of Key	Key Remainder
1	Cust. Ledger Entry	FIND/NEXT	-		Customer No.,Open	Positive,Due Date,Currency
2	Cust. Ledger Entry	FIND/NEXT	=	=	Customer No.,Open	Positive,Due Date,Currency
3	Cust. Ledger Entry	FIND/NEXT	>	>	Customer No.,Open	Positive,Due Date,Currency
4	Cust. Ledger Entry	FIND/NEXT	>	>	Customer No.,Open	Positive,Due Date,Currency
5	Cust. Ledger Entry	FIND/NEXT	>	>	Customer No.,Open	Positive,Due Date,Currency
6	Cust. Ledger Entry	FIND/NEXT	>	>	Customer No.,Open	Positive,Due Date,Currency
7	Cust. Ledger Entry	FIND/NEXT	>	>	Customer No.,Open	Positive,Due Date,Currency
8	Cust. Ledger Entry	FIND/NEXT	=	=	Customer No.,Open,Positive	Positive,Due Date,Currency
9	Cust. Ledger Entry	FIND/NEXT	=	=	Customer No.,Open	Positive,Due Date,Currency

C/AL Code | Export... | Help

Fig. 10. Client Monitor (Key Usage)
Rys. 10. Client Monitor (Key Usage)

This window shows queries with filters that do not use proper keys. The key being used is shown in fields Good Filtered Start of Key and Key Remainder. Inefficiently filtered fields are displayed in field Key Candidate Fields. SQL Server always proposes a single-value field as a key or a filter. A developer should focus only on large and fast growing tables in a database and then decide whether follow the Client Monitor's suggestion.

On SQL Server sometime a function NEXT can generate separate query on a server instead of using data stored in the cache. The Client Monitor also contains a tool that can help locate such C/AL code.



Entry No.	Table Name	S...	S...	Problem Description	Elapsed Ti...	Table Siz...
1	Sales Line	-		No data for NEXT	70	60
2	Sales Line	=	=	No data for NEXT	140	60
3	Sales Line	>	>	No data for NEXT	140	60
4	Customer	>	>	No data for NEXT	50	2022
5	Customer	>	>	No data for NEXT	80	2022
6	Item	>	>	No data for NEXT	110	3045
7	Sales Line	>	>	No data for NEXT	220	60
8	Sales Line	=	=	No data for NEXT	60	60
9	Sales Line	=	=	No data for NEXT	50	60

Fig. 11. Client Monitor (Cache Usage)
Rys. 11. Client Monitor (Cache Usage)

Window lists only problematic NEXT calls. The solution of the problem is to change cache size or modify the C/AL code.

3.10. Object Cache

Object Cache is used on a client computer to store in the memory the objects retrieved from the server. In standard Navision the total size of all objects is about 20 MB. It is recommended to set it up at least as 1 MB. The more objects are being used during a client session the more memory should be reserved as an Object Cache. According to researches the average client session should have allocated about 5 MB as an Object Cache. For C/AL developers this amount should be increased at least up to 10 MB.

4. Abstract

Administrators daily routines in terms of system performance are: tracing bottlenecks and slow performance issues, finding their sources and optimizing them. Administrator's tools presented in this article assist them to improve their systems work.

It is crucial to design properly the architecture of a server, a database and an application. Statistics show that incorrect project of an application can cause 80% - 90% of performance

problems. The design phase should regard future database usage, planned data growth, coding plans as important as customer expectations.

BIBLIOGRAPHY

1. Microsoft Business Solutions Navision 4.0 Course: 8404B Installation and Configuration Training. Microsoft Corporation, 2004.
2. Dupont-Roc P., Gatchalian R., Serrano D.: Microsoft Dynamics NAV 4.0 Hardware Guide. 2006.
3. Muhlbaher H.: Optimizing Dynamics NAV on SQL Server - Application. 2007.
4. Muhlbaher H.: Optimizing Dynamics NAV on SQL Server - Infrastructure. 2007.
5. Raheem M., Sonkin D., D'Hers T., LeMonds K.: Inside SQL Server 2005 Tools. Addison Wesley Professional, Boston 2006.
6. Woody B.: Administrator's Guide to SQL Server 2005. Addison Wesley Professional, Boston 2006.
7. Rankins R., Jensen P., Bertucci P.: Microsoft SQL Server 2000. Księga eksperta. Helion, Gliwice 2003.
8. Bieniek D., Dyess R., Hotek M., Loria J., Machanic A., Soto A., Wiernik A.: Microsoft SQL Server 2005 Implementation and Maintenance, Self Paced Training Kit. MSPress, Redmont 2006.
9. Nielsen P.: SQL Server 2005 Bible. Wiley Publishing, Inc.. Indianapolis 2007.
10. Thomas O., McLean I.: Optimizing and Maintaining a Database Administration Solution by Using Microsoft SQL Server 2005. Microsoft Press, Redmont 2006.
11. Microsoft Dynamics NAV 5.00 Application Designer's Guide. Microsoft Corporation, Redmont 2007.
12. Woch M.: MS SQL Server Optimization for MS Dynamics NAV. ZN Pol. Śl. Studia Informatica Vol. 28, No. 2 (71), Gliwice 2007.
13. Microsoft Business Solutions – Navision SQL Server Option Resource Kit. 2004.
14. Microsoft Business Solutions – Navision Application Benchmark Tool 1.00. 2004.

Recenzent: Dr inż. Paweł Kasprowski

Wpłynęło do Redakcji 3 sierpnia 2007 r.

Omówienie

Microsoft Dynamics NAV współpracuje z bazą natywną, a także z serwerami MS SQL w wersjach 2000 oraz 2005 (od wersji 4.0 SP1). Prawidłowa administracja serwerem bazodanowym oraz aplikacją pozwala na optymalizację zapytań oraz manipulację danymi [12].

Rutynowe prace administratora serwera oraz systemu Navision obejmują między innymi śledzenie logów, pilnowanie blokad, zmianę parametrów wraz ze wzrostem bazy danych, optymalizację zapytań [12].

Istnieje szereg parametrów oraz narzędzi, które pozwalają monitorować pracę systemu. Narzędzia te pozwalają monitorować zarówno pracę serwera bazy danych, np. SQL Profiler, jak ich samej aplikacji, np. Client Monitor.

W codziennej pracy związanej z optymalizacją systemu Navision najczęściej używanymi narzędziami są Client Monitor oraz Session Monitor. Client Monitor pozwala na śledzenie kodu C/AL aplikacji, który powoduje lub może powodować spowolnienie procesów, a także blokady. Session Monitor natomiast służy do analizowania bieżących sesji użytkownika. Umożliwia on wskazanie pewnych „wąskich gardeł", związanych z nieprawidłowym ustawieniem systemu (Cache), a także ze sprzętem (RAM, procesor).

Liczba dostępnych narzędzi dla administratorów jest bardzo duża. Oprócz opisanych narzędzi należy wspomnieć Navision Application Benchmark Tool, który jest używany do pomiarów wydajności systemu Navision, a także wbudowanych w SQL Server analiz dyskowych, analiz cache, statystyk, flag 1204 oraz 3605, widoków, tabel i poleceń systemowych (fn_virtualfilestats, sysprocesses, DBCC SQLPERF(WAITSTATS)) itp.

Wszystkie te narzędzie są niewątpliwie bardzo przydatne w rękach wprawnego administratora, jednak najważniejszym elementem wpływającym na pracę całego systemu jest odpowiedni projekt. Nieprawidłowo zaprojektowana baza danych oraz aplikacja powodują nawet do 90% spadek wydajności. Im późniejsza faza trwania projektu, tym modyfikacje i przebudowa systemu stają się coraz trudniejsze i kosztowne.

Adres

Marcin WOCH: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16,
44-100 Gliwice, Polska, marcinwoch@wp.pl.