

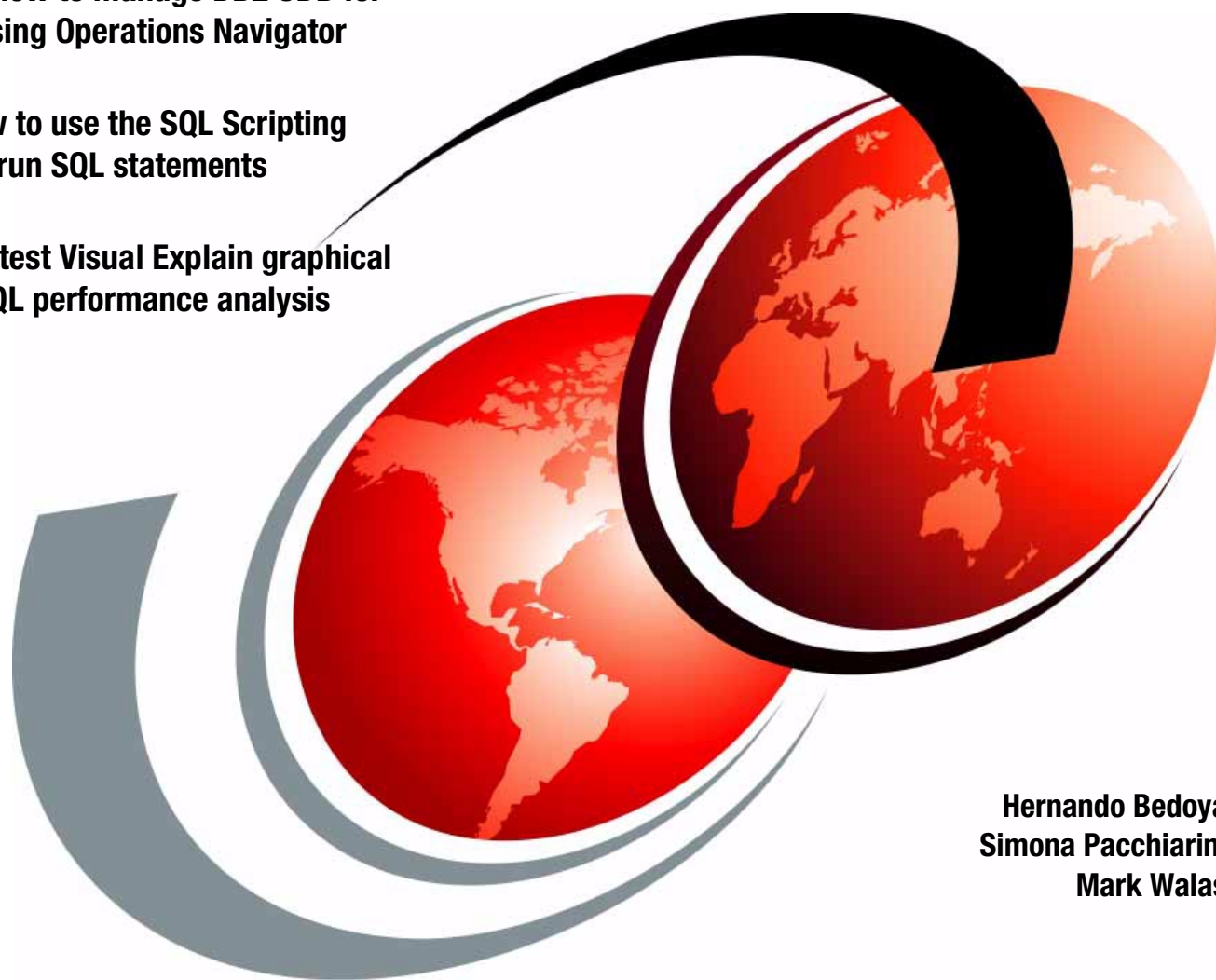
# DB2 UDB for AS/400

## Database Administration with Operations Navigator V4R5

Discover how to manage DB2 UDB for AS/400 using Operations Navigator

Learn how to use the SQL Scripting Center to run SQL statements

Use the latest Visual Explain graphical tool for SQL performance analysis



Hernando Bedoya  
Simona Pacchiarini  
Mark Walas

# Redbooks





International Technical Support Organization

SG24-5993-00

**DB2 UDB for AS/400: Database Administration with  
Operations Navigator V4R5**

January 2001

**Take Note!**

Before using this information and the product it supports, be sure to read the general information in Appendix A, "Special notices" on page 99.

**First Edition (January 2001)**

This edition applies to Version 4 Release 5, Modification Level 0 of OS/400 (5769-SS1), Version 4 Release 5 Modification Level 0 of IBM Client Access Express for Windows (5769-XE1).

This document was created or updated on January 9, 2001.

Comments may be addressed to:  
IBM Corporation, International Technical Support Organization  
Dept. JLU Building 107-2  
3605 Highway 52N  
Rochester, Minnesota 55901-7829

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 2001. All rights reserved.

Note to U.S Government Users - Documentation related to restricted rights - Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

---

## Contents

<b>Preface</b> . . . . .	v
The team that wrote this redpiece . . . . .	v
Comments welcome . . . . .	vi
<b>Chapter 1. Database administration</b> . . . . .	1
1.1 DB2 UDB for AS/400 through Operations Navigator overview . . . . .	4
1.2 Database functions overview . . . . .	7
1.3 Database library functions overview . . . . .	8
1.3.1 Creating an OS/400 library or collection . . . . .	10
1.3.2 Library-based functions . . . . .	12
1.3.3 Object-based functions . . . . .	26
1.4 ODBC Data Sources overview . . . . .	39
1.4.1 IBM-provided ODBC Data Sources . . . . .	40
1.4.2 ODBC Data Source setup parameters . . . . .	41
1.5 Run SQL Script . . . . .	45
1.5.1 Running a CL command under SQL script . . . . .	49
1.5.2 Run SQL Scripts example using a VPN journal . . . . .	51
1.5.3 Run SQL Scripts run options . . . . .	53
1.5.4 DDM/DRDA Run SQL Script configuration summary . . . . .	59
1.6 Change Query Attributes . . . . .	60
1.7 Current SQL for a Job . . . . .	62
1.8 SQL Performance Monitors overview . . . . .	64
1.8.1 Starting the SQL Performance Monitor . . . . .	66
1.8.2 Reviewing the SQL Performance Monitor results . . . . .	71
1.8.3 Importing data collected with Database Monitor . . . . .	80
1.9 Visual Explain . . . . .	85
1.9.1 Navigating Visual Explain . . . . .	87
1.9.2 Menu options . . . . .	90
1.9.3 Visual Explain example . . . . .	90
1.9.4 The Visual Explain icons . . . . .	94
<b>Appendix A. Special notices</b> . . . . .	99
<b>Appendix B. Related publications</b> . . . . .	101
B.1 IBM Redbooks . . . . .	101
B.2 IBM Redbooks collections . . . . .	102
B.3 Other resources . . . . .	102
B.4 Referenced Web sites . . . . .	104
<b>How to get IBM Redbooks</b> . . . . .	107
IBM Redbooks fax order form . . . . .	108
<b>Index</b> . . . . .	109
<b>IBM Redbooks review</b> . . . . .	111



---

## Preface

Operations Navigator offers a Windows-like graphical interface to configure, monitor, and manage the OS/400 environment. This redpiece gives you insight into the wide range of DB2 UDB for AS/400 database administration functions available through the AS/400 Operations Navigator graphical interface, which comes packaged with AS/400 Client Access Express for Windows V4R5M0.

This redpiece is an update of a Chapter 11 from the existing IBM redbook *Managing AS/400 V4R4 with Operations Navigator*, SG24-5646. It includes and updates all the information, graphics, and screens with the V4R5 features. Apart from the updates to Chapter 11 provided in this redpiece, the remainder of the redbook *Managing AS/400 V4R4 with Operations Navigator*, SG24-5646, continues to support the features of V4R4M0.

With the release of V4R5, Operations Navigator has been enhanced to improve the manageability of DB2 UDB for AS/400. The major additions to this update were DSPFD type of output for tables, views, and indexes. But perhaps the biggest enhancement is the addition of Visual Explain to the Operations Navigator database toolset. Visual Explain is a tool that graphically displays the query optimizer's implementation of a query request.

**Note:** In this redpiece, we refer to sections and chapters of the original book. However, the sections in this redpiece do not reflect the page, table, and figure numbers from the original book because they have been renumbered.

---

## The team that wrote this redpiece

This redpiece was produced by a team of specialists from around the world working at the International Technical Support Organization Rochester Center.

**Hernando Bedoya** is an IT Specialist at the International Technical Support Organization (ITSO), in Rochester, MN. He writes extensively and teaches IBM classes worldwide in all areas of DB2 UDB for AS/400. Before joining the ITSO more than six months ago, he worked in IBM Colombia as an AS/400 IT Specialist doing presales support for the Andean countries. He has 16 years experience in the computing field and has taught database classes in Colombian universities. He holds a Masters Degree in computer science from EAFIT, Colombia. His areas of expertise are database technology, application development, and data warehousing.

**Simona Pacchiarini** joined IBM in 1989 and has been working in the Client Connectivity area since 1990. Since 1999, she has been involved in the database and business intelligence arena. She is part of the midrange pre-sales technical support team—iSeries 400—Server Division for EMEA South region. In the past, she has participated in various ITSO residencies in Rochester, MN, and cooperated in writing several IBM Redbooks.

**Mark Walas** is the technical director of Sierra Training Services Limited in England. Sierra Training is a leading AS/400 education provider in the United Kingdom. He teaches AS/400 courses extensively. He has 23 years of experience in the computing field. He is currently responsible for the education strategy and course development of Sierra Training Services.

The advisor for the redbook was:

Jim Cook  
Senior Software Engineer at the ITSO, Rochester

The authors of the original redbook were:

Henning Borchers  
Debbie Carroll  
Phil Goodman  
Justin Saunders

Thanks to the following for their invaluable contributions to producing this redpiece:

Robert Driesch  
Andrew Fellows  
Jim Flanagan  
Daniel Lema  
Kent Milligan  
Jarek Miszczyk  
Tom Schreiber

---

## Comments welcome

### Your comments are important to us!

We want our Redbooks to be as helpful as possible. Please send us your comments about this or other Redbooks in one of the following ways:

- Fax the evaluation form found in “IBM Redbooks review” on page 111 to the fax number shown on the form.
- Use the online evaluation form found at [ibm.com/redbooks](http://ibm.com/redbooks)
- Send your comments in an Internet note to [redbook@us.ibm.com](mailto:redbook@us.ibm.com)



---

## Chapter 1. Database administration

The Database component of Operations Navigator provides a graphical interface for many DB2 Universal Database (UDB) for AS/400 database operations, including:

- Creating and managing tables, views, indexes, SQL and stored procedures
- Creating and managing OS/400 journals (record changes to database and other functions supporting journals)
- Entering new or modifying already created SQL statements
- Running and debugging previously created SQL statements (referred to as *scripts*)
- Saving SQL statements for later use
- Doing performance analysis of SQL statements
- Capturing current SQL statement for any job running on the system

The Database component of AS/400 Operations Navigator is not installed by default when choosing a *Typical* installation option of IBM AS/400 Client Access Express. If the Database component is not currently installed, you can run Selective Setup to install it as discussed in 2.2.4.1, “Selective Setup” of the existing redbook *Managing AS/400 V4R4 with Operations Navigator*, SG24-5646.

With proper authorization to the database objects, the user of the database graphical interface has easy access to OS/400 server administration tools, has a clear overview of the entire database system, can perform remote database management, and receives assistance for complex tasks.

For OS/400 V4R4, key enhancements to DB2 UDB for AS/400 included an interface to the SQL-specific performance monitor, and new universal database Object Relational Support functions, such as various types of binary large objects (LOBs), User Defined Data Types (UDTs), User Defined Functions (UDFs), and DataLinks.

OS/400 V4R5 delivers a mix of enhancements across a wide variety of DB2 UDB functions and interfaces.

On the performance front, the database engine is enhanced to reduce the number of cases where SQL open data paths (ODPs) are non-reusable. Specifically, it targets cases where the use of a temporary result table causes the SQL ODP to be non-reusable. The engine still uses temporary result tables, but they no longer cause non-reusable ODPs.

The maximum number of rows allowed in a table increases from 2.1 billion to 4.2 billion in V4R5. The maximum table size remains half of a terabyte (TB). In addition, you can reference more tables—up to 256—on a single SQL statement. A journal receiver maximum size increases from 2 GB to 1 TB, which reduces the frequency of changing journal receivers. Similarly, the maximum number of journal sequence numbers increased from 2 billion to 10 billion to reduce the frequency of sequence-number resets.

The AS/400 Data Loader utilities (CPYFRMIMPF and CPYTOIMPF CL commands) are easier to use in V4R5. The Copy From Import File

(CPYFRMIMPF) command was enhanced to allow the removal of leading blanks and to let a column's default value be assigned (instead of set as null) when the target database column doesn't allow null values. The Copy To Import File (CPYTOIMPF) command has a new parameter that lets you specify the code page of the target stream file, so you no longer must create the target stream file ahead of time when you need a specific code page.

Porting the database components of an application to the AS/400 system can be much easier in V4R5. Improvements to the SQL Stored Procedure language and the SQL Call Level Interface (CLI) top the list of portability enhancements. The SQL Stored Procedure language has been available since V4R2 and has been widely used to successfully port procedures written in proprietary languages such as Transact SQL and PL/SQL to DB2 UDB for AS/400. The AS/400's external stored procedure support has also been upgraded in V4R5 with the addition of Java as a supported language. A Java stored procedure maps to a method in a Java class. DB2 UDB performs the necessary setup to enable the AS/400 JVM to perform the specified requests and to convert between SQL data types and Java objects. AS/400 Java Stored Procedures provide great flexibility for application development. Existing application logic (written in RPG, Cobol, or other HLLs) can be packaged into stored procedures for reuse in e-business solutions, while writing newer logic in a more portable procedure language such as Java or SQL.

Further Java database improvements were also made to the AS/400 SQLJ support. For example, the implementation was re-engineered to deliver performance similar to static, embedded SQL and extended dynamic SQL.

V4R5 includes full autocommit support that improves the integrity transactions performed by ODBC- and JDBC-based client applications and lifts the restrictions that prevented stored procedures and triggers from making committable database changes. IBM has also made autocommit support available for V4R4 through the latest V4R4 database group PTF (see the Authorized Problem Analysis Reports APARs site <http://as400service.ibm.com/supporthome.nsf/document/10000035> and check APAR II10982 for information on available group PTFs).

The AS/400 SQL CLI has been significantly enhanced to make it more compatible with the ODBC standard. For example, applications can now fetch numeric data into character variables and use character variables to populate numeric columns. These compatibility improvements make it easier to port applications using standard ODBC calls to the AS/400 system.

V4R5 also provides Distributed Relational Database Architecture (DRDA) password encryption to improve the security of your Internet and intranet solutions.

IBM Software also has several new products available for the AS/400 to make it easier to deliver business intelligence and data warehouse solutions for your end users. These include DB2 OLAP for AS/400 and QMF for Windows for AS/400.

In V4R5, Operations Navigator has been enhanced to improve the manageability of DB2 UDB for AS/400. The major additions include a DSPFD type of output for Tables, Views, and Indexes and Visual Explain for graphical analysis of query implementations. These Operations Navigator functions, along with the pre-existing ones, are extensively discussed in this redbook.

Although OS/400 integrated DB2 UDB for AS/400 support is one of the major strengths of AS/400 systems, a complete description of this support is beyond the goal of this redbook. Good sources for details of DB2 UDB for AS/400 capabilities include:

- AS/400 Information Center: <http://www.iseries.ibm.com/infocenter>

Here you can select **Database and File Systems->Database management**. Under Database management, by selecting **DB2 Universal Database for AS/400 books online**, you can find a list of publications that contain even more information. Most of these publications are listed here:

- *DB2 UDB for AS/400 Database Programming*, SC41-5701

This book describes database capabilities, primarily outside of SQL terminology. This includes physical files (correspond to SQL tables), logical files (correspond to SQL views), fields (correspond to SQL columns), records (correspond to SQL rows), file management, and file security.

- *DB2 UDB for AS/400 SQL Programming*, SC41-5611
- *DB2 UDB for AS/400 SQL Reference*, SC41-5612
- *DB2 UDB for AS/400 Database Performance and Query Optimization*, (softcopy only)
- *Distributed Data Management*, SC41-5307
- *DB2 UDB for AS/400 Advanced Database Function*, SG24-4249
- *Developing Cross-Platform DB2 Stored Procedures*, SG24-5485
- *DB2/400: Mastering Data Warehousing Functions*, SG24-5184
- *DB2 UDB for AS/400 Object Relational Support*, SG24-5409

- DB2 Universal Database for AS/400 home page:

<http://www.iseries.ibm.com/db2/db2main.htm>

Use this link to learn about the AS/400 Database, its most recent announcements, support information, and related products.

This page features many useful links to database related issues and products (like Business Intelligence) and gives you access to a wealth of articles, white papers, coding examples, tips, and techniques.

- Self-study lab exercise with sample OS/400 database, installation instructions and lab instructions that can be downloaded from PartnerWorld for Developers, AS/400 (AS/400 Partners in Development) Web site:

<http://www.iseries.ibm.com/developer>

Select **Education->Internet Based Offerings->DB2 UDB->Piloting DB2 UDB for AS/400 with Operations Navigator**.

Under OS/400, you can use SQL interfaces to access a *database file* or an *SQL table* since these terms refer to the same object, classified within OS/400 as a \*FILE object type. You can use SQL interfaces to access the file regardless of whether the object was created with the OS/400 Create Physical File (CRTPF) command or the SQL CREATE TABLE. OS/400 also supports access to the physical file or table through a logical file (Create Logical File (CRTLF) command) or an SQL view (SQL CREATE VIEW).

Table 1 shows the corresponding OS/400 term and SQL term for physical files or tables, records or rows, fields or columns, logical files or views, aliases, and indexes.

Table 1. OS/400 term and SQL term cross reference

OS/400 create statement or term	SQL Create statement	OS/400 object type	OS/400 object attribute	SQL term
CRTPF	CREATE TABLE	*FILE	Physical File (PF)	Table
CRTLF	CREATE VIEW	*FILE	Logical File (LF)	View
CRTDDMF	CREATE ALIAS	*FILE	DDM File (DDMF)	Alias
CRTLF	CREATE INDEX	*FILE	Logical File (LF)	Index
Field				Column
Record				Row

**Note:** A DDMF represents a Distributed Data Management File. This is the original OS/400 object used to link an OS/400 program's file, open to a file on a remote system. In the context of Table 1, an alias created by SQL has no remote system specification. To determine if the DDMF/alias has any remote system specification, you can use the Work with DDM File (`WRKDDMF`) command.

**Note:** OS/400 supports an object type of table (\*TBL). This object type is for data translation.

Throughout the remainder of this chapter, we use the SQL terms table, row, and column more frequently than their corresponding OS/400 terms file, record, and field. In some cases, we use both corresponding terms, such as field or column.

## 1.1 DB2 UDB for AS/400 through Operations Navigator overview

Click the + (plus) sign next to the **Database** function for the system to which you are attached to see the three major function areas shown in the left pane and right pane in Figure 1.

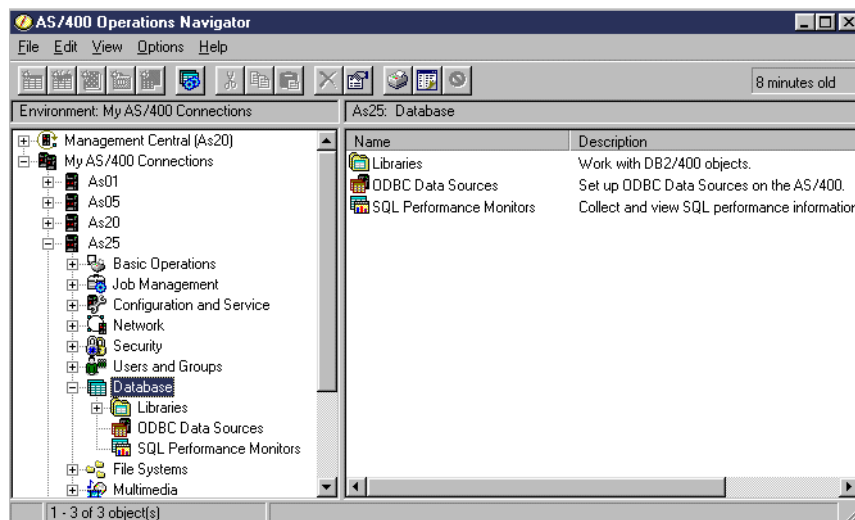


Figure 1. Database function functional areas

There are several other ways to make the same three database function areas also appear in the right pane as shown. We discuss some of these ways in this chapter. However, Operations Navigator database capabilities are actually grouped under *four* functional branches:

- Database
- Libraries
- ODBC Data Sources
- SQL Performance Monitors

The following sections summarize the capabilities under each of these four major database function groupings. Examples and tips on usage are given for selected subfunctions under each major function group to highlight Operations Navigator interfaces into the wide range of DB2 UDB for AS/400 capabilities.

We do not explain every action on every pull-down menu in this section. We explain the actions that are most significant. Actions such as Explore, Open, Shortcuts, and Print options are very similar to these same actions described for Operations Navigator in general, in 2.4, “The AS/400 Operations Navigator interface” in the existing redbook *Managing AS/400 V4R4 with Operations Navigator*. For some other database-specific actions or options, you must refer to Operations Navigator online help information.

For the database functions described in the following sections, you need the appropriate authority to perform the functions. You can use the SQL GRANT and REVOKE statements to define authority to a table, view, procedure, user-defined functions, and user-defined types. For tables and views, these statements may also specify processing authority, such as SELECT (read), INSERT (write), DELETE, and UPDATE. SQL GRANT and REVOKE can also specify column level authorities.

The Operations Navigator Database interface supports table, view, index, procedure, column, etc. database-related objects levels of authority through the Permissions action by right-clicking the database object name within a library.

You can specify permissions for all OS/400 objects, including database-related objects through Operations Navigator File Systems interface.

For general OS/400 security information and the Operations Navigator graphical interface to authorities, refer to the following chapters in the existing redbook *Managing AS/400 V4R4 with Operations Navigator*:

- Chapter 7, “Security” for General OS/400 security.
- Chapter 8, “Users and Groups” for OS/400 user and group profile support
- Chapter 9, “Authorization Lists and System Policies” for OS/400 authorization list and security policies.
- Chapter 10, “Permissions” for OS/400 Permissions.

This chapter includes information on permissions to database objects and columns.

An alternative to column level authority is to use an SQL CREATE VIEW to a table or a Create Logical File (CRTLF) command based on a file and specify only certain columns or fields. Then you specify authorities or permissions to the logical file or view.

SQL CREATE VIEW or CRTLF can also specify compare values for columns or fields that limit the rows or records that can be seen by those authorized to the view or logical file.

For additional authority details on the Object Relational Support items (functions and types), refer to *AS/400 DB2 UDB for Object Relational Support*, SG24-5409. For authority implications of using \*SYS or \*SQL naming convention when creating new DB objects with Operations Navigator, refer to document number 9510127 in *Support Line Knowledge Base* at <http://as400service.ibm.com/supporthome.nsf/document/10000051>

#### AS/400 SQL software requirements

Base OS/400 provides SQL “run time support”, not “program development for SQL support”. Run time support includes the following uses of SQL with no SQL software installation required:

- All Open Database Connectivity (ODBC) support, which includes Operations Navigator functions and Run SQL Scripts jobs and client workstation jobs using Client Access ODBC support, such as a Visual Basic program
- All Java Database Connectivity (JDBC) support, which includes client workstation Java applets and local AS/400 Java servlets accessing JDBC
- DB2 UDB for AS/400 support from an already compiled (created) local AS/400 program using embedded SQL in the RPG, COBOL, or C program
- DB2 UDB for AS/400 support from an already compiled (created) local AS/400 program using the SQL CLI (Call Level Interface) in RPG, COBOL, C, or Java

To use DB2 Query Manager support or to compile (create) local AS/400 programs using embedded SQL, such as AS/400 RPG, COBOL, and C programs, you must have licensed program DB2 Query Manager and SQL Development Kit for AS/400, 5769-ST1. This is for program development support.

## 1.2 Database functions overview

Right-click **Database** to access the pop-up menu shown in Figure 2.

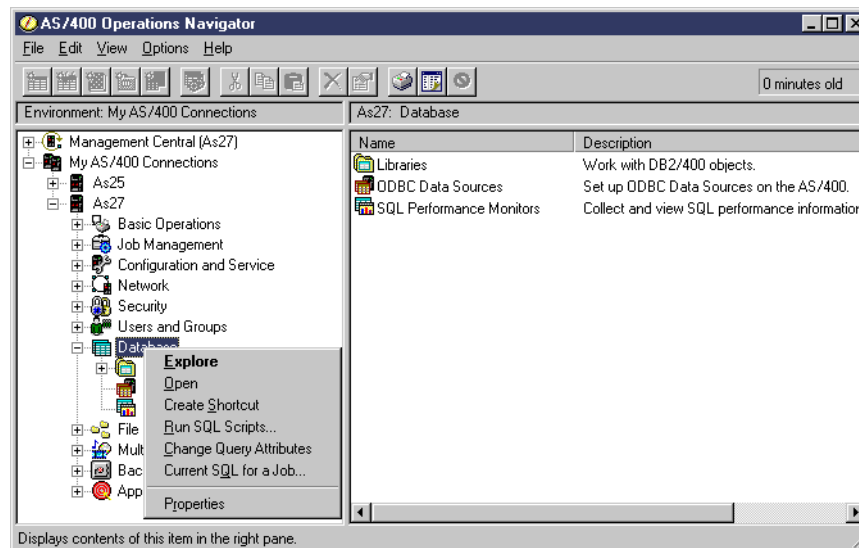


Figure 2. Database pop-up menu functions

The possible actions are:

- **Explore:** The right pane displays the three other major database function areas:
  - Libraries
  - ODBC Data Sources
  - SQL Performance Monitors

We discuss these functions later in:

- Section 1.3, “Database library functions overview” on page 8
- Section 1.4, “ODBC Data Sources overview” on page 39
- Section 1.8, “SQL Performance Monitors overview” on page 64

- **Open:** This is the same as choosing *Explore*, except that the contents of the selected file system are displayed in a separate window.
- **Run SQL Scripts:** This enables you to enter, edit, run, save, and debug SQL statements across tables within all libraries (includes SQL collections).

You can run all supported SQL statements from this action. OS/400 provides a set of base SQL statements for all supported functions that you can select and insert into your SQL statements. You can enter a completely new SQL statement or modify an already available statement for your own unique queries. You can save your own newly created or modified base statements for later use.

You must have appropriate file or table and field or column authorities (permissions) to perform the functions at run time.

Section 1.5, “Run SQL Script” on page 45, shows several examples of building and running SQL script.

- **Change Query Attributes:** Change Query Attributes enables you to specify attributes for database queries and database file keyed access path (index)

builds, rebuilds, and maintenance that are *run in a job*. Query attributes may be specified through the OS/400 Change Query Attributes (CHGQRYA) command.

In Operations Navigator, Change Query Attributes provides a graphical interface to apply a superset (more than CHGQRYA provides) of query attributes as stored in a file. These attributes can be applied to one or more active jobs that can be selected from a list.

OS/400 supplies a read-only version of the query attributes file—QAQQINI in library QSYS. Run SQL Scripts within Operations Navigator defaults to using the QAQQINI file in library QUSRSYS. You must copy the base QAQQINI file in QSYS into library QUSRSYS if you want Operations Navigator to use its values system wide, or use the CHGQRYA QRYOPTLIB (yourlib) CL command in the Run SQL Script center to default your job to another library where you have previously copied QAQQINI file. If there is no QAQQINI file in QUSRSYS, internal defaults are used.

In V4R5, you can use the *Change Query Attribute* graphical interface to easily make a copy of the default QAQQINI file in a library of your choice and to change the default values to what is most suitable for your job. We document this interface in 1.6, “Change Query Attributes” on page 60. Any changes to the attribute values are typically determined by an experienced query programmer. The best explanation of how to use these attributes can be found in *DB2 UDB for AS/400 Database Performance and Query Optimization*.

In addition to CHGQRYA, you can specify a subset of the query attributes available under Operations Navigator through the OS/400 system values QQRITIMLMT (time limit) and QQRYDEGREE (degree).

#### Restriction

You must have job control (\*JOBCTL) special authority to use this function.

- **Current SQL for a Job:** Using this feature you can select any job running on the system and display the SQL currently being run, if any. In addition to displaying the last SQL statement being run, you can edit or rerun it, through the *Run SQL Script* option, linked automatically, and to display the actual job log for the selected job or, even, to end the job.

This can also be used for database usage and performance analysis, linking into the Visual Explain tool documented in 1.9, “Visual Explain” on page 85.

- **Properties:** This enables you to specify to refresh the current display every time a list is displayed or when a specified time interval occurs.

There are several actions or functions available from the menu bar options for the Database, Libraries, ODBC Data Sources, and SQL Performance Monitors function groupings. We discuss a subset of all of these actions or functions in this chapter. You must review the online help text for a description of the entire set of actions or functions.

---

## 1.3 Database library functions overview

You can create, delete, and assign permissions (authority) to an OS/400 library under this group of functions. Under this function, you also can display the list



objects within a library and create, change, delete, or assign authorities (permissions) to an SQL table, view, alias, index or OS/400 journal, or OS/400 journal receiver listed within the library.

Figure 3 shows an example display after we clicked the + (plus) sign to the left of Libraries and then right-clicked Libraries to see the pop-up window of actions.

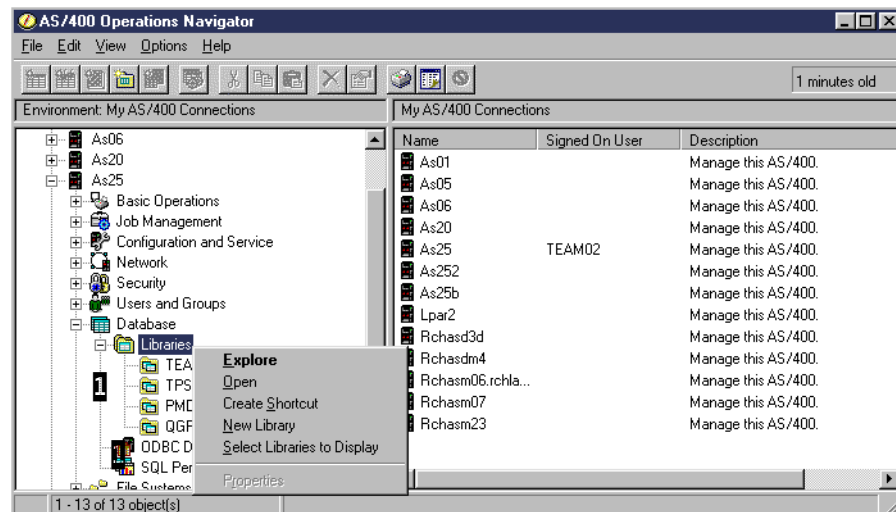


Figure 3. Operations Navigator: Database Library actions

The Explore, Open, and Create Shortcuts functions are discussed in this chapter (1.2, “Database functions overview” on page 7). You can also find greater detail on these functions in 2.4.4, “Explore and Open options” and in 2.4.6, “Shortcuts and desktop icons” in the redbook *Managing AS/400 V4R4 with Operations Navigator*.

By clicking the + (plus) sign next to Libraries, in this example, you see the library names TEAM02, TPSTAR02, PMDATA02, and QGPL at 1. These libraries were currently specified in the Initial Library List (INLLIBL) parameter of the OS/400 job description object used by the Operations Navigator session to the AS/400 system. The job description is associated with the OS/400 user profile you used to sign on under Operations Navigator when connecting to your AS/400 system. By default, only the libraries in your job’s user portion of your AS/400 library list are included under the Database component, plus any other library you asked to add here in previous working sessions.

You can add more libraries. Simply click **Select Libraries to Display** in the pop-up window by either entering a library name or selecting from a list of library names on the system. Then, click the **Add** button as shown in Figure 4 on page 10.

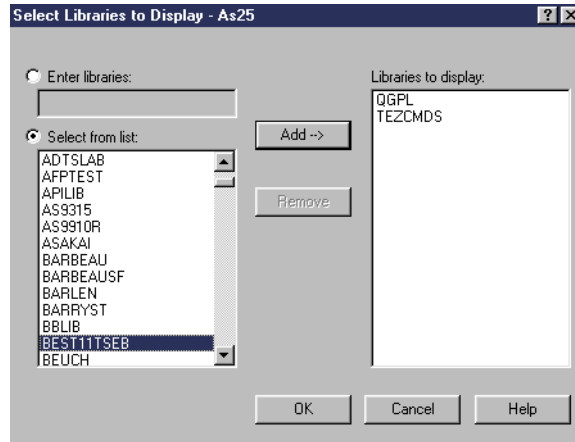


Figure 4. Database: Adding a library to your list of libraries

In V4R5, this change is retained across working sessions. Next time you reach this part of Operations Navigator, you will find your updated list of libraries.

This is done by maintaining a table in the AS/400, QAUGDBLL in QUSRSYS, which lists all users for the Database portion of Operations Navigator and all the libraries they chose to work with using this interface.

If you want to remove a library from this list, repeat the previous steps, but select **Remove**.

**Note:** Any library added here may be used when you perform actions or functions under this Libraries subcomponent of Database. Any library added here is *not* automatically used by the actions or functions under other database subcomponents such as Run SQL Scripts. Nor is it added to the user's library list on the AS/400 server. In other words, while the original list of libraries shown in Operations Navigator is built on the user's library list, a change in this interface is not going to affect the user's library list.

In V4R4, adding a library using this method caused a temporary change of the Operations Navigator list of libraries. You could permanently add a library to be displayed by setting up the user portion of your job's library list on the AS/400 system. In OS/400, each job has a library list that can be controlled by either putting a library name in the user library list system value QUSRLIBL or by putting a library name in the Initial Library List (INLLIBL) parameter of the job description (\*JOB) object associated with your OS/400 user profile.

### 1.3.1 Creating an OS/400 library or collection

There are several Operations Navigator higher level branches from which you can create an OS/400 library. This section discusses creating a library by selecting **Database->Libraries** and going to the **New Library** function as shown in Figure 5.

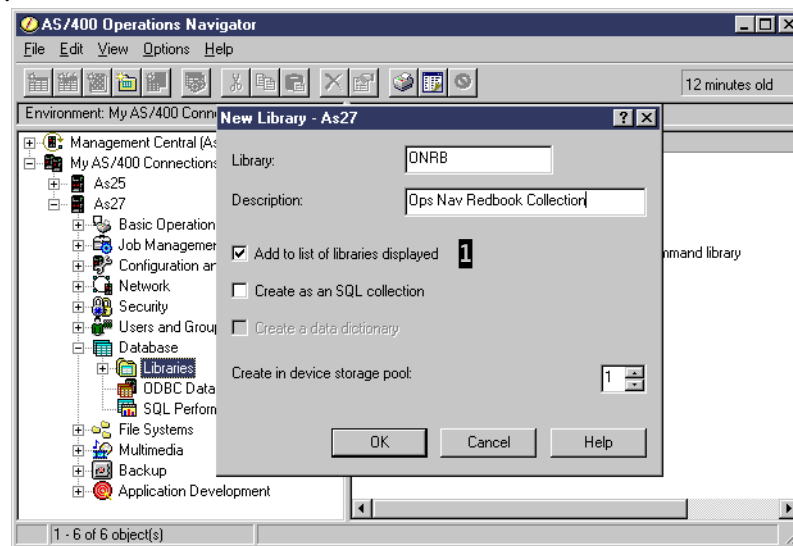


Figure 5. Database: Creating a new library

A library can contain any supported AS/400 object type. However, under the Operations Navigator's Database interface, you only work with objects related to database support. Under the New Library function, you can create a new library or you can create an SQL *collection*. In OS/400, an SQL collection automatically builds a library and within that library:

- A journal
- A journal receiver
- A catalog
- Optionally, a data dictionary

A data dictionary is used for migrated System/36 application environments.

When you select the Add to list of libraries displayed check box (indicated by **1** in Figure 5), the newly created library is added to the list of libraries that the user has in Operations Navigator. This has no effect on the AS/400 library list.

The library can be placed into the system auxiliary storage pool: ASP1 (default) or a user-defined ASP2 (up to 16). An ASP is a defined set of disk devices that contain only objects created into a library within that ASP. As shipped from IBM, ASP1 contains all disk devices. A user-defined ASP is typically used for a specific performance requirement to reduce disk arm movement or for a specific backup and recovery procedure.

For more information about ASP support, journaling support, and overall backup and recovery, please refer to:

- AS/400 Information Center (<http://www.iseries.ibm.com/infocenter>). You can select **Backup, Recovery, and Availability**.
- *OS/400 Backup and Recovery*, SC41-5304.

We show some examples using tables, views, and journals in 1.3.2, "Library-based functions" on page 12, and 1.3.3, "Object-based functions" on page 26.

All database-related objects, such as tables, views, journals, and other system objects, such as programs, message queues, spool queues, and so on, can be created, moved, or restored into any AS/400 library. All AS/400 tables or files can be created or moved into an SQL collection if the SQL collection does not contain a data dictionary.

An SQL collection can also contain catalog views that have descriptions and information for all tables, views, indexes, files, packages, and constraints created in the library. All tables created in the SQL collection automatically have journaling performed on them. When referring to an SQL collection in AS/400 documentation and screen panels, the collection name and the library name are one and the same.

### 1.3.2 Library-based functions

We use the display shown in Figure 6 to assist in providing an overview of the functions that are available when you select a specific library. These database functions include:

- Assign authorities (Permissions) to the library and objects within the library.
- Total the number of files and folders (directories) in the library and the storage size of all objects within the library (Properties).
- Create new tables (Table) within the library.
- Create new views (View) and aliases (Alias) within the library.

A *view* is an object that permits access to a subset of all rows in a table and columns within a row. An *alias* is an object that allows SQL applications to reference a table or view by another name.

Additionally aliases provide an easy way for SQL applications to access data in multiple-member native AS/400 files.

In the SQL standards, a table represents only one set of data (rows). OS/400 file support includes multiple members (sets of records or rows that contain the same field or column definitions, but different sets of data. For example, the MONTHS file can contain sets of rows for January data (member name JAN) and a set of rows for February data (member name FEB). At run time, a command parameter for Member Name (MBR) could specify JAN one time and FEB another time. Opening an SQL alias provides an equivalent function.

- Create new journals (Journal) to be used with the tables, views, or aliases.
- Create new SQL procedures: Programs called through SQL interfaces (Procedure).
- Create new DB2 UDB Object Relational user defined functions (Function).
- Create new DB2 UDB Object Relational user defined types (Type).

You need to refer to online help text or 2.4.3, “Properties windows” of the redbook *Managing AS/400 V4R4 with Operations Navigator*, for information on other actions shown on the menus in this section.

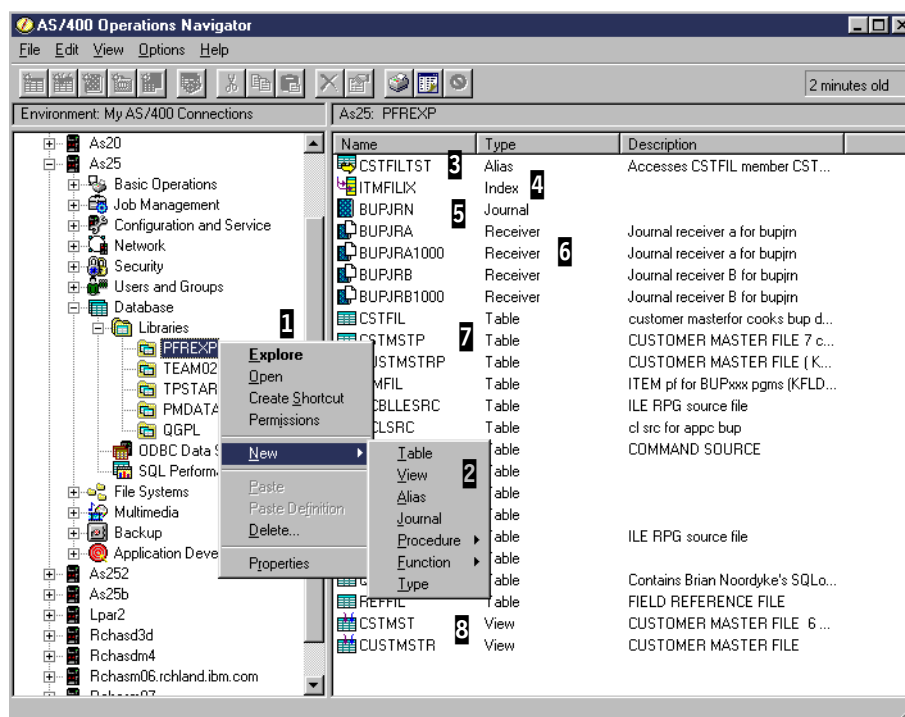


Figure 6. Database Library functions

To bring up the Library functions, right-click a specific library (PFREXP in our example). In this example, we already double-clicked PFREXP or selected the Open option in the pull-down menu shown at 1 to see the database-related objects in this library in the right pane.

By selecting New in the Library pull-down menu, the next level of objects to create (Table, View, and so forth) are shown in the menu at 2.

Before we discuss more about creating these objects, we discuss the existing objects shown for library PFREXP.

We created an alias CSTFILTST shown at 3, which accesses file CSTFIL, with member name CSTFILTST. CSTFIL is shown as a table at 7, but was originally created with the OS/400 Create Physical File (CRTPF) command.

SQL index object ITMFILIX is at 4. This object was created, based on the ITMFILE table. We describe creating an index in 1.3.3, "Object-based functions" on page 26. OS/400 files created through the Create Physical File (CRTPF) and Create Logical File (CRLF) OS/400 commands have access paths (indexes) if key fields are specified, but they are not visible as a separate object of type index.

The BUPJRN journal is at 5. Each journal can have one or a pair (dual) of attached journal receivers (where actions on the table or data within the table are actually recorded). BUPJRA and BUPJRB, as the original dual journal receivers, are shown at 6. In our example, BUPJRA and BUPJRB already reached their maximum space for journal entry information. Through journal configuration parameters, a second set of dual journal receivers, BUPJRA1000 and BUPJRB1000, have been created by OS/400, with the system generated 1000 suffix. They are now the *attached* (receiving entries) journal receivers.

A series of tables including CSTFIL, CSTMSTP, CUSTOMSTRP, and ITMFIL are shown at [7](#).

The two views CUSTOMST and CUSTOMSTR are shown at [8](#).

We do not give examples of every database function shown in these menus. In this section, we provide examples only for creating View and Journal. You may use online help information or the references given at the start of this chapter for additional information.

In 1.3.3, “Object-based functions” on page 26, we provide examples on managing tables, journals, and journal receivers.

### 1.3.2.1 Physical file and SQL TABLE differences

The OS/400 Create Physical File (CRTPF) command and the SQL CREATE TABLE statement (implicitly used by the Operations Navigator New->Table dialogue) create an OS/400 object type of \*FILE. There are CRTPF command OS/400 parameters that have no corresponding CREATE TABLE parameter. These parameters are part of every \*FILE object within OS/400 and affect the operating environment when accessing the file or table. Therefore, when you use an SQL-based interface to create a table, OS/400 uses default values for these CRTPF-only parameters. These CRTPF-only parameters include:

- **Maximum members (MAXMBR parameter):** OS/400 physical files can have multiple members (same record layout and field attributes, different sets of records or rows). All SQL tables default to a value of 1. This is also the default for CRTPF, but the user can specify a number or \*NOMAX (no limit on the number of members).
- **Member size (SIZE parameter):** OS/400 uses the number of records or rows value to implicitly allocate the initial amount of storage for the file or table. Other values in this parameter optionally specify how to allocate additional storage when the initial storage is exceeded.

CRTPF defaults to 10000 records with up to an additional allocation of up to 3000 records in 1000 record increments. A system operator message communicates each additional allocation.

CREATE TABLE defaults to \*NOMAX.

- **Reuse of deleted record or row storage (REUSEDLT and DLTPCT parameters):** When a row or record is deleted, the storage previously occupied by the record or row remains as part of the total file or table storage allocation.

DLTPCT is the percent of deleted records or rows compared to all active records or rows in the file or table. At file or table close time, if the number of deleted records or rows exceeds this percentage, a message is issued to the OS/400 History Log (viewed with the Display Log (DSPLOG) command).

REUSEDLT specifies to OS/400 whether to insert a new record or row into a new physical storage space (REUSEDLT(\*NO)) or the into the storage of a previously deleted record or row (REUSEDLT(\*YES)).

CRTPF defaults to DLTPCT(\*NONE) and REUSEDLT(\*NO). CREATE TABLE defaults to DLTPCT(\*NONE) and REUSEDLT(\*YES).

**Note:** Regardless of the DLTPCT and REUSEDLT parameter values for a file or table, you may have an application environment that you know or suspect may

have files or tables with a large number of deleted records (for example, disk storage is increasing with no known increase in the number of new records). In this case, you should consider running the OS/400 Reorganize Physical File Member (RGZPFM) command or its equivalent Operations Navigator Database Reorganize function (see 1.3.3.1, “Managing tables and views” on page 26) on a specific file or table. You can use the DLTPCT parameter message to assist you. Or, you can periodically use the Display File Description (DSPFD) command with TYPE parameter specifying \*MBRLIST to see both the number of records or rows in the file or table and the number of deleted records in each member of the file or table.

You can specify, change, and view the values for these and additional OS/400 parameters for a file or table through use of the following OS/400 commands:

- Create Physical File (CRTPF) command
- Change Physical File (CHGPF) command
- Display File Description (DSPFD) command

For more information on these and other file attributes, refer to *DB2 UDB for AS/400 Database Programming*, SC41-5701, and *OS/400 CL Reference*, SC41-5722.

You can view the above mentioned settings and other file or table parameters, such as database constraints and triggers, in Operations Navigator by right-clicking the table and selecting the menu options **Table Description** and **Properties**.

### 1.3.2.2 Create Table example

To create a new table (or file) on the AS/400 with the traditional interface, you can use DDS or the create table SQL statement. In both cases, you need the appropriate skill, whether it is programming with DDS or SQL knowledge.

Creating a new table using Operations Navigator is a simple task. You only need to understand what your needs are in terms of database structure and data types.

1. Click **Database->Libraries**. Then, right-click the library in which you want to create the new object. You are presented with a list of choices, as shown in Figure 7 on page 16.

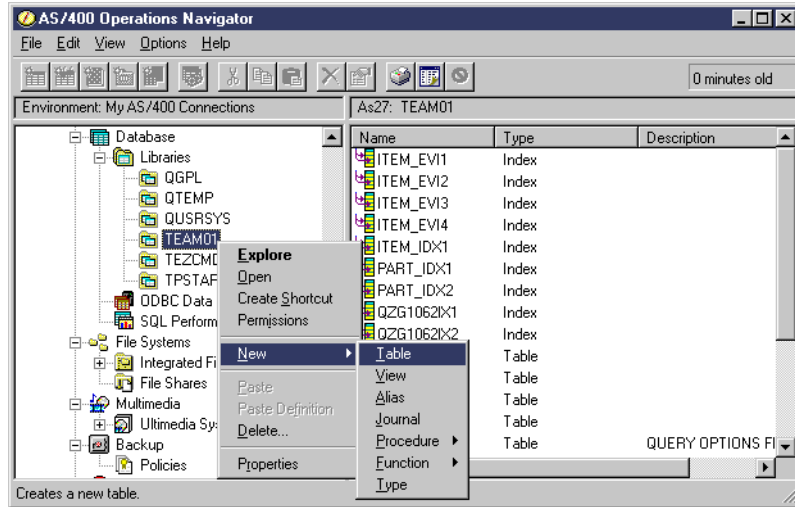


Figure 7. Create Table example (Part 1 of 3)

2. Select **Table**. Then, you are presented with a panel where you can specify the table name and description (see Figure 8). Click **OK**.

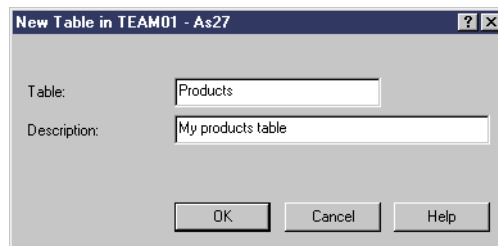


Figure 8. Create Table example (Part 2 of 3)

3. You are presented with a dialog where you can specify the columns that will constitute the table (see Figure 9). Click the **Insert** button at **|**, so you can insert new columns, for which you will be able to specify name, type, length, and an optional description.



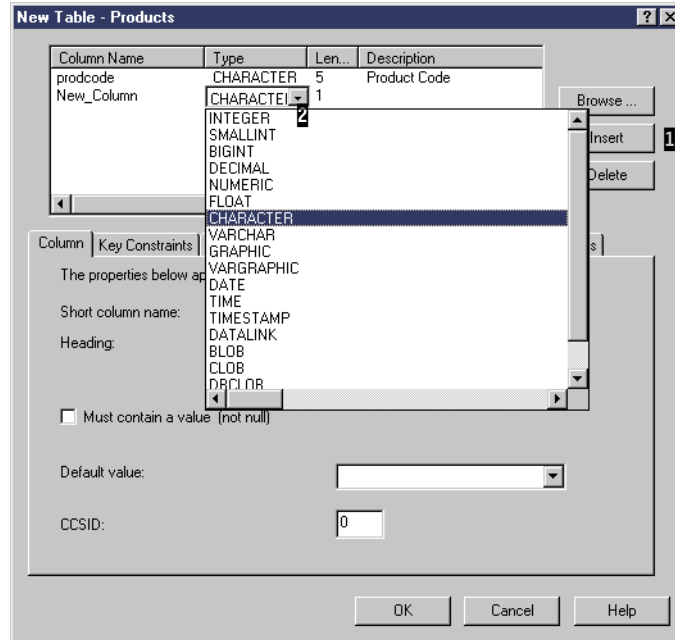


Figure 9. Create Table example (Part 3 of 3)

4. Using the pull-down list in the **Type** column (2), you can choose the data type for the column from a list. The content of this list depends on the version and release of OS/400 installed on your AS/400. Since V4R4 the AS/400 database have added support for BLOB, CLOB, DBCLOB, and datalink data types, these values only appear in the list if your AS/400 is running V4R4 or a later release.
5. When you are finished inserting columns, you can click **OK** to create the table or select any other item you may need to work on (constraints, indexes, triggers, etc.).

Follow the instructions in 1.3.3.1, “Managing tables and views” on page 26, to learn how to work on an existing table (for example, to add new columns or to work on constraints).

### 1.3.2.3 Create SQL View example

A view is typically used to represent a subset of the columns in a table and, if specified, a subset of the rows in the table.

For example, you have a customer table file that has several columns describing the customer, including customer number (key field), customer description, customer address, and customer telephone number. You want to show someone the customer number, customer name, and customer telephone number, but not their address. You also know that customers with a customer number greater than 500 do not want their telephone numbers known.

The following steps show you how to create a view (CUST\_DIMVU) over the CUST\_DIM table:

1. Determine the library into which you want to create the view. In our example, it is TSPSTAR02. Right-click the library (**TPSTAR02**) and then select **New->View** to access the new view panel shown in Figure 10 on page 18.

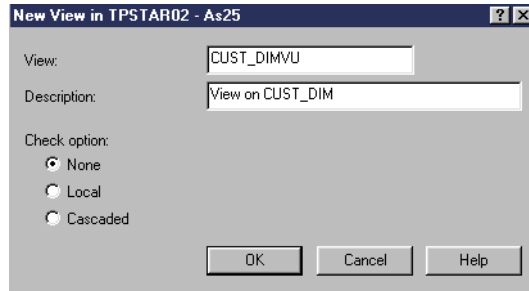


Figure 10. Create View example (Part 1 of 6)

2. Enter the name of the new view and the view's description for later reference. We specified this information for CUST\_DIMVU.

*Check option* specifies whether some type of data validity checking will be performed on an update or insert operation. You must view the help information for additional details. We selected None (default) in our example.

Click the **OK** button.

3. The next panel starts out with blank input areas, as shown in Figure 11. In this figure, click the **Select Tables** button **1**, which shows the current library list for your Operations Navigator session.

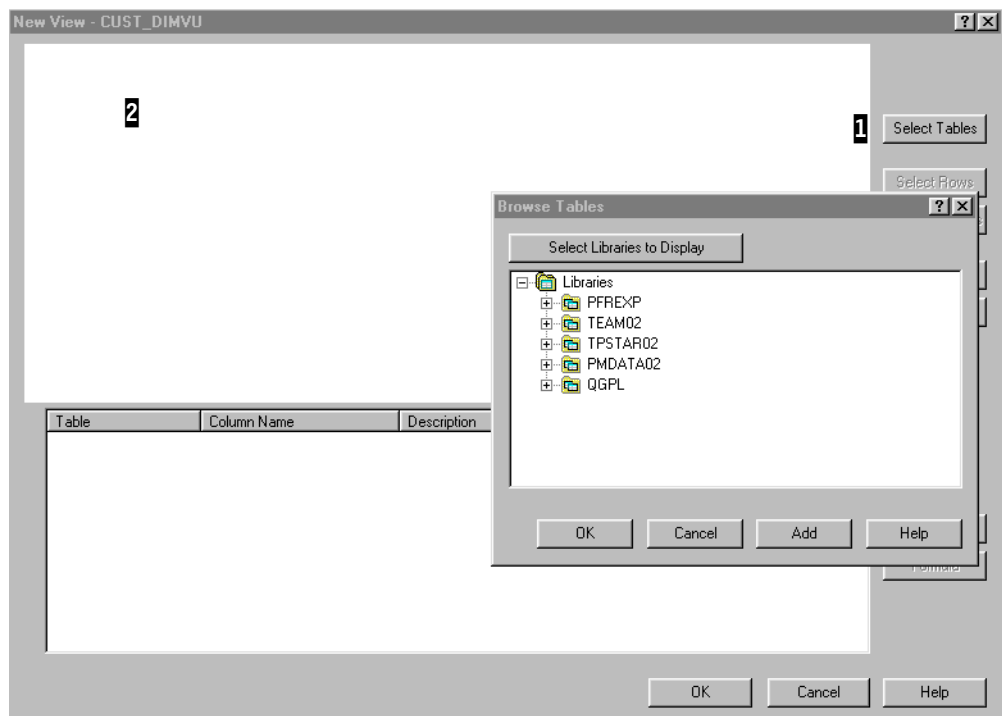


Figure 11. Create View example (Part 2 of 6)

4. Previously we decided to use library TPSTAR02 in which to create the view. However, the view can be created to use tables in various libraries. To keep this example simple, we select tables from library TPSTAR02.
5. To see the tables within a library, either click the + (plus) sign next to the library name or position the mouse on the library and double-click. Select the

table, and click the **OK** button, which places the column names in the upper pane **2** in Figure 11.

- As shown in Figure 12, select your table from the library. We selected the CUST\_DIM table from the TPSTAR02 library. Click the **OK** button shown in Figure 11. You can select another table from the library. We selected the PART\_DIM table from the TPSTAR02 library. Then, click the **Add** button.

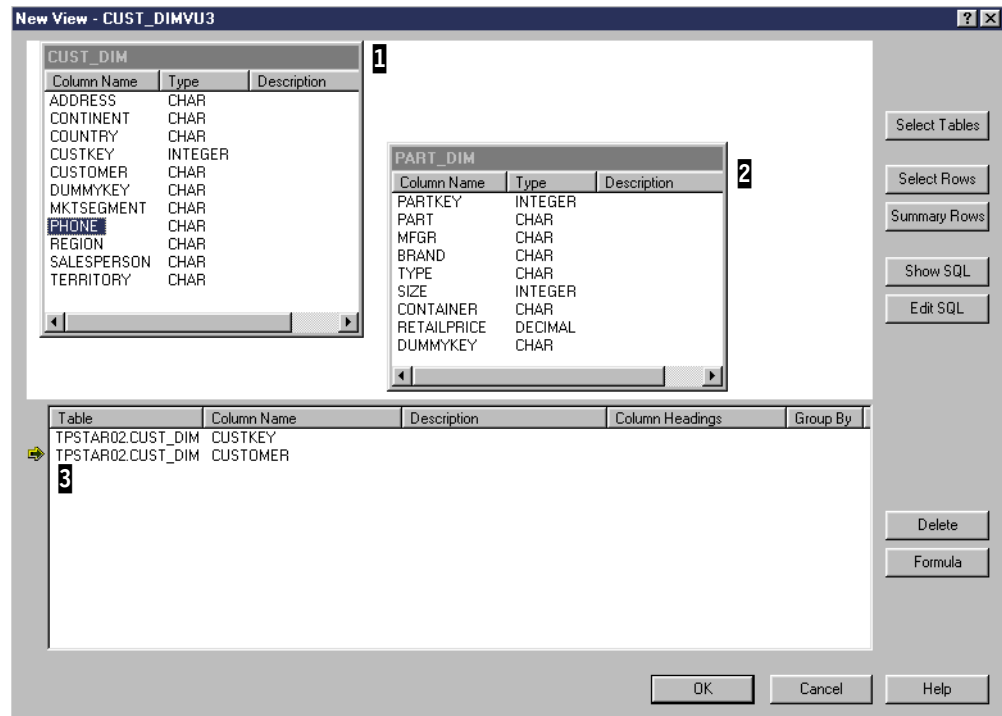


Figure 12. Create View example (Part 3 of 6)

This places the columns of both the CUST\_DIM **1** and PART\_DIM **2** tables into the upper pane. We chose two tables to show an example of how Operations Navigator assists you in building SQL statements that could become quite complex.

As shown in Figure 12, we selected the CUSTKEY and CUSTOMER columns from CUST\_DIM and dragged and dropped them into the lower pane. You can see the arrow to the left of the CUSTOMER column **3**, which indicates that the next column selection will be inserted after this statement.

You can reposition this arrow for the next insert of a new column by clicking any existing column in the lower pane. In this example, we selected the PHONE column, but have not yet dragged it to the lower pane.

In this example, we create a view using only columns from the CUST\_DIM table. If you select multiple tables to appear in the upper pane, Operations Navigator expects a JOIN clause in the VIEW statement and issues a message indicating this later if you continue showing more than one table in the upper pane. Since we are only going to use the CUST\_DIM table, we select the PART\_DIM table in the upper pane and press the Delete key to delete this table from the upper pane. The PART\_DIM column names no longer appear in the following displays.

7. As shown in Figure 13, we completed a column selection for the CUST\_DIMVU view and clicked the Select Rows button. The Select Rows button enables a WHERE clause. The Select Rows window shows the table columns, operators, and functions available in the upper pane. As a column operator or function is selected (double-click), it is inserted into the Clause pane. You may also manually enter your own text into the Clause area as we did, entering the value 500.

**Note:** If you click the Summary Rows button, the HAVING clause is enabled.

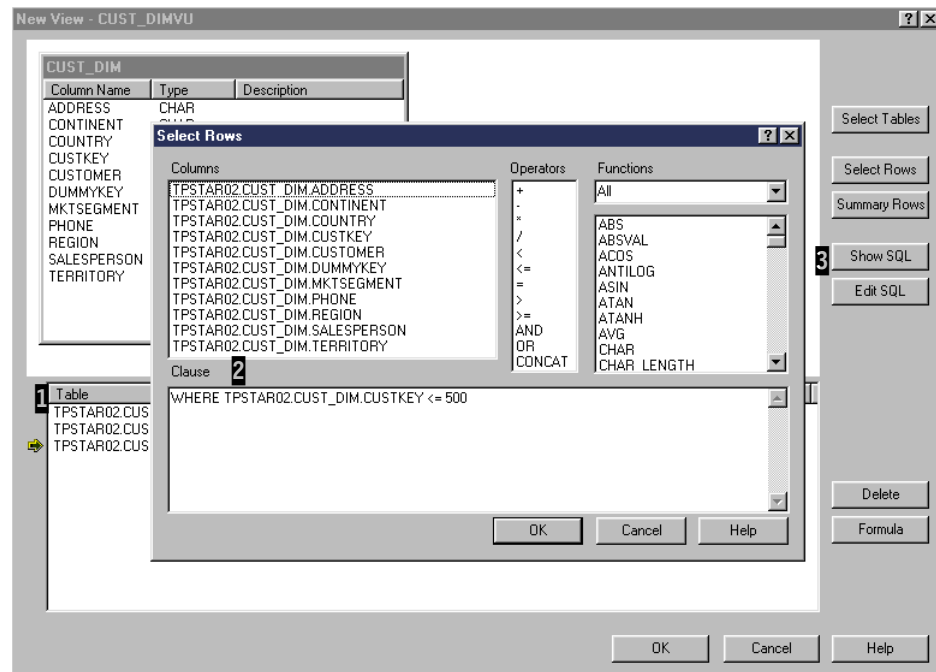


Figure 13. Create View example (Part 4 of 6)

As soon as you have at least one SQL column in the Table pane **1** or text in the Clause pane **2**, you can use the Show SQL **3** button to view the current SQL statement.

We clicked the Show SQL button to generate the Show Generated SQL window shown in Figure 14.

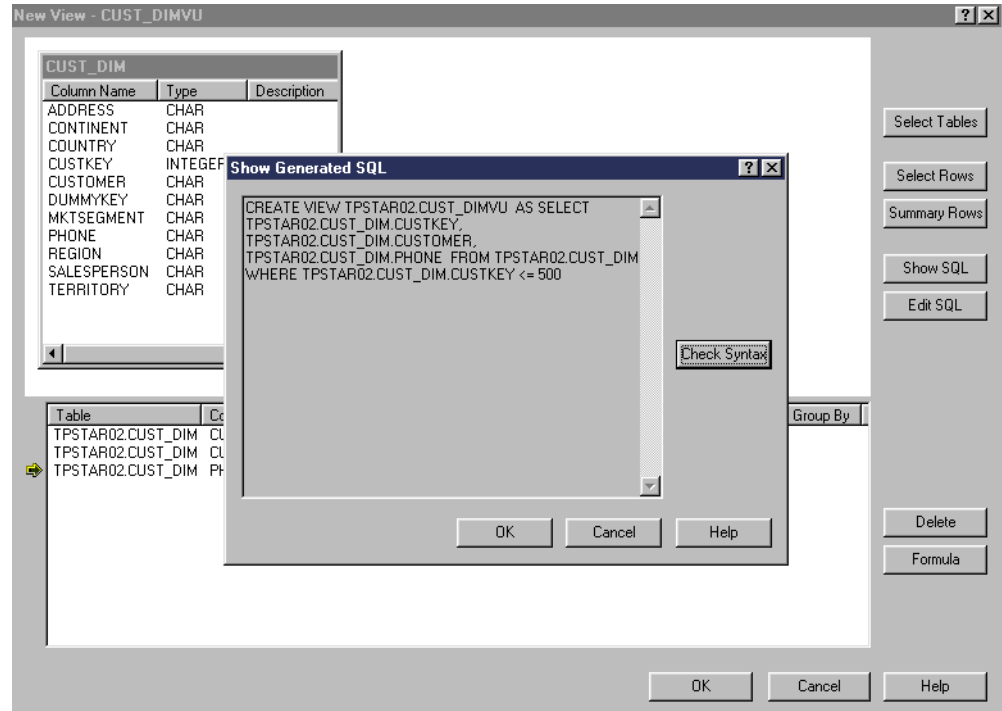


Figure 14. Create View example (Part 5 of 6)

8. On this window, click the **Check Syntax** button to view the generated SQL and have syntax checking performed. You cannot edit any text on this window.
9. If you are satisfied with the current SQL statement, you can click the **OK** button twice on successive windows and the View is created, assuming no errors are detected. Depending on your Operations Navigator refresh setting, a new view appears in an updated the display showing the contents of the library, such as the example shown in Figure 6 on page 13.
10. To edit the generated SQL, click the **Edit SQL** button, which opens the Edit Generated SQL window shown in Figure 15 on page 22.

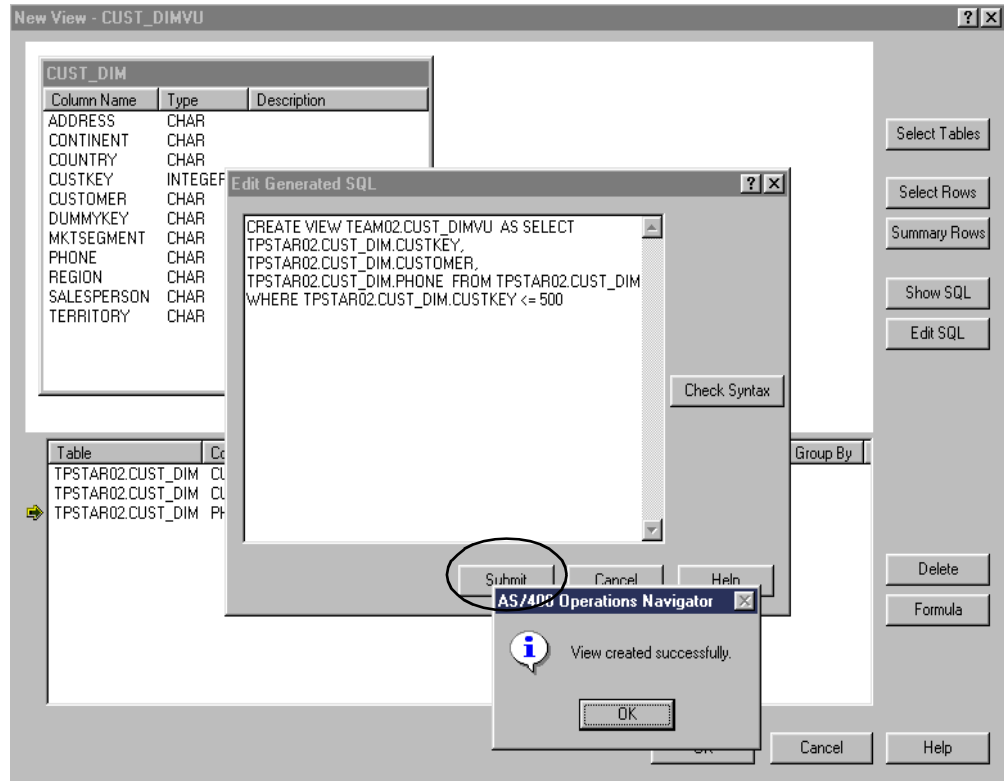


Figure 15. Create View example (Part 6 of 6)

11. In Figure 15, the SQL statement area now has a white background. Here, you can enter any characters and also have your syntax checked by using the Check Syntax button.
12. After we validated the SQL syntax, we clicked the Submit button (circled in Figure 15). Then, the view was created successfully as indicated by the Information window shown.

#### Edit SQL tip

If you make changes through this Edit SQL process, the changes are not saved. You may make changes and successfully create the view as we have done by using the Submit button. However, the changes are not saved in this dialogue because you must exit the Edit SQL function by clicking the Cancel button or using the Windows cancel (X button).

SQL changes are not saved because they could be extensive. You can even change the name of the view and the library already specified.

#### 1.3.2.4 Create journal example

A journal is an object used to record actions on database tables or files and other objects or software that support journaling, such as system auditing. For database, journals are typically used to recover unsaved changes when a serious error occurs. Commitment control, as discussed under 1.4.2, "ODBC Data Source setup parameters" on page 41, requires journaling to implement its COMMIT and ROLLBACK functions.

OS/400 uses the journal object as a *front end interface* to an *attached object*, a *journal receiver*, that actually contains the journaled data. Each set of related journal data is recorded as a *journal entry*.

Examples of non-database OS/400 software functions that optionally use journals and journal receivers include:


- OS/400 security: Action auditing
- OS/400 job accounting
- TCP/IP-based functions, including IP filters, IP Network Address Translation (NAT), and Virtual Private Network (VPN)
- OS/400 software license management tracking

Applications can also use OS/400 commands and Application Program Interfaces (APIs) to write to and read journal entries.

OS/400 supports defining and using *remote journals* as well. A journal associated with a local journal can be defined to reside on a remote AS/400 system. The remote journal can be defined so that OS/400 automatically sends journal entries made on the local system to the corresponding remote system journal. The primary intent of remote journal support is to quickly and easily replicate data onto a backup system in a high availability environment where the backup system can take over the production environment if a serious error occurs on the primary system.

To create and set up remote journaling through Operations Navigator, you must first create the local journal and journal receiver. Then use the Properties support for the journal to access actions that set up a remote journal. In 1.3.3.6, “Managing journals and journal receivers” on page 34, we show an example of journal and journal receiver properties.

The following example shows how to create a local journal (CUST\_DIMJ) in the TPSTAR02 library and create its associated journal receiver in the JRNLIB library:

1. Determine the library into which you want to create the journal. We decided to use TSPSTAR02. Right-click the library. Then, select **New->Journal** to access the New Journal panel  shown in Figure 16 on page 24.

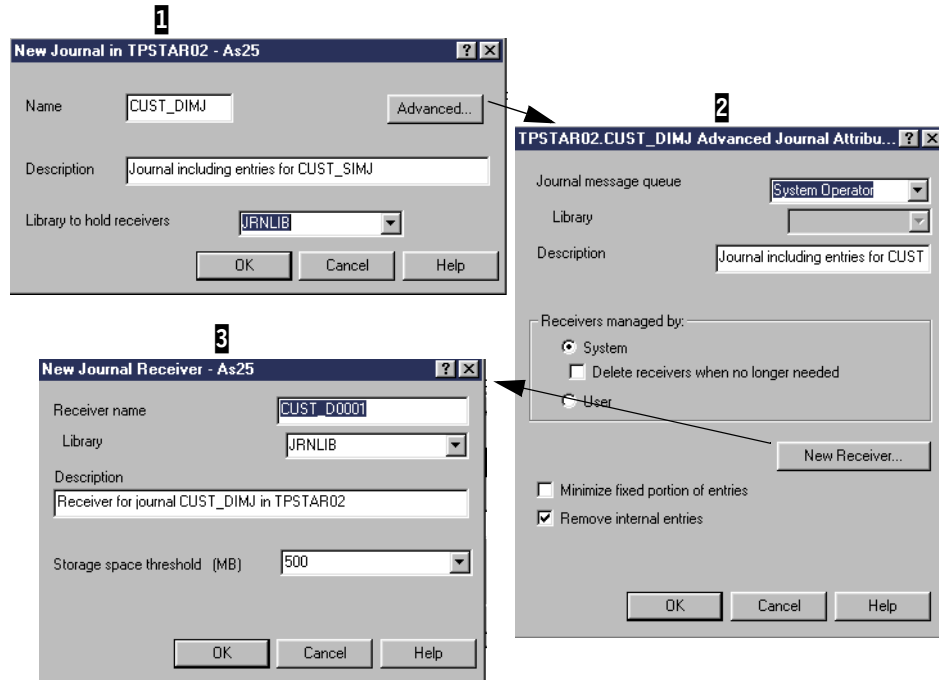


Figure 16. Creating the journal and journal receiver

2. In the New Journal panel, enter the journal name and description. Name the library to hold the journal receiver. You can select a library from a list of the current Operations Navigator session's library list, except for the library named to contain the journal. In our example, the TPSTAR02 library would not appear in the list. Although you can place a journal receiver in any library you want, including TPSTAR02 in our example, the OS/400 recommendation is to place the journal receiver in a library separate from the library that contains the journal itself.

Another recommendation for OS/400 journaling support is to place the library used for the journal receivers in its own user-defined ASP.

In our example, we specify library JRNLIB to emphasize a different library for the receiver. JRNLIB must already exist.

3. Click the **OK** button in the New Journal panel **1**. The journal is created, along with an attached journal receiver with a default name and default attributes.

If you click the Advanced button, you see the Advanced Journal Attributes panel as shown at **2**. You see the default attributes that were used in our example to create the CUST\_DIMJ journal. With the New Journal Receiver pane at **3**, we show the additional default new journal receiver attributes.

The following summary describes the key journal and journal receiver attributes. For a full discussion on these journaling attributes, refer to *Backup and Recovery*, SC41-5304.

#### **Advanced Journal Attributes**

Listed here are the Advanced Journal Attributes. Refer to the Advanced Journal Attributes window in Figure 16 as indicated by **2**.

- **Journal message queue:** OS/400 issues specific messages for specific changes to the journaling environment. The typical reason for a journaling



message is when a journal receiver is reaching its threshold of maximum entries, a message is issued indicating that the current receiver should be detached and a new, fresh journal receiver should be attached, *swap receivers*.

The default message queue is “System Operator”, which is actually message queue QSYSOPR.

In some environments, you may choose to manage your own journaling support, or you may have an application that manages the journaling through software. In those cases, you may want to use a message queue other than QSYSOPR.

- **Receiver managed by – System:** By clicking System, you tell OS/400 to automatically detach the current journal receiver and attach a new one when the journal receiver storage space threshold has been reached or when the attached journal receiver’s sequence number has reached a value of 1 TB. Each time the system attaches a new journal receiver to the journal, the journal receiver sequence number is incremented by one. Additionally the system resets the receiver sequence number during IPL, provided the receiver is not required for commitment control recovery. See Commit mode under 1.4.2.1, “ODBC Data Source Server parameters” on page 41, for information in this redbook on commitment control.


Under system managed receivers, you can also specify that OS/400 delete receivers when they are no longer needed. If you do not choose this option, the detached receivers remain on the system until you delete them.

- **Receiver managed by – User:** By clicking User, you accept full responsibility for changing journal receivers and determining when to delete receivers you no longer need.
- **Minimize fixed portion of entries:** By clicking this option, you remove job, program, and user profile information from each journal receiver entry. In a busy journaling environment, this can significantly reduce storage space required, but restricts selectivity by other OS/400 journal entry support.
- **Remove internal entries:** Depending on what is being journaled, OS/400 sometimes puts its own entries into a journal receiver. By selecting this option, OS/400 deletes these entries from the journal receiver when the system determines they are no longer needed.

One good example of these internal entries is those made to support System Managed Access Path (table index) Protection (SMAPP) support. SMAPP journals changes to access paths (that is, key columns or fields) independent of whether you use journaling of database tables or files. SMAPP is intended to minimize access path recovery following an abnormal system termination. Journaling access path changes helps SMAPP do this.

To enable SMAPP, you use the OS/400 Edit Recovery for Access Path (EDTRCYAP) command as explained in *Backup and Recovery*, SC41-5304.

### ***New Journal Receiver attributes***

Listed here are the New Journal Receiver attributes. Refer to the New Journal Receiver window indicated by  in Figure 16.

- **Journal name and description:** Enter here the journal receiver name and journal receiver descriptive text. As shown, the journal name and description are the default values generated by Operations Navigator. These values are

used if you never select the New Receiver button in the Advanced Journal Attributes pane.

- **Library:** Enter the journal receiver library. The default value shown (JRNLIB) was specified on the initial New Journal panel [1](#).
- **Storage space threshold:** Enter the maximum storage in megabytes that the journal receiver can take. You see the default value of 500 MB. The value 500 MB is specified as 5000 KB on the corresponding OS/400 Create Journal Receiver (CRTJRNRCV) command Threshold parameter.

The number of journal receiver entries this space can contain depends on the amount of data contained in each entry. When this threshold is reached, a message is sent to the message queue specified in the window pane at [2](#). See the online help information for additional details.

In addition to the powerful Operations Navigator interface to creating and managing journals and journal receivers discussed in this section and in 1.3.3, “Object-based functions” on page 26, there are several OS/400 journal creation and management commands. To view these commands and access the related online 5250 display-based help information, enter the following command on a 5250 command line:

```
GO CMDJRN
```

### 1.3.3 Object-based functions

When you right-click a specific database-related object, a pull-down menu appears with functions that are unique for that object type. At this specific object-level interface, you have some additional create functions and a wide range of management functions. Object-based functions for a database include:

- Managing a table and view
- Adding and managing constraints and triggers for a table
- Assigning and changing authorities and permissions to these objects
- Creating and managing an index for a table
- Managing a journal
- Adding and managing an associated journal receiver or a remote journal

#### 1.3.3.1 Managing tables and views

Right-clicking a table brings up a menu similar to the example shown in Figure 17.

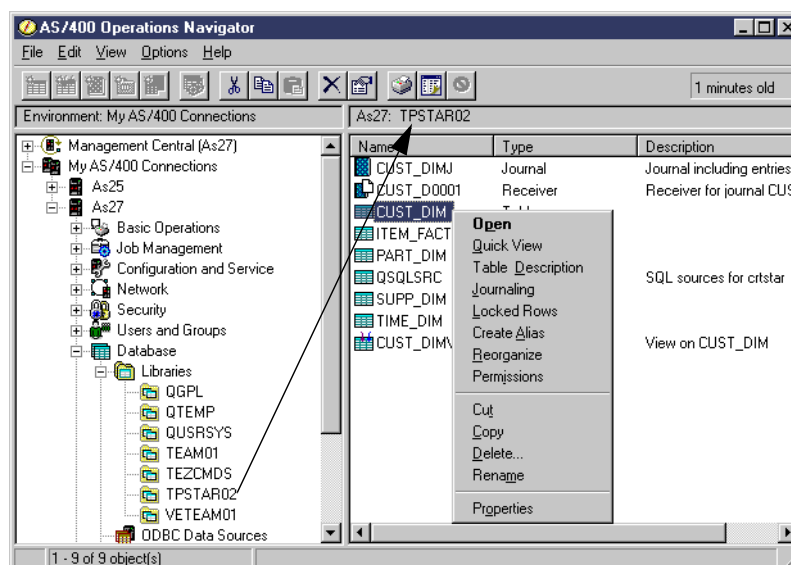


Figure 17. Managing table actions

For the CUSTDIM table, these are the actions:

- Open:** This displays, in the right pane, the first “n” rows of the table. The number of rows displayed and the number of columns displayed for each row depend on the window size, which can be adjusted to be shorter or longer (less or more rows) or narrower or wider (less columns or more columns). Assuming proper permissions, you can update columns, delete rows, and insert new rows.
 

OS/400 watches the changes you are making to a table. If you do not make exactly valid changes, OS/400 issues an error message to you. See 1.3.3.2, “Open table example” on page 28.
- Quick View:** This displays the table data as Open does, but is a read-only view. No changes can be made to the data.
- Table Description:** Using this item you can gather similar information as you would using the DSPFD command on the AS/400 system. However, here, you are also allowed to change some attributes such as Reuse of Deleted Records and Share Open Data Path. See 1.3.3.3, “Table Description example” on page 29, for details.
- Journaling:** This option displays information about the journals that are associated with tables in the database, if any exists, and lets you start journaling with all the associated options when it is needed. Journals can, therefore, be started with the Operations Navigator interface. This is similar to how they are here created, instead of using traditional AS/400 commands, as detailed in 1.3.3.6, “Managing journals and journal receivers” on page 34.
- Locked Rows:** When the table is in use, this displays whether records are locked, their relative number, which job (fully qualified job name) is actually locking them, and whether the lock type is Read or Update. From this panel, it is also possible to access the locking job’s job log, check what SQL statement is used, and copy it to a Run SQL Scripts instance to work with it. The interface also allows you to end the locking job. How to use it is documented in 1.3.3.7, “Working on Locked Rows” on page 38.

- **Reorganize:** This enables you to reorganize the rows within the table according to a specified table key, a named index, or by compressing storage currently occupied by deleted rows.

If your application implementation frequently inserts new rows and then deletes them, such as in a work file, you should consider using the compression of deleted rows function.

**Note:** Although both the OS/400 Create Physical File (CRTPF) command and SQL CREATE TABLE create an OS/400 object of type \*FILE, there are CRTPF command parameters that have no corresponding SQL CREATE TABLE parameter. Therefore, creating a table either via CREATE TABLE or by using the Operations Navigator New->Table interface requires OS/400 to use default values for these physical file parameters. One such parameter is the Reuse deleted record storage (REUSEDLT) parameter. See 1.3.2.1, “Physical file and SQL TABLE differences” on page 14, for notes on creating a new table.

- **Create Alias:** An alias is an object that allows SQL applications to reference a table or view by another name. Additionally aliases provide an easy way for SQL applications to access data in multiple-member native AS/400 files.
- **Permissions:** This enables you to view and change user profile and public authority or permissions to the table and its columns. See Chapter 10, “Permissions” of the redbook *Managing AS/400 V4R4 with Operations Navigator*, for a general discussion on Operations Navigator Permissions support. Database examples are shown in 10.2.1, “Changing permissions” of the same redbook.
- **Cut:** This enables you to select a database object and drag and drop it to a different library. When the drop is completed, the database object is deleted (cut) from the original library.
- **Copy:** This enables you to select a database object and drag and drop it to a different library. When the drop is completed, the database object exists in both the original and the target library.
- **Delete:** This enables you to select a database object and permanently delete it.
- **Rename:** This enables you to select a database object and rename it.
- **Properties:** This enables you to select a database object and display its properties. Depending on the object type (for example, a table compared to a journal or journal receiver), different property values are displayed. Also, depending on the object type, you may be able to add, change, or remove property values.

For example, when you click Properties for an SQL-created view, you can see a read-only view of the SQL used to create the view. If you click Properties for a view created by Create Logical File (CRTLF) command, you see only a message panel that states there is no SQL statement available.

### 1.3.3.2 Open table example

Figure 18 shows some example windows when performing an insert, delete, or update to a table through Operations Navigator.

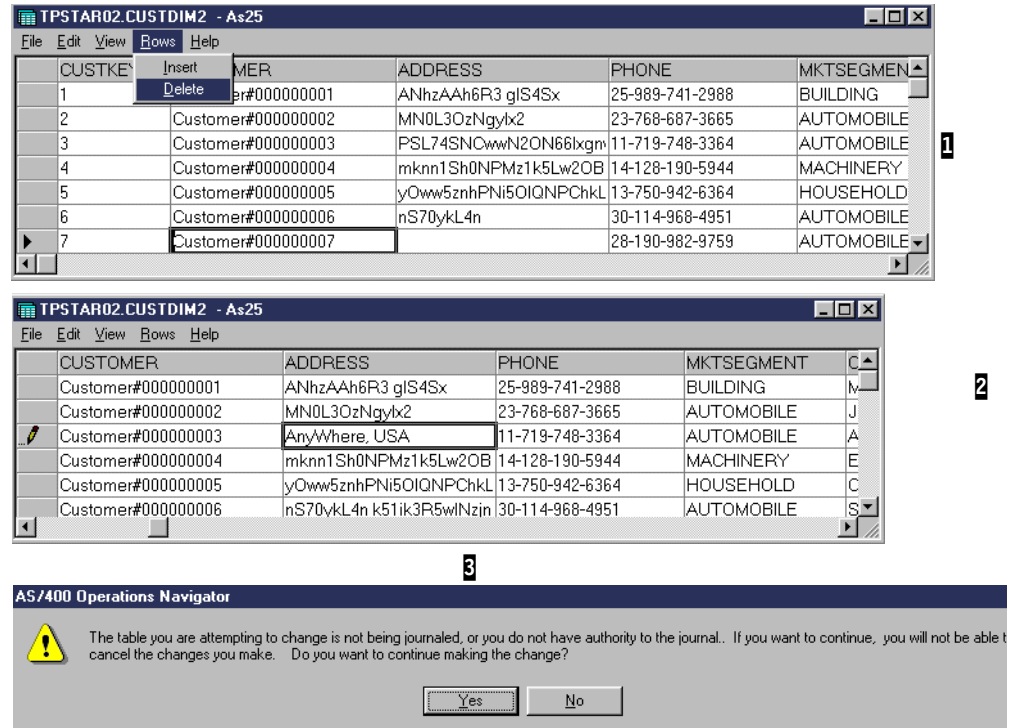


Figure 18. Open table example

In the Insert or Delete window **1**, you can insert a completely new row or delete an existing row, such as row 7 (customer number 7) in this example. For insert, you must enter the data according to each column's valid data format or you receive an error message. If you attempt to delete a row or update a column in a row (update window at **2**), you see an error message window similar to the one shown in **3**. This message cautions about recovering the original data if the table is not being journaled.

### 1.3.3.3 Table Description example

Right-click a table and select **Table Description** to see context information similar to what you see with the OS/400 DSPFD command.

This interface provides information formatted on different notebook pages and is structured as follows:

- **General tab** (Figure 19 on page 30): Displays such information as the member name, its description, size, number of current and deleted rows (therefore, making it easy to judge whether reorganize can be useful), maximum percentage of deleted rows allowed, and gives you the ability to change the Reuse Deleted Rows (REUSEDLT) parameter and the table description.

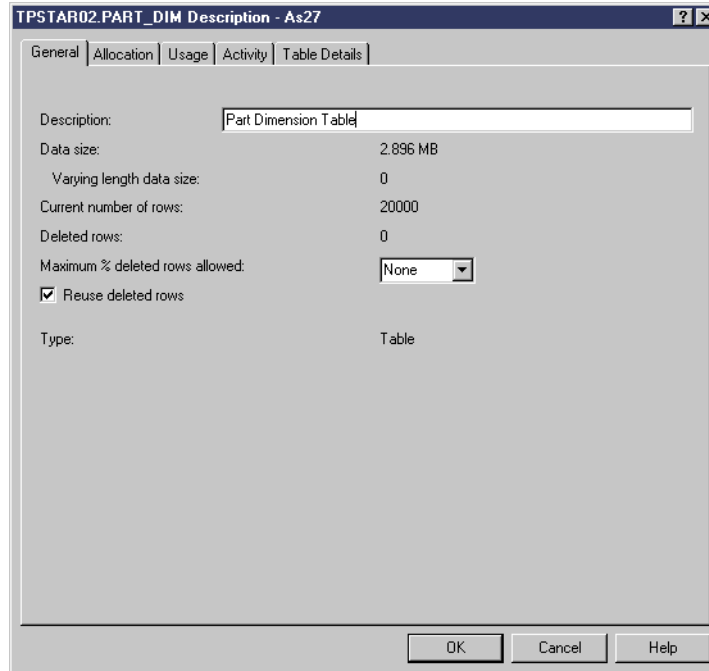


Figure 19. Table Description Panel - General

- **Allocation tab:** Allows you to verify if there is a maximum number of rows set on the table, what was the initial number of rows, its subsequent increase, if there is a value set for forcing the writing of updates to auxiliary storage, and to change these settings.

- **Usage tab:** Contains information on creation, modifications, backups and if this files data can share open data paths across jobs (SHARE \*YES/NO).

The *Share open data path* check box offers an easy way for you to change this setting for the table.

- **Activity tab:** Allows you to record the level of activity, documenting such information as the number of insert, update and delete operations, the logical and physical reads, clear operations, index builds/rebuilds, a full open and close, reorganize operations, and the number of rows being rejected in open operations (by key, non-key, group by/having selection methods).

This tab also contains important information regarding the number of *valid* and *invalid indexes* built over the table.

- **Table Details tab** (Figure 20): The last tab gives information on:
  - Creation
  - Number of allowed members
  - Maximum time a program is to wait for the file and its rows to be available
  - Maximum row length
  - Sort sequence
  - Language identifier
  - Format level check and identifier
  - Allowed activity level
  - Unique identifier for this table in the system
  - Disk pool it is using and if it is a distributed file

Some of the above mentioned values can be changed using this interface.

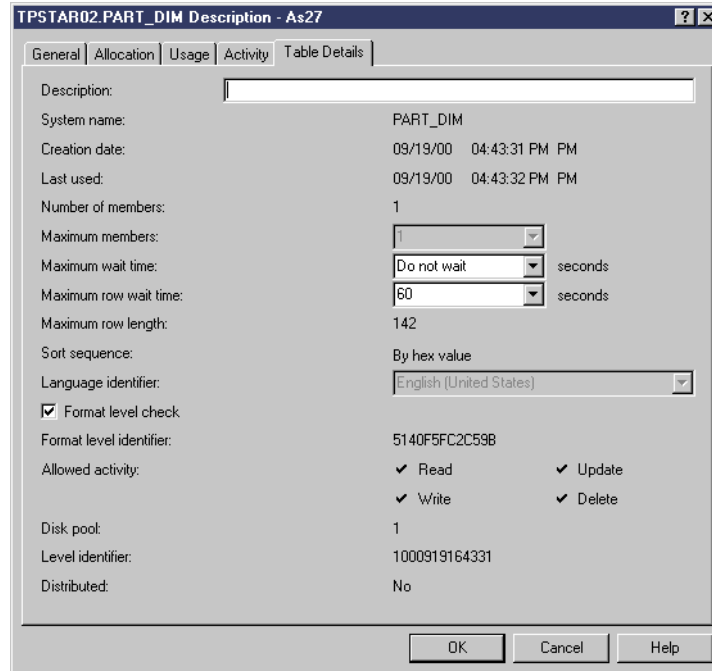


Figure 20. Table Description - Table Details

#### Changing properties cautions

You must ensure that proper authorization or permission has been given to the Operations Navigator user to access the Table Description and Properties function for the object. You must also ensure the authorized user understands the importance of any table modifications they make. For example, the properly authorized user can delete fields or columns, and therefore, lose the associated data.

Programs created (compiled) against the table that has a field or column added or removed may encounter an error during the next file or table open function. Through the Create Physical File (CRTPF) or Change Physical File (CHGPF) command, you can specify the Level Check (LVLCHK) parameter. A table with LVLCHK(\*YES) specified detects the added or removed column during file open. Re-creating the program usually resolves the problem if the program does not need to use the column.

A program that may have already been performing column validity checking performs unnecessary duplicate processing if a check constraint is added to the table.

#### 1.3.3.4 Table Properties example

Right-click a table, and select **Properties** to display all the table properties. We use the initial properties panel (Column information) as shown in Figure 21 on page 32 to discuss table properties:

- Column properties
- Key constraints
- Indexes

- Referential constraints
- Triggers
- Check constraints

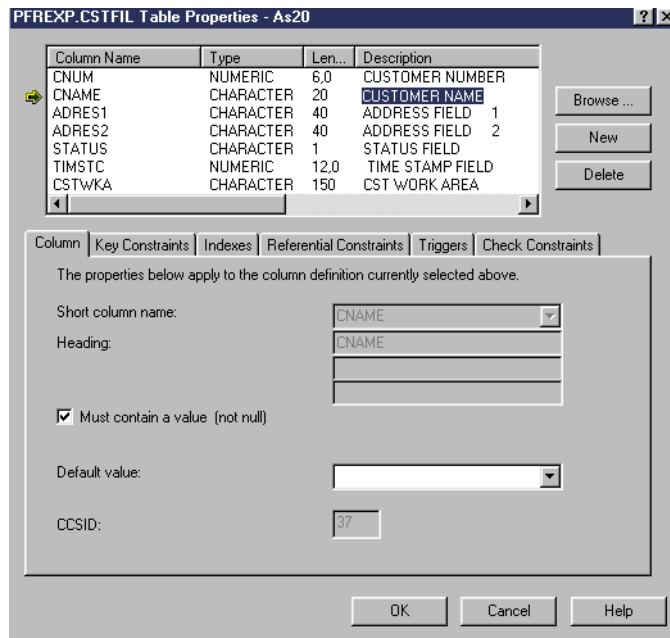


Figure 21. Table properties example

### Column properties

As shown in Figure 21, we moved the cursor to the CNAME field or column, as indicated by the yellow arrow to the left of the column and the highlighted column description. In the upper column list, you see the column data type and length. In the lower pane, you see some information about the column, including the Coded Character Set Identifier (CCSID).

The CCSID numeric value specifies how character data is stored on your system. For user-created tables, the character data is defaulted to be stored in the format according to your primary language ID. For example, on the systems used for this redbook, the OS/400 Language ID system value QLANGID is set to ENU – English for United States (uppercase and lowercase). The default CCSID value for ENU is 37, as shown in our Properties example. For more details on CCSID support, refer to *AS/400 National Language Support*, SC41-5101.

The Browse button leads to a dialogue in which you can view other tables that you may want to use as a base definition to add (copy in) a new column to table CSTFIL.

The New button enables the Column window shown at the top of Figure 21 to accept a new column definition. In the Column window, you enter the appropriate definition information. Select a column in the Column window, and click the **Delete** button to remove the column from the table.

You can make other changes or additions to the table and, when finished, click the **OK** button to make the changes permanent. The changes or additions are run as if you entered the SQL ALTER TABLE statement. If the table was created with the CRTPF command, the original file is deleted and the new file is recreated. The field or column deleted also deletes the associated data.



### **Key constraints**

Constraints place some controls on the action to an object or portion of an object. This Key Constraints tab enables you to add, modify, view, or delete the primary key and unique keys for a table. You may modify a constraint if it was defined during your current table editing session. If you added the constraint and then clicked OK on either the New Table dialog or Table Properties dialog, you may only view the constraint.

A unique constraint is the rule that the values of the key are valid only if they are unique. Unique constraints can be created using the CREATE TABLE and ALTER TABLE statements. Unique constraints are enforced during the execution of INSERT and UPDATE statements.

A PRIMARY KEY constraint is a form of the UNIQUE constraint. The difference is that a PRIMARY KEY cannot contain any nullable columns.

### **Indexes**

Indexes are your specific definition of key fields or columns and the order of those fields or columns within the complete key. During performance analysis, the OS/400 query optimizer may issue a job log message that recommends a new index be created to improve performance. You can use SQL CREATE INDEX or this tab dialogue to create a new index.

The Indexes tab enables you to add modify, view, or delete an index for the table with which you are currently working. You may modify an index only if it was defined during your current table editing session. If you added the index and then clicked OK on either the New Table dialog or Table Properties dialog, you can only view the index.

### **Referential constraints**

A referential constraint is where one or more columns of a table refer to values of columns in the table you are currently working on or another table that is referred to as the *parent* table for the current table.

The Referential Constraints tab enables you to add, modify, view, or delete referential constraints for the table you are currently working on. You may modify a constraint only if it was defined during your current table editing session. If you added the constraint and then clicked OK on either the New Table dialog or Table Properties dialog, you may only view the constraint.

### **Triggers**

A trigger is a call to a program when the current application program accesses the table to Insert, Delete, or Update a row. For each of these I/O operations, you can specify to call a program before the operations and/or a program to call after the operation. The functions provided here correspond to the OS/400 Add Physical File Trigger (ADDPFTRG) and Remove Physical File Trigger (RMVPFTRG) commands.

You need to be cautious when using triggers. They offer powerful functions without knowledge of the current program, but they are called synchronously. If they do too much work before returning control to the original program, you may observe performance degradation.

**Check constraint**

A check constraint is specified at the field or column level. A check constraint examines the validity of the data in one or more of the columns in the same table.

The Check Constraints tab enables you to add, modify, view, or delete check constraints for the table you are currently working on. You may modify a constraint only if it was defined during your current table editing session. If you added the constraint and then clicked OK on either the New Table dialog or Table Properties dialog, you may only view the constraint.

**1.3.3.5 Database table constraints tips**

Constraints offer powerful system-provided DB2 UDB functions that need to be understood before you use them. In addition to the Operations Navigator graphical interface to constraints, OS/400 provides several commands to constraints support, such as the Add Physical File Constraint (ADD PF CST) and Remove Physical File Constraint (RMVPFTRG) commands.

You can access the full range of OS/400 constraints support by using the OS/400 Work with Physical File Constraints (WRKPF CST) command. For additional constraints information, refer to:

- Operations Navigator online help information.
- AS/400 Information Center (<http://www.iseries.ibm.com/infocenter>). You can use the search word *constraints*.
- Chapter 15, “Controlling the integrity of your database with constraints”, in *Database Programming*, SC41-5701.
- Online help for the OS/400 commands on constraints, accessed through the Work with Physical File Constraints (WRKPF CST) command.

**1.3.3.6 Managing journals and journal receivers**

As discussed in 1.3.2.4, “Create journal example” on page 22, a journal and its attached journal receiver record the changes and actions made to a table. Once you create a journal and its initial journal receiver, you can perform additional journal management by right-clicking either the journal or a journal receiver within a library.

Figure 22 shows the actions that are possible on an existing journal.

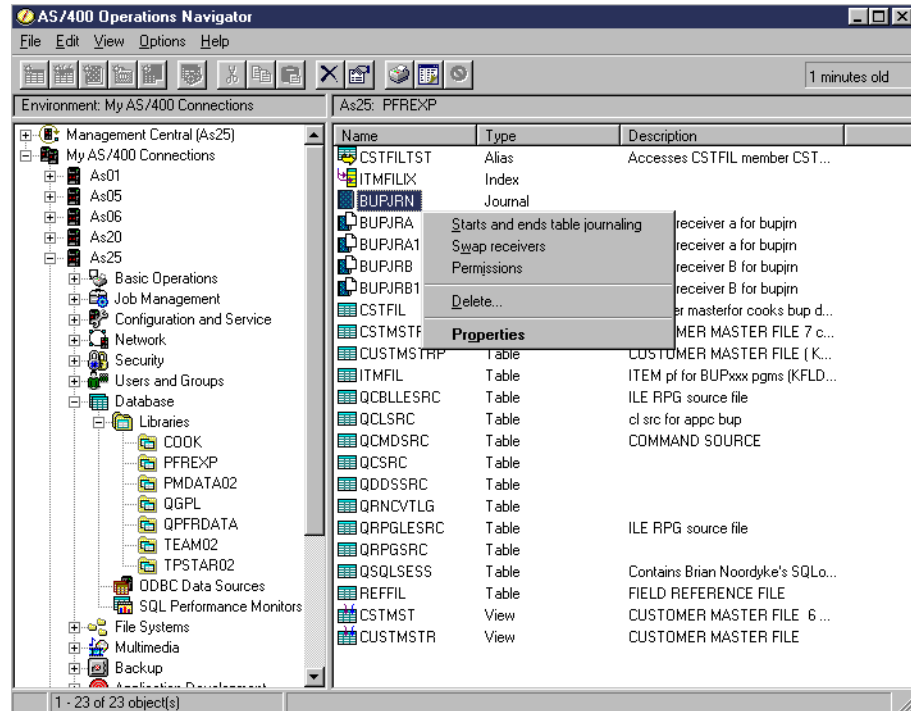


Figure 22. Managing a journal

The actions are explained in the following list:

- Starts and ends journaling:** This action starts or ends journaling for one or more specific files or tables. Clicking this action brings up the Start/End Journaling panel shown in Figure 23 on page 36.
 

The start and end functions correspond to the OS/400 Start Journaling Physical File (STRJRNPf) and End Journal Physical File Change (ENDJRNPf) commands.

Journaling can be started and ended from the item you obtain by right-clicking a table name.
- Swap receivers:** Clicking this action immediately detaches the currently attached journal receiver and creates a new journal receiver by adding 1 to a sequential number suffix to the journal receiver name. You can also manually swap receivers by using either an option from the Properties action or by using the OS/400 Change Journal (CHGJRN) command.
- Permissions:** This action lets you view and change the authorities to the journal as discussed, in general, in Chapter 10, "Permissions" of the redbook *Managing AS/400 V4R4 with Operations Navigator*.
- Delete:** Clicking this action brings up a confirmation window for completing the journal deletion request or canceling it.
- Properties:** This action brings up a panel that shows the original create journal attributes including journal receiver attributes and remote journal attributes, if any. You can also create a new journal receiver or remote journal by using the buttons that lead to additional panels. Figure 24 on page 37 shows an example of Journal properties information.

We right-clicked the BUPJRN journal and selected Starts and ends table journaling. Figure 23 shows the Start/End Journaling display for BUPJRN after we performed some journal-related operations earlier.

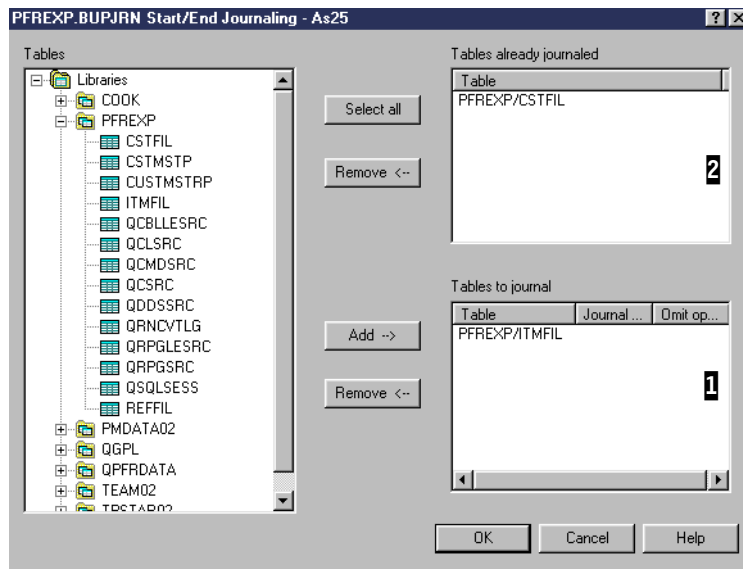


Figure 23. Start/End Journaling display

To start journaling for a file or table, you can select the table and either click the **Add** button or drag and drop the file name into the list box at **1**. When all tables you want journaled have been added to the list box, click the **OK** button. This starts journaling for these files or tables.

Alternatively, you could have used the OS/400 Start Journaling Physical File (STRJRNPF) command.

Notice the “Journal...” and the “Omit op...” column headings in the list box at **1**. The “Journal...” heading corresponds to the STRJRNPF command IMAGES (Record images) parameter. The “Omit op...” column heading corresponds to the STRJRNPF command OMTJRNE (Omit journal entries) parameter.

If you click under the “Journal” or “Omit op” heading to the right of a file or table name, an “X” character appears. If you click again, the “X” disappears. An “X” under Journal means that a journal entry, before a change to a row and after a change (both), is made to the receiver. If no “X” appears, only an *after image* is recorded in the journal entry. An “X” under Omit op... means that file or table open and close actions are not recorded in the receiver. If no “X” appears, all actions on the journaled file or table are recorded in the receiver.

In the list box at **2**, you see the PFREXP/CSTFIL (system naming convention) table is already being journaled at the time the Properties action was selected.

You can stop journaling for a file or table by selecting the file or table listed in **2** and clicking the **Remove** button and then clicking the **OK** button. This function corresponds to the OS/400 End Journaling Physical Files (ENDJRNPF) command.

### ***Journal Properties example***

When we right-clicked the BUPJRN journal and selected Properties, the Journal Properties panel appeared as shown in Figure 24. This shows the original parameters used to create the journal and enables you to make some changes and additions.

The Tables button shows you the Start or End journaling panel we already described.

The Receivers button shows you the currently attached receiver and previously detached journal receivers still on the system. You can also add a new journal receiver.

The Remote Journals button shows you the current status of a remote journal, if any. You can also add a new remote journal.

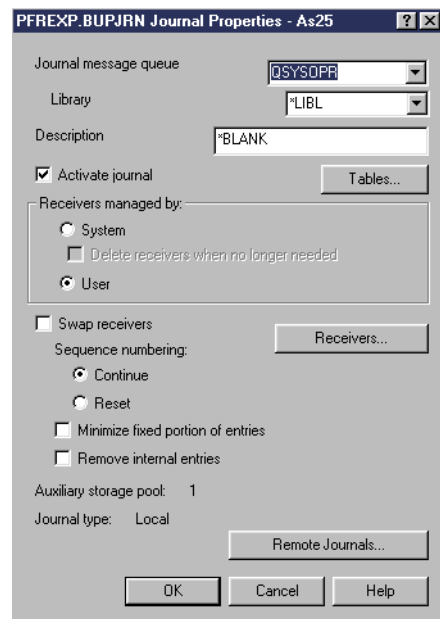


Figure 24. *Journal Properties example*

You can select the Swap receivers box and optionally specify either Continue or Reset to specify the sequence numbering to be used with the new receiver. Then click the **OK** button to have an immediate detach of the current journal receiver and creation of a new receiver that is immediately attached to the journal. Review online help information (place the window's ? symbol on the Swap receivers text) to determine if Swap receivers applies to your journaling environment.

In this example, we clicked the Receivers button to show you the panel in Figure 25 on page 38. In our example, we have three online, but detached, receivers. The currently attached receiver is BUPJRA0002.

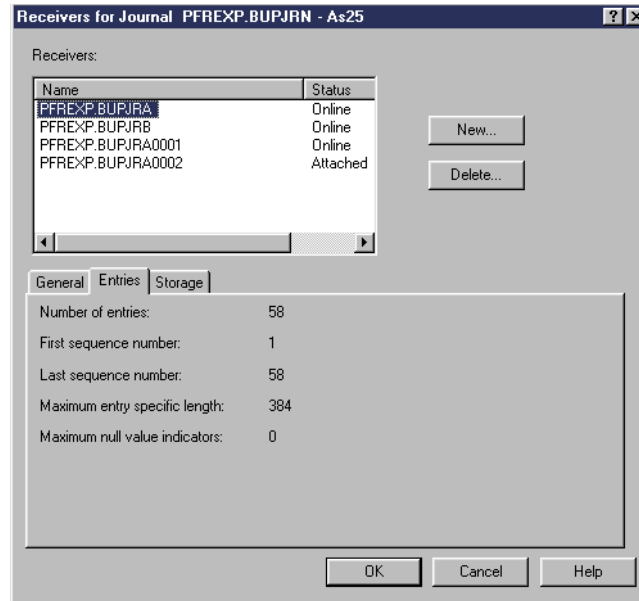


Figure 25. Journal receivers list and properties example

By selecting a journal receiver (BUPJRA in our example), the lower portion of the panel automatically displays the General properties of this receiver. We have already selected the Entries tab information.

Select a journal receiver. Click the **Delete** button to remove the journal receiver and its entries from the system when you no longer need this journaled information.

When you click the **New** button, you see an *Add Journal Receiver* panel. Click the **OK** button makes any new or delete function permanent.

You may also find the journal receiver General, Entries, and Storage information in a separate Properties panel for a specific receiver by performing either of the following actions from the library panel (as shown in Figure 22 on page 35):

- Double-clicking the journal receiver name
- Right-clicking the journal receiver name and selecting **Properties**

### 1.3.3.7 Working on Locked Rows

One of the new features added to the Database component of Operations Navigator in V4R5 is the **Locked Rows** function. To gain access to this function, right-click a table.

The Locked Rows dialog (Figure 26) displays the row number, job, user, job number, current user, status, and lock type for rows that have a row lock placed on them. A row lock is placed on a row when you read a table that is opened for update. While the row lock is in effect, no other job can read the same row for update, which keeps another job from unintentionally deleting the first job's update.

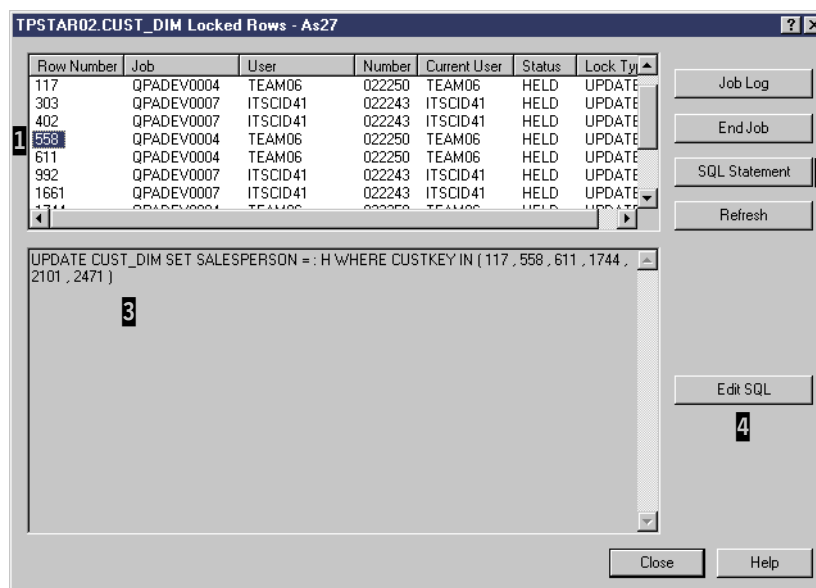


Figure 26. Locked Rows

The Locked Rows panel allows you to perform various tasks:

- Check which jobs are locking which rows
- View the job log for a job
- View an SQL statement that is running or has run
- Use the above mentioned SQL Statement with the Run SQL Script center
- End a job that is listed (provided you have the right authority)

Since most of these tasks are rather intuitive, we only document how to link to the SQL Script center to investigate on what is happening in the database.

After you start the Locked Rows function, select the job **1** you want to examine (as shown in Figure 26). Click the **SQL Statement** button on the right-hand side of the picture at **2** to bring it into the bottom part of the panel as shown at **3**.

At this point, when you click the **Edit SQL** button **4**, the Run SQL Script center starts and the SQL statement is brought into it for you to use. Refer to 1.5.3.1, “Running a single SQL statement” on page 54, for a discussion on how to use this tool and to 1.5.3.5, “Linking to the Visual Explain component” on page 56, to see how to use it to conduct database performance analysis.

## 1.4 ODBC Data Sources overview

Open Database Connectivity (ODBC) is a standard interface for database connectivity defined by the Microsoft Corporation. ODBC establishes the standard interface to any database as Structured Query Language (SQL). In general, the ODBC architecture accounts for an application using the ODBC interface, an ODBC Driver Manager, one or more ODBC Drivers, and an ODBC Data Source (place where the data is stored).

Client Access Express for AS/400 provides the AS/400 ODBC Driver that runs on the client workstation and the ODBC Data Source support that runs on the AS/400 database server - (production mode job name starts with QZDASOINIT (or

QZDASSINIT if SSL is being used) as described in 5.5.3, “Client Access servers” in the redbook *Managing AS/400 V4R4 with Operations Navigator*.

With ODBC Data Sources, you can set up a Client Access Express ODBC data source by providing a data source name (a name meaningful to you) and an AS/400 system name. An ODBC data source consists of the data that the user wants to access and its associated operating system, Database Management System (DBMS), and network platform (if any) used to access the DBMS.

Setup information is associated with a data source, and may include, for example, data formatting and performance options. Data formatting options include qualified name separators, date and time formats, and data translation. Performance options include when to use record blocking, data compression or an SQL Package. An SQL package stores previously parsed SQL statements to improve performance when used later.

You can also specify if Secure Socket Layer (SSL) is to be used with the ODBC connection. For more on SSL support, see 7.1.9, “Using Secure Sockets Layer (SSL) with Operations Navigator” in the redbook *Managing AS/400 V4R4 with Operations Navigator*.

Some client applications (including Operations Navigator) may provide their own unique data source definition.

A good source for more information on AS/400 ODBC support is *AS/400 Client Access Express for Windows ODBC User's Guide V4*, SC41-5509.

### 1.4.1 IBM-provided ODBC Data Sources

With Operations Navigator, you can create your own data source to limit the libraries that can be used and, as previously described, your own set of name separators, date and time formats, performance options, and so on.

OS/400 provides two data sources that you should understand even if you are not creating your own data source:

- A data source used by Operations Navigator itself to perform its functions. This data source is identified by the system name to which you are connected. For example, if your system name is As25, the data source used by Operations Navigator is named QSDN\_As25.

**Note:** Unless you are an ODBC expert, do not change any of the default settings for this data source. If you change them, Operations Navigator may fail to operate correctly.

- Data source used if you use Database-> Run SQL Scripts. The first time you select the action to Run SQL Scripts to a specific AS/400 system, OS/400 creates a data source named, in our example, QDS2\_As25. You do not have to create you own ODBC Data Source and understand the data source parameters to run SQL statements against libraries and files or tables to which you are authorized.

In 1.5, “Run SQL Script” on page 45, we use this IBM-created data source in our examples.



## 1.4.2 ODBC Data Source setup parameters

To create your own ODBC data source, right-click **ODBC Data Sources** (refer to Figure 1 on page 4), and select **New Data Source**. You are presented with a set of panels where you can name the ODBC data source, the name of the associated system and all the setup parameters.

**Note:** We do not provide examples of configuring your own ODBC data sources in this book.

In this section, we show you ODBC data source setup parameters used by Operations Navigator for the Run SQL Scripts function. You can create your own ODBC data source using these parameters as a reference.

To display or modify ODBC data source QDS2\_system-name, select **Database->Run SQL Scripts**. From the Run SQL Scripts menu bar, select **Connections->ODBC Setup** to access a panel similar to the example shown in Figure 27.

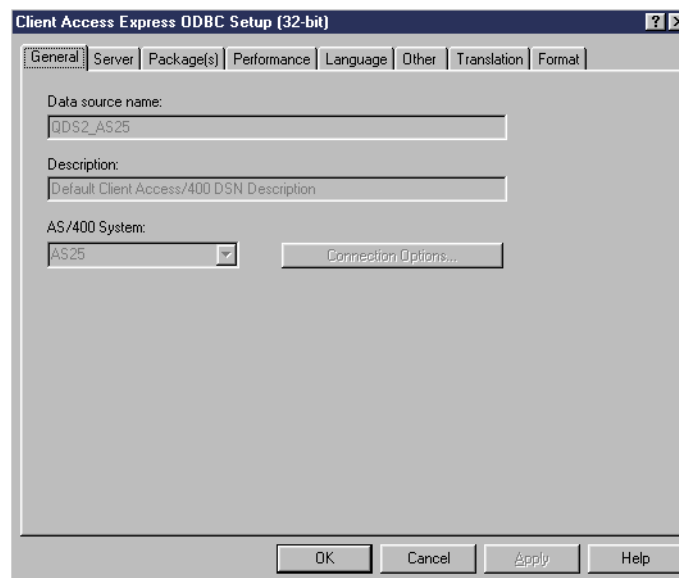


Figure 27. ODBC Data Source: Run SQL Scripts General tab

Here you see the IBM provided ODBC data source name and description and the attached system name. These values (QDS2\_AS25, ... AS25 in this example) are grayed out because this system-supplied ODBC data source for Run SQL Scripts name cannot be modified. We do not show you all the setup parameter values that are possible for an ODBC data source. However, we discuss the Server, Translation, and Format tabs parameters in the following sections.

### 1.4.2.1 ODBC Data Source Server parameters

The Server set of parameters is important in understanding the impact of some of the SQL functions you perform on the data on the system to which you are connected. Figure 28 on page 42 shows the Server set of parameters. We also include the help window for the Commit modes parameter, because of the part it plays in transaction integrity.

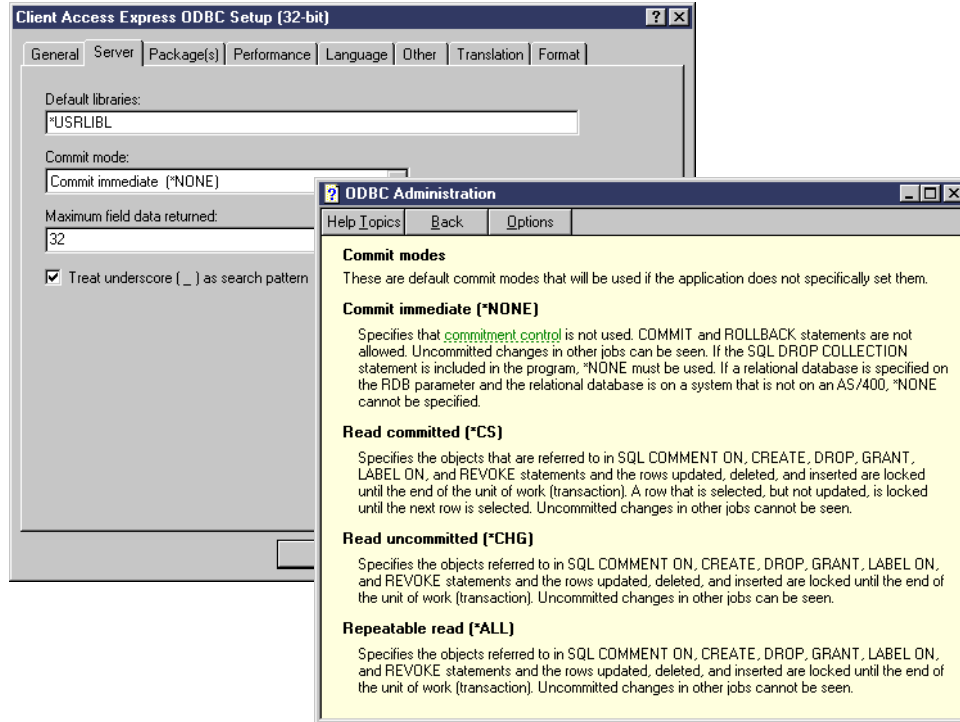


Figure 28. ODBC data source: Run SQL Scripts Server tab

Default libraries enables you to change the set of libraries available to the user of this ODBC data sources. The default (\*USRLIBL) means to use the initial library list (INLLIBL) parameter specified on the job description for the OS/400 user profile using this ODBC data source.

Commit mode controls the level of DB2 UDB for AS/400 commitment control, including when database changes are considered permanent and whether other users of the same database rows can see column updates that are not yet permanent.

A complete description of commitment control is beyond the scope of this redbook. However, you should understand that in the industry, users of SQL typically expect commitment control to be active. That is, an application design determines what a completed transaction (also called a unit of work) is. Any database row changes (column updates, rows deleted, rows inserted) are not considered permanent until a successful transaction has been completed (transaction boundary). At that time, the application performs a *commit* and all changes are now made permanent. If the application determines that an in-progress transaction should be terminated, it performs a *rollback*. All changes are as if they had never occurred. If the application abnormally terminates before issuing a commit or rollback, the underlying SQL support performs the rollback.

To support commitment control on OS/400, you must also have the tables journaled, and the job using these tables must issue a system operation that starts commitment control for the job. This system operation can be invoked by using the OS/400 Start Commitment Control (STRCMCTCTL) command or be implicitly invoked by this parameter for values other than \*NONE.

A *commit group* refers to the rows that are in the process of being updated, deleted, or inserted. As the help text shows, objects referred to on the COMMENT ON, CREATE, and so forth are also part of this commit group. The commit or rollback applies to all of these rows and objects.

We include the help text here because the OS/400 default is \*NONE, which is not generally supported in the industry. This provides a flexible operating environment, such as letting other applications or users access the latest database changes. However, \*NONE exposes the table rows, even while being processed by the properly authorized Operations Navigator user, to be modified without a required database Commit or Rollback operation sequence to make any database changes permanent.

For example, using \*NONE means any valid SQL statement that changes column data has made a permanent change to the data. If the properly authorized Operations Navigator user mistakenly updates a column using a wrong value for a key, there is no rollback function available to undo the change to the wrong row. You need either a backup copy of the data or an OS/400 journal to recover the original data.

The other commit values specify row locking rules (other applications prevented from updating the same row) and visibility of in-progress changes among applications accessing the same rows.

Maximum field data returned (in kilobytes) specifies the internal buffer size to allocate for a single transfer of data between your client workstation and the AS/400 system. If more data than the value specified is to be exchanged, another I/O operation between the AS/400 system and your workstation must be performed to exchange all of the data. The default (32K) is normally the best buffer size for best performance. If you change this size to a significantly smaller value, you may degrade performance when exchanging large amounts of data, because multiple transmissions are required.

The Search Pattern option, when selected, allows the underscore character (\_) in the AS/400 library and table names to be treated as search patterns (wildcards). When it is not selected, the underscore character is treated as a character. For example, the search pattern is not selected, and LIB\_A is treated as it is. When it is selected, LIB\_A would include LIB1A or LIB2A and so on. Using an underscore character in an SQL is similar to using a question mark in PC DOS as a single position wildcard, for example: `dir LIB?A /s`.

#### **1.4.2.2 ODBC Data Source Format parameters**

Format parameters are important if you have a special operating environment, such as your system requiring country-specific or multiple-country support. Figure 29 on page 44 shows you the ODBC data source Format parameters available to you through Operations Navigator.

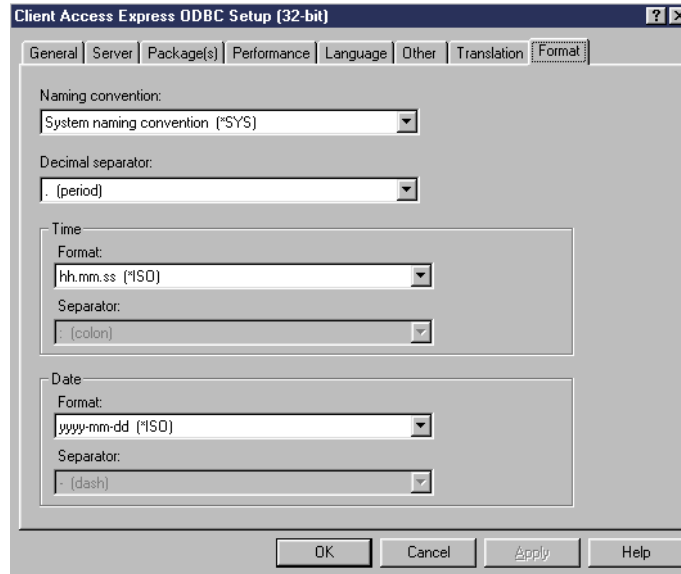


Figure 29. ODBC data source: Run SQL Scripts Format tab

You must review the online help text to get the details for all of these parameters. These settings shown can be modified. The settings are determined by your requirements.

We include the Format settings here because we do not use the default *SQL naming convention* syntax (includes the period (.) character as a name separator) in some of our examples in this chapter. We use the (AS/400) *system naming convention*. This means the forward slash character (/) may be used for this session with As25. This is the normal OS/400 naming convention.

**Note:** There is an important operational difference between using the SQL naming convention and the system naming convention when running SQL statements under Operations Navigator Run SQL Scripts. If you are using the system naming convention and use a non-qualified name, such as a table name with no library qualifier, the system searches for the table within all libraries currently in the session's (job's) current library list. If you are using the SQL naming convention, the ANSI standard specification causes the system to look *only in the current library* within the session's current library list.

For example, assume the user portion of the session's library list is in the order of TEAM02, followed by library TPSTAR02. Also, assume the unqualified table name is CUST\_DIM and is stored in library TPSTAR02. Using the SQL naming convention, the system looks for CUST\_DIM only in library TEAM02 and does not find it, which results in an error condition. Using the System naming convention, the system first searches library TEAM02 and then library TPSTAR02. The CUST\_DIM table will be found and the SQL statement will run successfully.

#### 1.4.2.3 ODBC Data Source Translation parameters

In most cases, you never need to view or change the ODBC data source Translation parameters. This is because your application tables or files are typically stored as using the Coded Character Set Identifier (CCSID) numeric value that stores the data according to your national language encoding. In these

cases, any OS/400 data accessed by the client workstation is translated into the appropriate ASCII format as required for viewing or processing on the client.

However, certain OS/400 system files or tables are defined to use the special CCSID 65535. By default, ODBC data source processing does not translate data from a file or table with CCSID 65535.

For example, if you want to use Run SQL Scripts against the performance collection files (prefix QAPM...) or a table generated from a Virtual Private Network (VPN) journal (copied to a database file or table), you need to have the character columns translated in most cases. Select the ODBC data source **Translate** tab, and select the **Translate CCSID 65535** check box.

For more information on CCSID support, refer to *AS/400 National Language Support*, SC41-5101.

## 1.5 Run SQL Script

The Run SQL Script center is a powerful interface to your AS/400 database. With it, you can use any SQL statements to issue any kind of operations you are authorized to on the AS/400 database objects. Licensed product program 5769-ST1 DB2 Query Manager and SQL Development Kit for AS/400 is not a prerequisite to it. This component of Operations Navigator uses ODBC to access the server. It uses a data source (QDS2\_*yoursystemname*), which is created when you install the database component of Operations Navigator.

To use SQL from Operations Navigator, right-click the **Database** component under the AS/400 system that contains the data. Figure 30 shows the Database context menu with the Run SQL Scripts action highlighted.

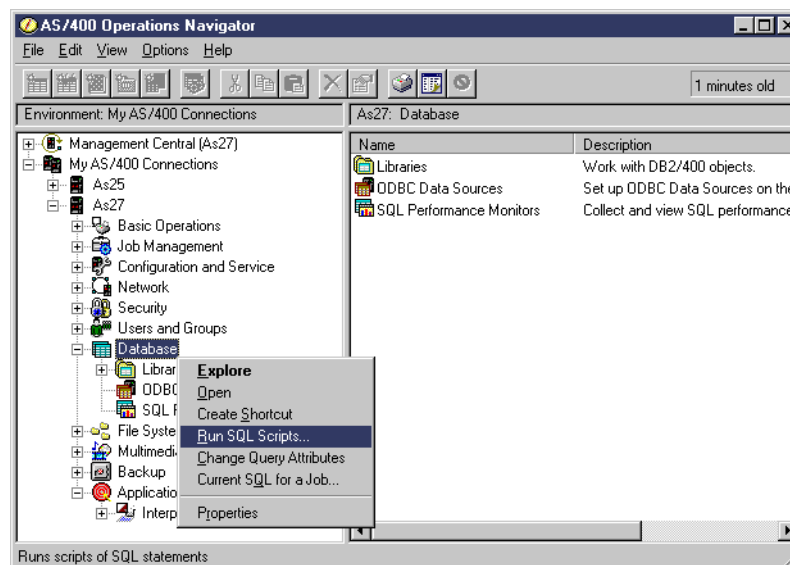


Figure 30. Run SQL Script

Right-click **Database** to bring up the pull-down menu. *Do not* click Libraries, because Operations Navigator enables you to potentially access the entire system, rather than limiting you to just the data within a library.

Figure 31 shows an example of the initial Run SQL Scripts panel.

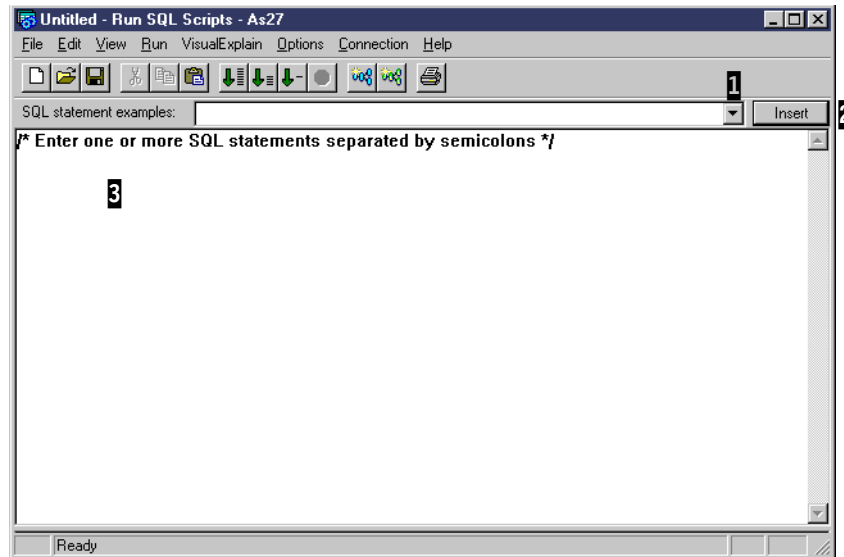


Figure 31. Run SQL scripts: Initial input panel

The Run SQL Scripts window lets you create, edit, run, and troubleshoot scripts of SQL statements. You can also save the SQL scripts with which you work into a PC file on your PC workstation. There are several run options for the SQL statements that are entered into the SQL statement input area at **3**. We discuss them later in this section.

As shown at **1** in Figure 31, you can select to review a list of already provided SQL statements. OS/400 provides a large set of base syntax for almost every possible SQL statement that can be used. You can display the list of existing SQL statements by clicking the down arrow in this area of the panel. You can then select an SQL statement from the list shown and have it *inserted* into the statement input area at **3** by clicking the Insert button **2**.

You can modify the selected SQL statement or enter your own SQL statement. You can run one or more of your entered your SQL statements in different ways and stop between statements.

Before we discuss the run actions, we show Figure 32 to illustrate the different panels within the Run SQL Scripts function.

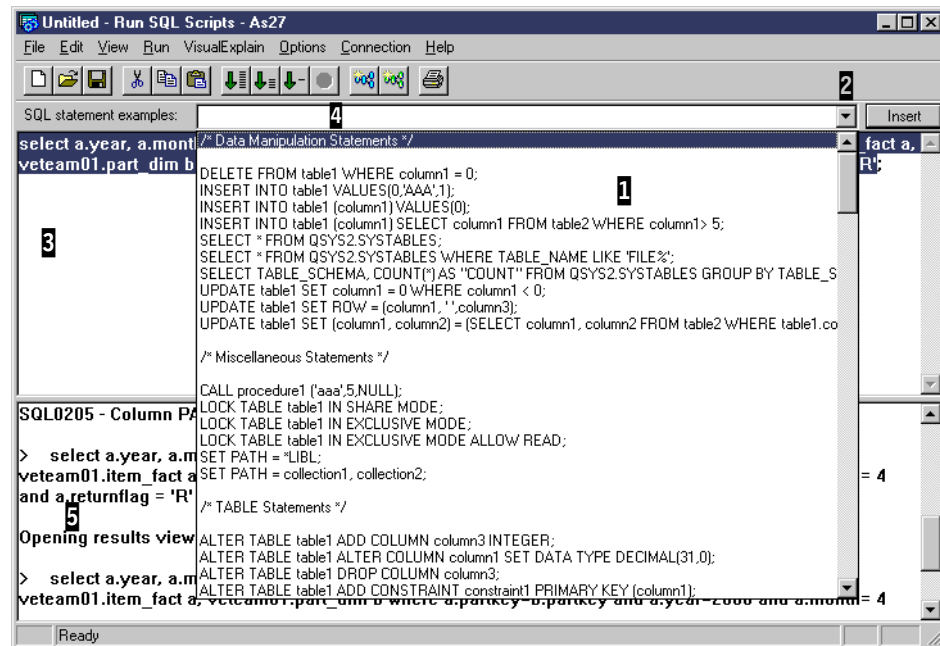


Figure 32. Run SQL Scripts window pane example

The beginning of the list of provided SQL statements is shown at **1**. This list was produced by clicking the down arrow shown at **2**. In this example, we do not select an SQL statement to be placed into the statement input area shown at **3**. However, if we selected one or more SQL statements in the window at **1**, the statement or statements would appear in the “SQL statement example” area at **4**, and you could click the Insert button to place the statements into SQL input area.

In the SQL statement input area at **3**, we already entered two simple SQL statements that are partially hidden. We separately ran the SQL statements, `select * from item_fact;` and `select * from cust_dim;` and viewed the results on a panel not shown, prior to selecting the list of SQL statements shown at **1**.

The Run History panel at **5** shows you the results of the SQL statements run. You see an SQL error (SQL0104) message because we erroneously ended the SQL statement with a colon (:) instead of a semi-colon (;). In this example, the pane showing the SQL statements list hides the `select * from dim_table: error`.

When you select the Edit option from the menu bar, you have the option to clear the *run history* information.

In Figure 33 on page 48, we include the previous SQL SELECT statements, but added SQL statements to illustrate more of the power of DB2 UDB for AS/400 accessible through Operations Navigator.

We also use Figure 33 to illustrate some of the *run options* for the SQL statements we showed under Run SQL Scripts support.

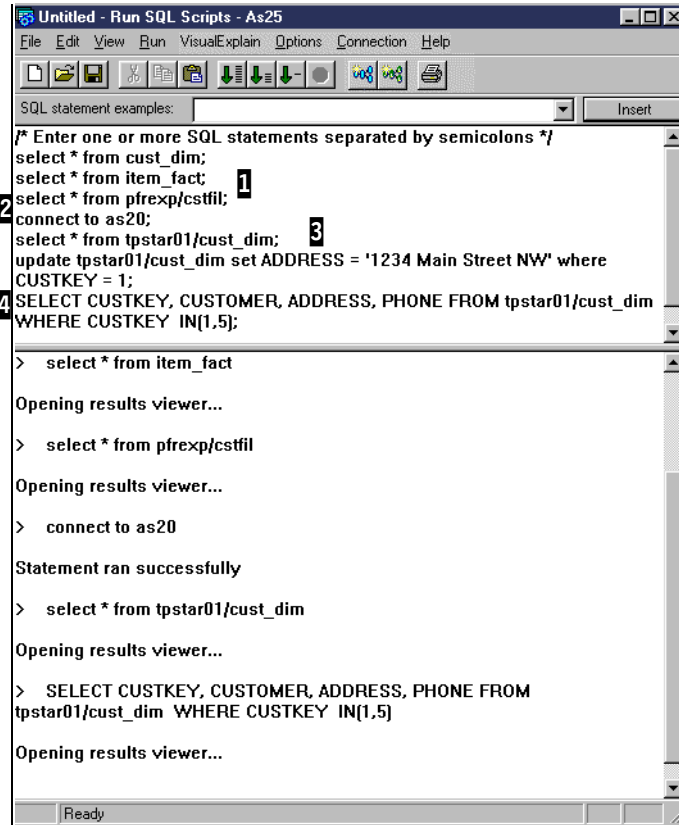


Figure 33. Run SQL scripts: Additional sources

At 1 in Figure 33, we show that we selected from a table that is in a different OS/400 library (pfxexp) than the libraries included in our job description's initial library list (refer to Figure 3 on page 9). We did this by qualifying the table with pfxexp/.

The slash separator character (/) is valid because, in Figure 29 on page 44, we showed that we changed from the default SQL naming convention to the system naming convention.

By default, we are running SQL statements on the system to which we are connected. The SQL CONNECT statement used to connect to a remote system as20 ("As20" in our Operations Navigator screen example figures), using OS/400 Distributed Relational Database Architecture (DRDA) over TCP/IP is shown at 2. Assuming this CONNECT statement is successful, from the CONNECT statement onward, all SQL statements are directed to remote system as20 until an SQL "release all" statement is issued, when the connection returns to access only the local As25 system.

OS/400 supports connections to multiple remote systems during the same session. For example, following the statement shown at 4, you can issue a "connect to as05" statement. Assuming this is successful, all the following SQL statements are directed to system As05. You can then issue a "set connection to as20" statement that resets the current dialogue back to system As20. You need to keep track of which system (remote database) you are connected to and on which system you are performing operations.



The next statement at [3](#) selects the cust\_dim table in library tpstar01 on remote system as20.

**Note:** DRDA is the IBM-defined architecture for accessing remote databases. It is implemented on all IBM operating systems and some non-IBM operating system database support. At a base set of functions level, it is similar to the ODBC and Java Database Connectivity (JDBC) set of capabilities. On IBM systems, Distributed Data Management (DDM) is a higher level interface to DRDA capabilities.

While we cannot go into DDM/DRDA details in this book, we discuss basic setup requirements for the DDM/DRDA example shown here to work over TCP/IP. Refer to 1.6, “Change Query Attributes” on page 60, for more information.

A select statement that uses only some of the fields or columns in the cust\_dim table and displays only the records or rows where the key field or the CUSTKEY column has a value of 1 or a value of 5 is shown at [4](#).

**Note:** With our examples, each table index (set of key fields or columns) structure is relatively simple, and the number of rows is relative to a million or more rows that would be present in a data warehouse environment. We also do not have complex join statements (columns joined together from two or more tables). In a more complex data structure and performance critical environments, you would want to use a combination of the following options:

- The Run SQL Scripts option to include query optimizer debug messages in the job log (see 1.5.3, “Run SQL Scripts run options” on page 53)
- SQL Performance Monitor support (see 1.8, “SQL Performance Monitors overview” on page 64)
- Visual Explain (see 1.9, “Visual Explain” on page 85)

By reviewing the job log, using Visual Explain or going through monitored data, you can determine if the most efficient method is used by OS/400 query support to perform the SQL function.

### 1.5.1 Running a CL command under SQL script

In addition to running SQL statements under Run SQL Script, Operations Navigator allows the properly authorized user to run any OS/400 Control Language (CL) statement that can be validly run in a batch (no 5250 workstation required) environment. You must precede the OS/400 command syntax with the prefix `CL:` (uppercase or lowercase) as shown in Figure 34 on page 50.

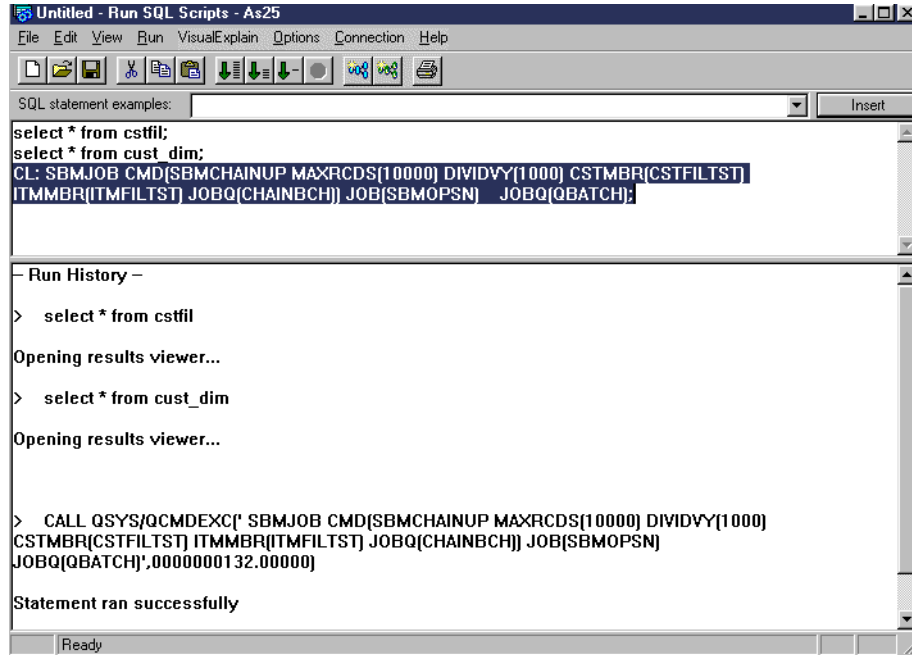


Figure 34. Run SQL Scripts: Running a CL command

The selected CL command is an OS/400 Submit Job (SBMJOB) command that submits the job to job queue QBATCH, which is one of the IBM-supplied job queues associated with the IBM-provided subsystem QBATCH. The submit job command parameter (CMD) value can be any OS/400 command or user-defined command. In our example, we use a user-defined command, SBMCHAINUP, with its own set of parameters.

Our example runs in the QBATCH subsystem. However, user programming immediately causes another submit job to a different job queue JOBQ(CHAINBCH).

You may also use much simpler OS/400 commands, such as:

- Adding a new library to the current library list of the Operations Navigator session using the CL command:

```
ADDLIB LIB(PFREXP)
```

- Sending a message to the system operator using the CL command:

```
SNDMSG MSG('This message is from an Operations Navigator session from user TEAM02.') TOUSR(*SYSOPR)
```

**Tips for running CL in Run SQL Scripts**

Running SQL Scripts is a powerful way to test new SQL statements, especially in the sequence you may want to run them in a program. In an actual application environment, you may also want to integrate running system functions through CL commands with your SQL statements. Here are some tips:

- Starting with Client Access Express Service Pack 5 (SP5) for V4R4 the following restriction has been removed:

For the CL command to be recognized successfully, you must remove (delete) any comment statement, such as:

```
"/* Enter one or more SQL statements separated by semicolons */."
```

- The IBM-supplied SQL statement examples include some CL command examples at the end of the SQL statements.
- The key to making the OS/400 command work from an Operations Navigator Run SQL Scripts session is to ensure the objects referenced in the command can be found in the Operations Navigator session's (job's) library list or the system library list (system value QSYSLIBL).

Adding a library name under the Database->Libraries branch does not carry over to the Run SQL Scripts function. OS/400 commands can always be found through the system value QSYSLIBL. However, user objects, such as the SBMCHAINUP command, require the appropriate library to be in the Operations Navigator Run SQL Scripts session's library list.

**1.5.2 Run SQL Scripts example using a VPN journal**

This section shows an example of using Run SQL Script to identify the IP packets, if any, that were *denied* routing based on OS/400 Virtual Private Networking (VPN) filtering rules. The standard OS/400 VPN support records *permit*, *deny*, and *filter rule* change occurrences in a system journal named QIPFILTER, stored in library QSYS. The OS/400 Display Journal (DSPJRN) command provides a journal entry time stamp and other compare values to selectively display, print, or copy to a database file or table.

If you choose the database option, you can process the copied data several different ways through SQL. Run SQL Scripts is a good way to experiment with viewing different journal entry field or column data. Once you see a view of the data you want to use repetitively, you can save the SQL statements for later reuse or copy the SQL statements into a program that does further processing or graphical display.

In this section, we use the journal data discussed in the "AS/400 VPN problem determination," chapter in *AS/400 Internet Security: Implementing AS/400 Virtual Private Networks*, SG24-5404.

We performed the following steps to query the VPN logging data originally placed into the QIPFILTER journal. The query results show the journal entries for packets that have been denied routing, since a large number of *deny entries* may require further investigation by your security personnel.

The steps are:

1. Create a copy of the IBM-supplied file QSYS/QATOFIPF into a library of your choice, using the OS/400 Create Duplicate Object (CRTDUPOBJ) command, for example:

```
CRTDUPOBJ OBJ(QATOFIPF) FROMLIB(QSYS) OBJTYPE(*FILE) + TOLIB(mylib)
NEWOBJ(myfile)
```

The system file or table QATOFIPF provides the column definitions used by the IBM-supplied queries. In our example, we duplicate this table as OP\_IPFTRT.

2. Use the DSPJRN command to copy the journal entries from the QUSRSYS/QIPFILTER journal to the output database file created in the preceding step:

```
DSPJRN JRN(QIPFILTER) JRNCODE(M) ENTYP(TF) OUTPUT(*OUTFILE) +
OUTFILFMT(*TYPE4) OUTFILE(mylib/myfile) ENTDTALEN(*CALC)
```

The DSPJRN command has both starting and ending time-stamp values and starting and ending journal entry sequence numbers so you do not need to copy the entire set of journal entries to the file or table.

3. You need to review the field or column names and descriptions for file or table ON\_IPFTRT to determine which columns to select and use for row selection. You may use the OS/400 Display File Field Description (DSPFFD) command or use Operations Navigator to display the table Properties by right-clicking the table name.

The redbook *AS/400 Internet Security: Implementing AS/400 Virtual Private Networks*, SG24-5404, provides good background information to help select the appropriate fields or columns.

4. Using Run SQL Scripts, build the SQL statement and view the results.

Figure 35 shows our example SQL statement and sample output.

The screenshot shows a window titled 'Untitled - Run SQL Scripts - As05'. The main text area contains the following SQL query:

```
select ADAN.ON_IPFTRT.TFRNUM, ADAN.ON_IPFTRT.TFFACT, ADAN.ON_IPFTRT.TFPDIR,
ADAN.ON_IPFTRT.TFSRCA, ADAN.ON_IPFTRT.TFSRCP, ADAN.ON_IPFTRT.TFDSTA,
ADAN.ON_IPFTRT.TFDSTP, ADAN.ON_IPFTRT.TFTIME FROM ADAN.ON_IPFTRT where
ADAN.ON_IPFTRT.TFFACT='DENY ' and ADAN.ON_IPFTRT.TFSEQN >= 6300;
```

Callout 1 points to the 'DENY ' value in the WHERE clause. Callout 2 points to the '>= 6300' value in the WHERE clause. Below the query is a 'Run History' section showing the same query was executed. At the bottom, a table displays the results of the query:

	TFRNUM	TFFACT	TFPDIR	TFSRCA	TFSRCP	TFDSTA	TFDSTP	TFTIME
1	8	DENY	O	10.196.8.5	137	10.196.8.255	137	1999-07-24 13:32:52.878560
2	8	DENY	O	204.146.18.5	137	204.146.18.255	137	1999-07-24 13:32:52.972128
3	8	DENY	O	10.196.8.5	137	10.196.8.255	137	1999-07-24 13:32:53.082448
4	8	DENY	O	204.146.18.5	137	204.146.18.255	137	1999-07-24 13:32:53.169424
5	8	DENY	O	10.196.8.5	137	10.196.8.255	137	1999-07-24 13:32:53.582032
6	8	DENY	O	204.146.18.5	137	204.146.18.255	137	1999-07-24 13:32:53.653728
7	8	DENY	O	10.196.8.5	137	10.196.8.255	137	1999-07-24 13:32:54.085984
8	8	DENY	O	204.146.18.5	137	204.146.18.255	137	1999-07-24 13:32:54.175344
9	8	DENY	O	10.196.8.5	137	10.196.8.255	137	1999-07-24 13:32:54.409520
10	8	DENY	O	204.146.18.5	137	204.146.18.255	137	1999-07-24 13:32:54.459792
11	8	DENY	O	10.196.8.5	137	10.196.8.255	137	1999-07-24 13:32:54.935600
12	8	DENY	O	204.146.18.5	137	204.146.18.255	137	1999-07-24 13:32:54.989056

Callout 3 points to the source IP address '204.146.18.5' in the TFSRCA column. Callout 4 points to the destination IP address '204.146.18.255' in the TFDSTA column.

Figure 35. Run SQL Scripts: Viewing 'denied' VPN packets

Column TFACT (filter action), shown at 1, records values such as PERMIT, DENY, or additional values for adding and changing filter rules and starting and stopping filtering. At 1, you see our SQL compare value for DENY. At 2, you can see that we did not want to look at all (13,000) journal entries, so we started around the middle of the entries with sequence number 6300.

Column TFPDIR (packet direction) specifies "O" for output packet and "I" for input packet.

Using the source IP address and port number 3 and the destination IP address and port number 4, a TCP/IP expert can determine the actual workstation and TCP/IP function. A TCP/IP expert may also choose different columns to include in the SQL SELECT statement.

### 1.5.3 Run SQL Scripts run options

In the Run SQL Scripts examples section, we present different window panes within the Run SQL Scripts window with sample SQL and CL statements. In all of those examples, we ran only one statement at a time.

In this section, we explain the *run options* available for these SQL statements. We use Figure 36 on page 54 as a base for explaining the run options.

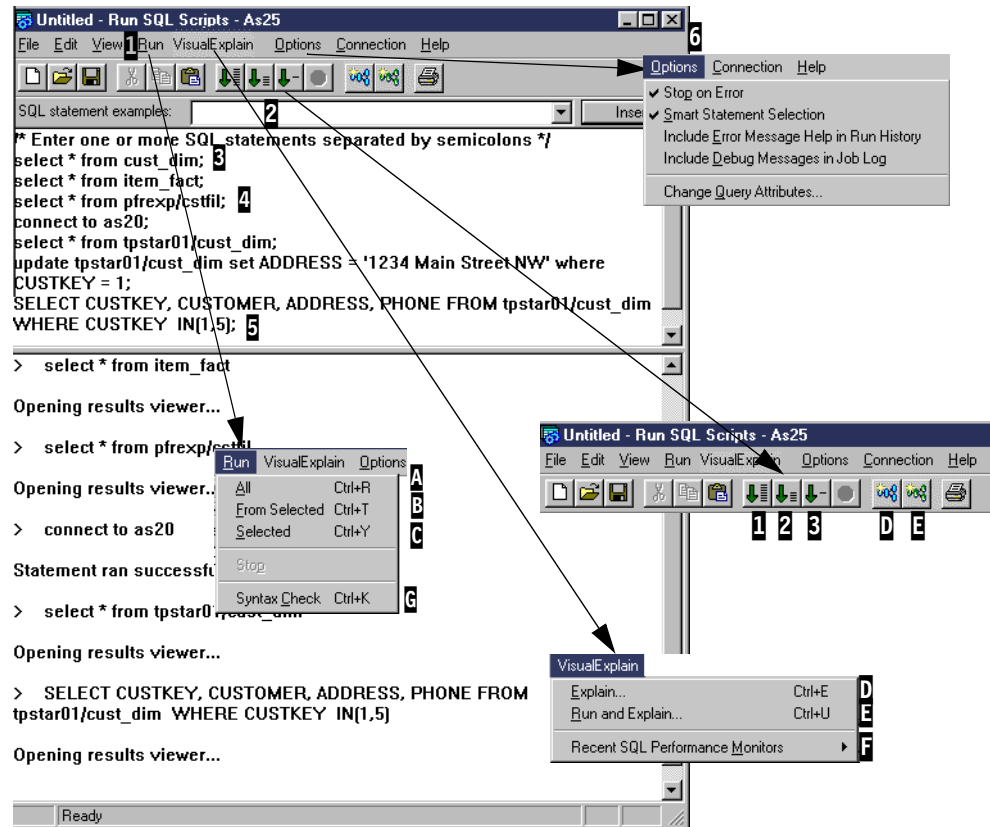


Figure 36. Run SQL Script: Run options

There are two “selection lists” types from which you can choose to run one or more SQL statements at a single time. You can select the Run option **1** from the Run SQL Scripts menu bar or select one of the *green arrow* run action icons **2** from the toolbar. These have corresponding functions. You can also select the Run option with a key sequence as shown under the Run pull-down menu.

You can pre-specify (defaults are provided) some controls over the Run function through the Options action in the menu bar **6**. We discuss these controls in 1.5.3.7, “Controlling SQL run options” on page 56, after we explain the three levels of run options:

- Run a single SQL statement
- Run a set of SQL statements
- Run all SQL statements currently specified

### 1.5.3.1 Running a single SQL statement

Place the active screen cursor within the SQL statement text you want to run, for example, `select * from pfrexp/cstfil`, referenced at **4** in Figure 36. You can run only this statement by using one of the following actions:

- Click the **Selected** action **C**.
- Click the “select one line” icon associated with **C** in our figure example.
- Press **Ctrl+Y** from the workstation keyboard.

Only the single statement will run. If it is a SELECT statement, the results are presented as a window on your Operations Navigator workstation. The column names are presented as column headings. If you want to select only a subset of

columns later, you can use these headings and displayed column data to help you select the appropriate columns. Figure 37 shows some of the column headings and associated data for `pfrexp/cstfil`.

	CNUM	CNAME	ADRES1	ADRES2	STATUS	TIMSTC
1	1	CUST NAME 000001	ADRS 1 000001	ADRS 2 000001	*	141138123099
2	2	CUST NAME 000002	ADRS 1 000002	ADRS 2 000002	*	141138123099
3	3	CUST NAME 000003	ADRS 1 000003	ADRS 2 000003	*	141138123099
4	4	CUST NAME 000004	ADRS 1 000004	ADRS 2 000004	*	141138123099
5	5	CUST NAME 000005	ADRS 1 000005	ADRS 2 000005	*	141138123099
6	6	CUST NAME 000006	ADRS 1 000006	ADRS 2 000006	*	141138123099
7	7	CUST NAME 000007	ADRS 1 000007	ADRS 2 000007	*	141138123099
8	8	CUST NAME 000008	ADRS 1 000008	ADRS 2 000008	*	141138123099
9	9	CUST NAME 000009	ADRS 1 000009	ADRS 2 000009	*	141138123099
10	10	CUST NAME 000010	ADRS 1 000010	ADRS 2 000010	*	141138123099
11	11	CUST NAME 000011	ADRS 1 000011	ADRS 2 000011	*	141138123099
12	12	CUST NAME 000012	ADRS 1 000012	ADRS 2 000012	*	141138123099
13	13	CUST NAME 000013	ADRS 1 000013	ADRS 2 000013	*	141138123099
14	14	CUST NAME 000014	ADRS 1 000014	ADRS 2 000014	*	141138123099
15	15	CUST NAME 000015	ADRS 1 000015	ADRS 2 000015	*	141138123099
16	16	CUST NAME 000016	ADRS 1 000016	ADRS 2 000016	*	141138123099
17	17	CUST NAME 000017	ADRS 1 000017	ADRS 2 000017	*	141138123099
18	18	CUST NAME 000018	ADRS 1 000018	ADRS 2 000018	*	141138123099
19	19	CUST NAME 000019	ADRS 1 000019	ADRS 2 000019	*	141138123099
20	20	CUST NAME 000020	ADRS 1 000020	ADRS 2 000020	*	141138123099
21	21	CUST NAME 000021	ADRS 1 000021	ADRS 2 000021	*	141138123099
22	22	CUST NAME 000022	ADRS 1 000022	ADRS 2 000022	*	141138123099
23	23	CUST NAME 000023	ADRS 1 000023	ADRS 2 000023	*	141138123099

Figure 37. Run SQL Script: Sample SQL SELECT output

### 1.5.3.2 Running a set of SQL statements

You can run a sequence of SQL statements that are currently active in your session to the AS/400 system. Using our example in Figure 36, you would run `select * from pfrexp/cstfil;` **4** through `SELECT CUSTKEY ... IN(1,5);` **5**

You do this by placing the active screen cursor within the SQL statement text at **4** and performing one of the following actions:

- Click the **From Selected** action **3**.
- Click the **From Selected** icon associated with **3** in our figure example.
- Press `Ctrl+T` from the workstation keyboard.

This runs each statement sequentially, beginning with `select * from pfrexp/cstfil`. We have three SELECT statements in our example. For each SELECT statement, a window of data is presented; all three windows are produced. However, if the SELECTs are fast enough, you may notice only the last SELECT output.

The three windows are active on the screen and can be viewed by selecting the appropriate task from the windows task bar, typically at the bottom of a window.

If an error occurs and a Stop on error option is selected (as specified under the Options pull-down menu shown at **6** in Figure 36, the program stops and the statement where the error occurred remains selected. The statement is ready to be run after it is corrected.

### 1.5.3.3 Running all SQL statements currently active

You can run sequentially all the SQL statements that are currently active in your session to the AS/400 system. Using our example, this would start with `select * from cust_dim;` at **3**, through `SELECT CUSTKEY ... IN(1,5);` at **5**.

You run all the SQL statements by doing one of the following tasks:

- Click the **All** action **A**.
- Click the **All** icon associated with **A** in our figure example.
- Press Ctl+R from the workstation keyboard.

If an error occurs and a Stop on error option is selected (as specified under the Options pull-down menu shown at **5**), the program stops, and the statement where the error occurred remains selected. When you use All, the Run History pane is cleared.

### 1.5.3.4 SQL statement syntax check

Using this option, **6** in Figure 36 on page 54, it is possible to validate a selected SQL statements or statements. This function performs a formal syntax check of the statement, while validating that the objects referenced (libraries, tables, columns) actually exist in the linked database. Resulting messages appear in the result panel. This option can also be invoked by pressing Ctrl+K after selecting an SQL statement.

### 1.5.3.5 Linking to the Visual Explain component

In V4R5, two more icons, **D** and **E**, in Figure 36 on page 54, have been added to the Run SQL Script tool bar. These icons provide access to the Visual Explain function, as do the two new menu items (**D** and **E**) under *Visual Explain*. For more information, refer to 1.9, “Visual Explain” on page 85.

The *Explain* option **D** (or CTRL+E) allows you to review the Optimizer access plan that will be used when executing an SQL statement; the statement is not actually run but optimized with the query attribute Time Limit set to 0. For details on query attributes, see 1.6, “Change Query Attributes” on page 60. It produces a visual explanation of the statement but will not access the actual data from the database, thus avoiding the unnecessary I/O overload.

The *Run and Explain* option **E** (CTRL+U) runs the SQL statement and gathers execution time statistics from the statement. It uses the actual access plan from the statement and the statistics and presents these in a graphical format. With this option, the statements executed before the analysis graph is reported.

### 1.5.3.6 Linking to the SQL Performance Monitor component

Using option **F** under the Visual Explain menu item in Figure 36 on page 54, you can obtain a list of the most recent SQL Performance Monitor collections and can then link into the tool to analyze collected data. See 1.8, “SQL Performance Monitors overview” on page 64, for a discussion on the characteristics and usage of this tool.

### 1.5.3.7 Controlling SQL run options

By selecting Options from the Run SQL Script menu bar (**G** in Figure 36 on page 54), you can control what to do if an SQL error occurs and what levels of additional information should be included in your session to the AS/400 system:



- **Stop on Error:** This turns stopping on or off when there is more than one SQL statement to run and an error occurs. If it is turned on (default), the SQL statements are stopped at the SQL statement in error, which remains selected. If it is turned off, all SQL statements continue to run until the end of the script has completed.
- **Smart Statement Selection:** This turns on or off treating the selected SQL statement as a complete statement or attempting to run only the selected text. If it is turned on (default), the complete statement, up to the ending semi-colon (;) character, is attempted. If it is turned off, only the selected text is attempted. If you do only a portion of the original statement, the statement may complete successfully. However, you are subject to at least two error conditions:
  - Omitting some text may make the SQL statement fail, because the statement is incomplete.
  - Omitting some text may still result in successful completion. However, if the ODBC data source used for your session is set to \*NONE for commitment control, omitting a phrase an UPDATE statement, such as WHERE CUSTKEY = 1, may update all the rows in the table, which is not what was intended.

See 1.4.2.1, “ODBC Data Source Server parameters” on page 41, for additional information about commitment control. The most complete OS/400 documentation on commitment control is in *Backup and Recovery*, SC41-5304.
- **Include Error Message Help in Run History:** This turns on or off (default) the including of additional error message information in the Run History pane when an error occurs. Figure 38 shows an example where we specified an invalid column name (WRONGCOL) for the table.

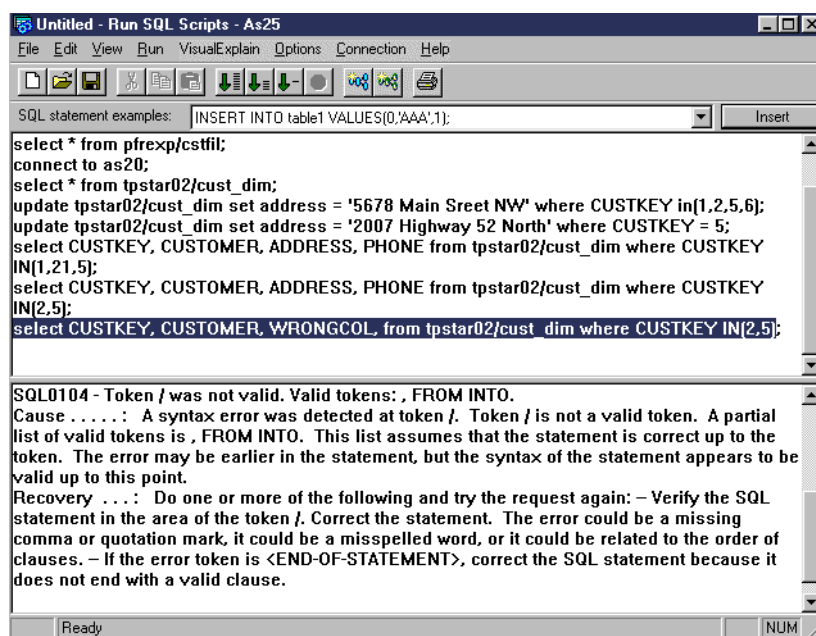


Figure 38. Run SQL Scripts: Include Error Message Help in Run History

- **Include Debug Messages in Job Log:** This option tells the OS/400 query optimizer support to record its decisions on how to process the SQL request,

including any recommendation for creating an index that may improve performance. The option is typically used only when debugging new and complex SQL statements or while analyzing a suspected performance problem.

Analyzing the job log messages may be sufficient to determine if a performance problem exists and what action should be taken to resolve the problem. You may also consider using the Operations Navigator interface to the SQL Performance Monitor, which is described in 1.8, “SQL Performance Monitors overview” on page 64, and Visual Explain, described in 1.9, “Visual Explain” on page 85.

Figure 39 shows an example of an SQL JOIN statement and the associated job log messages that should be reviewed.

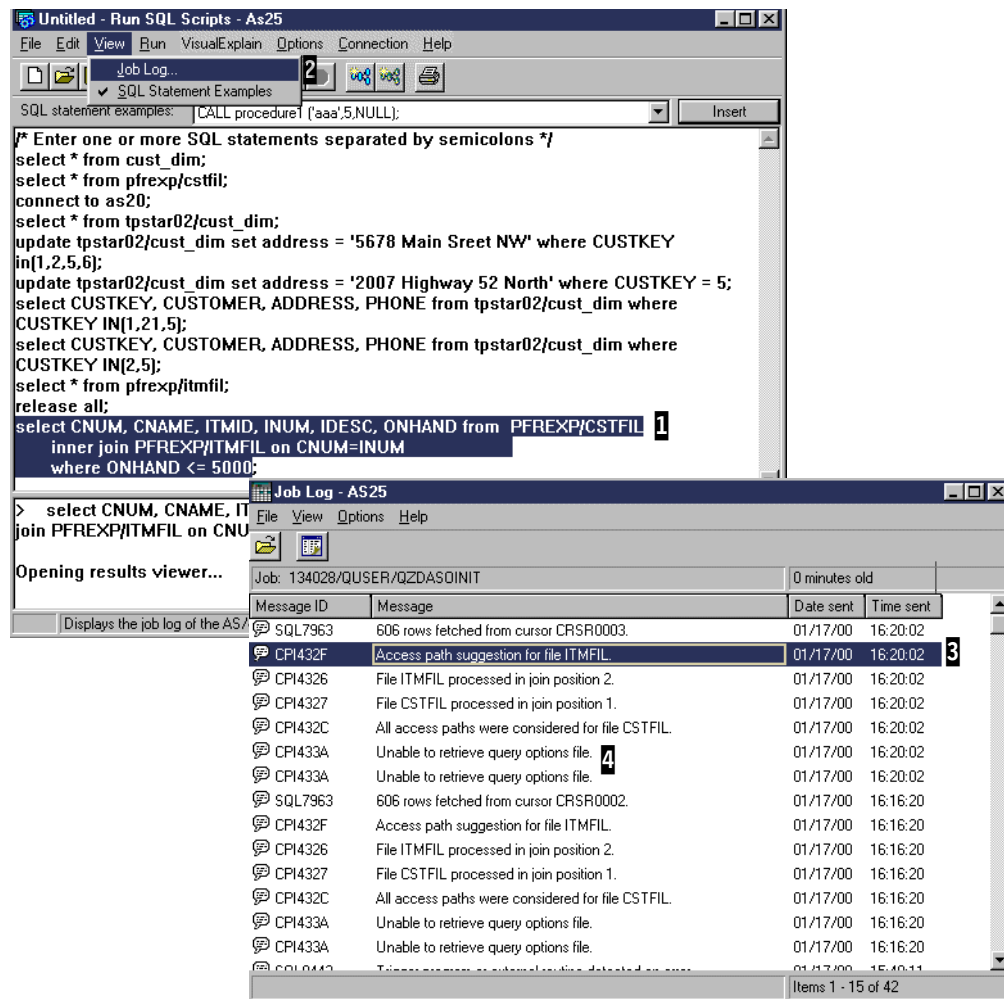


Figure 39. Run SQL Scripts: Job log debug messages example

We use the selected SQL SELECT with JOIN statement 1 to show the associated job log debug messages issued by the query optimizer. To see the current Operations Navigator session’s job log, complete these tasks:

1. Click **View** in the Run SQL Scripts panel.
2. Click **Job Log**.

In our example job log, we discuss two messages: the optimizer's suggestion for an access path (index) to file ITMFIL with message ID CPI432F [3](#) and error message CPI433A [4](#).

By double-clicking message CPI432F, the message details or "second level text" is displayed. The message text describes why the create index function is recommended and include the recommended column names to use in the new index.

Message CPI433A may appear multiple times in the job log of a job that has run several SQL statements. Each time an SQL statement is run, the system looks for a file or table by the name of QAQQINI in the QUSRSYS library. This table can be set up by you to specify query attributes that the OS/400 query optimizer will use while processing each SQL statement.

If you are not attempting to modify the default OS/400 query processing algorithm through this table, the table will not be in the QUSRSYS library, and this message is considered information only.

- **Change Query Attributes:** This allows you to easily modify the query options file QAQQINI for your job, provided you remember the job number previously checked in the job log, or for any other job in the system.

This is done using the same interface as documented in 1.6, "Change Query Attributes" on page 60.

#### 1.5.4 DDM/DRDA Run SQL Script configuration summary

Using Figure 33 on page 48, at [2](#), we showed and discussed an SQL CONNECT statement ("connect to as20;") to access data on a remote system. For ease of reference, this statement also appears in Figure 39. This section provides overview information on configuration parameters required to successfully access remote data.

Good sources for more information on DDM/DRDA over TCP/IP are:

- *Distributed Data Management*, SC41-5307
- *V4 TCP/IP for AS/400: More Cool Things Than Ever*, SG24-5190
- *DB2/400 Advanced Database Functions*, SG24-4249

For DRDA to work between a source system (function requester) and target system (request server) where the actual data is and the SQL function is performed, you need certain DRDA configuration set up correctly. The following steps summarize the configuration required (using As25 as the source or requester system and As20 as the target or server system).

On the As25 (source system), complete the following steps:

1. TCP/IP is started
2. Enter the OS/400 Add Relational Database Directory Entry (ADDRDBDIRE) command:

```
ADDRDBDIRE RDB(AS20) RMTLOCNAME(AS20 *IP) TEXT('Remote DB system via
TCP/IP')
```

This relational database entry identifies a database name (RDB parameter), the remote system name, and that the connection is over TCP/IP. TCP/IP must be active on both the source and target systems. A Domain Name Services (DNS) server must be active in the network to resolve to the actual IP address.

Note that DRDA runs over SNA connections as well as TCP/IP.

3. Enter the Add Server Authentication Entry (ADDSVRAUTE) command:

```
ADDSVRAUTE USRPRF (TEAM02) SERVER (AS20) PASSWORD (T02EAM)
```

The SQL CONNECT TO *target system (remote server)-database* statement can explicitly specify USER (user ID) and USING (password) information. If it does not, the user ID and password information specified in the ADDSVRAUTE command are passed to the remote server. Depending on the target system's (remote server) security requirements for clients to connect to it, a user ID and, optionally, a user password are required that must be successfully validated on the remote server.

We strongly recommend that you enter the user profile, server name, and password values in uppercase.

4. To specify a password value for the ADDSVRAUTE command's PASSWORD parameter, the source system Retain server security data (QRETSVRSEC) system value must be set to 1.

Refer to 8.6.5.1, "Network Remote Servers" in the redbook *Managing AS/400 V4R4 with Operations Navigator*, for the Operations Navigator graphical interface that corresponds to the ADDSVRAUTE command.

**Note:** To use ADDSVRAUTE support, your user profile must specify \*SECADM special authority. You must also have \*OBJMGT and \*USE authorities to the user profile specified on this command.

On the As20 target (remote server) system, follow the steps in this process:

1. TCP/IP is started.
2. TCP/IP DDM server is started.

The DDM server jobs run in subsystem QSYSWRK. The jobs are named QRWTLSTN (daemon) and QRWTSRVR (server, one per connection). Refer to 5.5.2, "TCP/IP servers" in the redbook *Managing AS/400 V4R4 with Operations Navigator* for Operations Navigator interfaces to OS/400 TCP/IP servers.

3. The network attributes' DDM/DRDA Request (DDMACC) parameter for processing received DDM/DRDA requests is set to \*OBJAUT. This means normal OS/400 processing user profile authority to the requested file or table is performed.

This target or server system can be configured to not require a password from the source system. You do this by using the OS/400 Configure TCP (CFGTCP) command interface. Then select **Configure TCP/IP applications->Change DDM TCP/IP Attributes**.

4. A target system user ID and password must correspond to the user ID and password, if used, received from the requesting source system.

---

## 1.6 Change Query Attributes

Using the Change Query Attributes item that becomes available when you right-click Database gives you an easy way to change your query options for accessing the database. However, you must be aware that some of the options that are available here can be manipulated using the Change Query Attribute (CHGQRYA) CL command on the AS/400 system. There is not an exact

one-to-one correspondence. For a detailed discussion on the implications of changing query attributes, refer to the manual *DB2 UDB for AS/400 Database Performance and Query Optimization* in the AS/400 Information Center.

You can access the Change Query Attributes panel in two ways:

- From Operations Navigator, right-click **Database** and select **Change Query Attributes**.
- From the Run SQL Script center, select **Options** and **Change Query Attributes**.

You will see the Change Query Attributes dialog as shown in Figure 40.

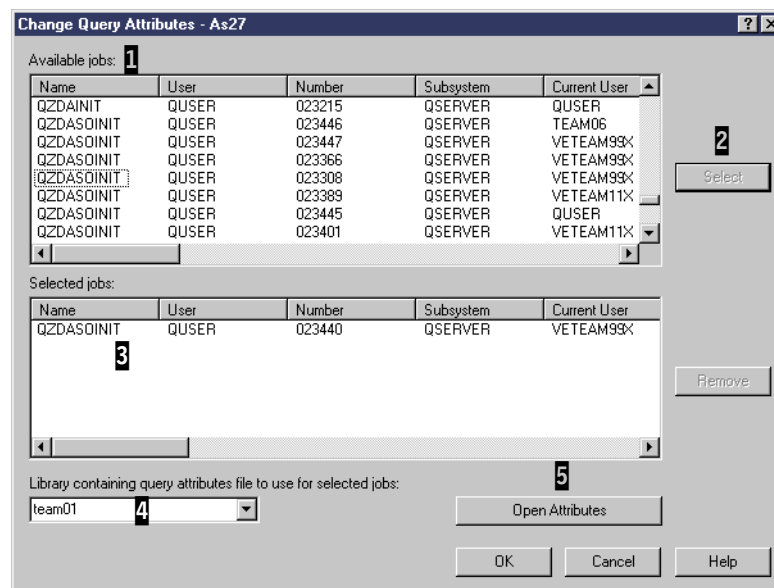
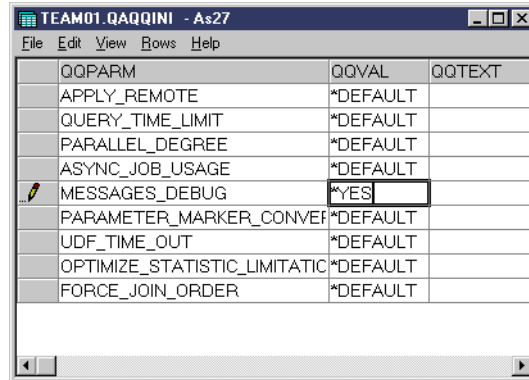


Figure 40. Change Query Attributes panel

In the upper part of the window (1), you see a list of all jobs currently active in the system. Scroll through the list to locate the job you are interested in, click on its name, and use the **Select** button (2) to move the selection in the bottom part of the panel (3). You can select more than one job and set common query attributes for all of them at the same time.

At this point, you can specify a library (4) in which you want the original QAQQINI file to be copied. Click the **Open Attribute** button (5), which allows you to edit the copy of QAQQINI you just made in library 4, as shown in Figure 41 on page 62.



QQPARAM	QQVAL	QQTEXT
APPLY_REMOTE	*DEFAULT	
QUERY_TIME_LIMIT	*DEFAULT	
PARALLEL_DEGREE	*DEFAULT	
ASYNC_JOB_USAGE	*DEFAULT	
MESSAGES_DEBUG	*YES	
PARAMETER_MARKER_CONVEP	*DEFAULT	
UDF_TIME_OUT	*DEFAULT	
OPTIMIZE_STATISTIC_LIMITATIC	*DEFAULT	
FORCE_JOIN_ORDER	*DEFAULT	

Figure 41. Editing QAQQINI

Click the cell you want to change and type the new value. As shown in Figure 41, we change the setting for `MESSAGES_DEBUG` from the original value `*DEFAULT` to `*YES`, therefore, stating that we want debug messages to be recorded for the selected job. When you press Enter to activate your changes, you receive a warning message (Figure 42). Click **Yes**.

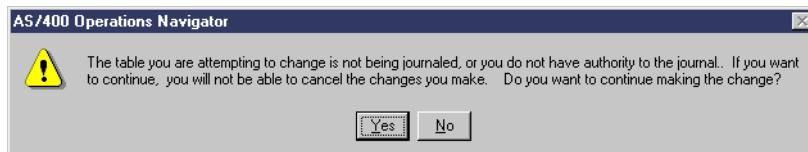


Figure 42. Warning message on modification of the QAQQINI file

You are brought back to the Change Query Attributes panel. Click **OK** to make the change effective.

Options currently managed in the QAQQINI file and their values are documented in *DB2 UDB for AS/400 Database Performance and Query Optimization*.

## 1.7 Current SQL for a Job

You can use this function to select any job running on the system and display the current SQL statement being run, if any. Besides displaying the last SQL statement being run, you can edit and rerun it through the Run SQL Script option (linked automatically) and display the actual job log for the selected job or, even end the job. This can also be used for database usage and performance analysis, with the Visual Explain tool documented in 1.9, “Visual Explain” on page 85.

To start it, right-click the **Database** item in Operations Navigator and select **Current SQL for a Job**. You are presented with the dialog shown in Figure 43.

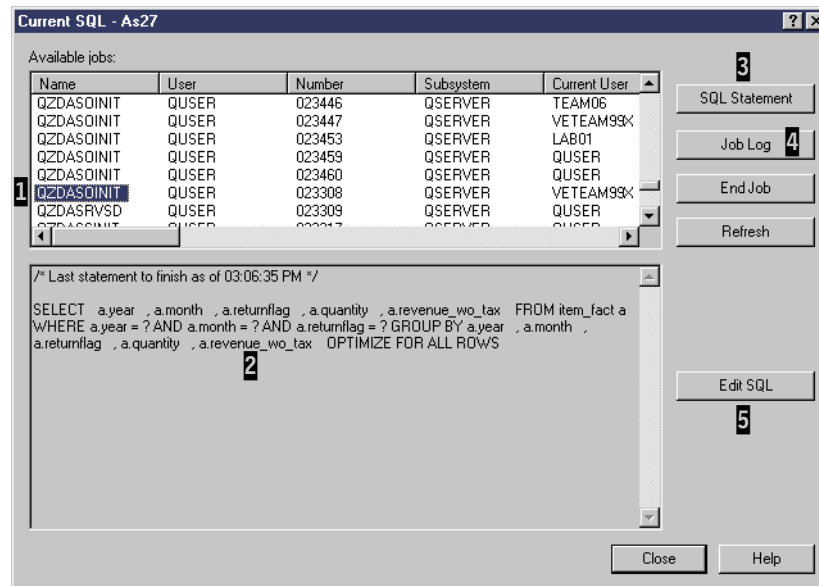


Figure 43. Current SQL for a Job

The Current SQL window displays the name, user, job number, job subsystem, and current user for the available jobs on your system. You can select a job and display its job log, the SQL statement currently being run, if any, decide to reuse this statement in the Run SQL Script Center, or even end the job, provided you have sufficient authority.

In our example, we selected an ODBC job (1) and displayed the last SQL statement it ran in the bottom part of the panel (2) using the SQL Statement button (3). To go to its job log, we would have used the button shown at 4.

After the SQL statement is brought in the bottom part of the panel, it is possible to use the **Edit SQL** button (5) to work on this same statement with the Run SQL Script center that was previously documented in this redbook. See Figure 44.

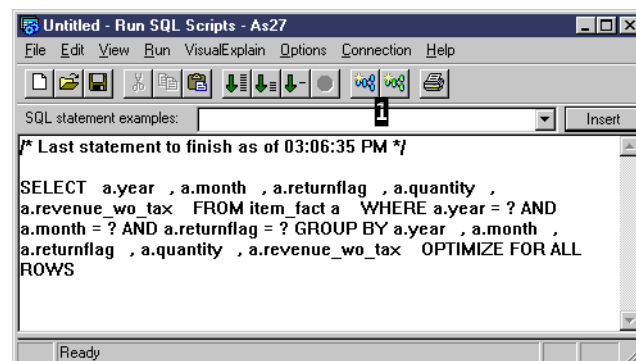


Figure 44. Working with Current SQL for a Job

From here, it is also possible to link into Visual Explain, using the appropriate menu item or the icons at 1 to help you with database performance analyses. For a discussion on this tool, refer to 1.9, “Visual Explain” on page 85.

As you may have already noticed, all Operations Navigator database tools are tightly integrated into each other to make it easier for the user to fully exploit their capabilities.

---

## 1.8 SQL Performance Monitors overview

You can analyze the performance of AS/400 SQL statements by putting the appropriate OS/400 job into debug mode, running the SQL statements, and viewing the Query Optimizer messages in the job log. We have an example of using job log messages in 1.5.3.7, “Controlling SQL run options” on page 56.

This section describes a more powerful SQL performance analysis tool that appeared in V4R4 Operations Navigator and further enhanced in V4R5. This support provides a graphical interface to IBM-provided SQL queries against data collected by the *Memory Resident Database Monitor* that was introduced in V4R3. In addition to output equivalent to the debug mode optimizer messages, this monitor can monitor multiple jobs and show the actual SQL statement. This interface is referred to as the *SQL Performance Monitors*.

The SQL Performance Monitor, which was originally available in V4R4, only allowed gathering *summary* performance information from the Memory Resident Database Monitor. While in V4R5, its possible to enhance the usability of this interface by collecting *detailed* performance information. For a detailed discussion on the *Memory Resident Database Monitor*, refer to the manual *DB2 UDB for AS/400 Database Performance and Query Optimization*.

Before you start an SQL Performance Monitor, you need to determine which job or jobs you want to monitor. There are several techniques you can use to determine the job. We list some of them here:

- If you are using SQL statements running Operations Navigator **Database-> Run SQL Scripts**, you can click the **View** option from the menu bar. On the drop-down menu that appears, click **Job Log** to see your current job’s job log. Included in the gray header portion of the job log messages is the name of the job, for example, 139224/QUSER/QZDASOINIT. You can scan down to the earliest job log messages to confirm this job is actually running under the user profile you think it should be.
- If you are not running the job that needs to be monitored, you can get the job name from the user of the job, if possible.
- If you know the user profile running the SQL jobs, but do not know which job is the one you want to monitor, you can use the OS/400 Work with Object Locks (`WRKOBJLCK`) command to find the jobs running with that user profile. You may receive more jobs than you anticipated. Then, you may need to look in the job logs of each job for some SQL-like messages to determine which job or jobs to monitor, for example:

```
WRKOBJLCK OBJ(QSYS/TEAM02) OBJTYPE(*USRPRF) MBR(*NONE)
```

This command resulted in five jobs running with user profile TEAM02: one job name starting with QPADEV000L (5250 emulation), two jobs running Client Access Express database serving with job name starting with QZDASOINIT (not using SSL), and two jobs with the job name starting with QZRCSRVS (central server functions). We looked in the job logs for the two QZDASOINIT



jobs and in one of them found the messages: 148 rows fetched from cursor CRSR0002.

This QZDASOINIT job was set by Operations Navigator Run SQL Scripts to Include debug messages in a job log.

- You can use the Operations Navigator server jobs interface to find the job by selecting from Operations Navigator **Network->Servers-> Client Access** to view the Client Access Express servers as shown in Figure 45.

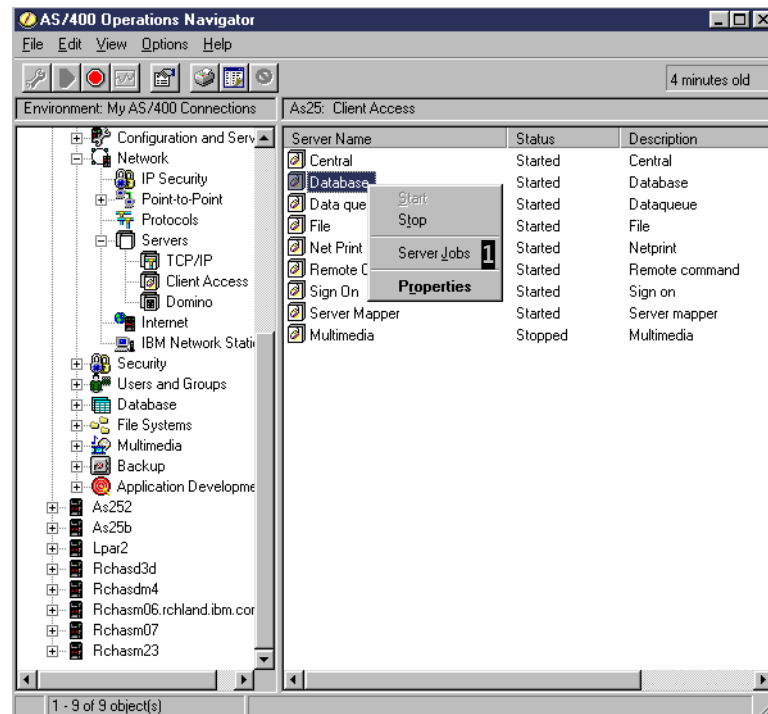


Figure 45. Finding the database server job (Part 1 of 2)

When you click Server Jobs (see **1**), a window appears similar to the one shown in Figure 46. This display shows the database server jobs, QZDASOINIT (not using SSL), that are currently started and shows a current user ID for jobs currently doing active database functions.

The screenshot shows the 'Server Jobs - AS25' window. The table below lists the jobs currently running on the system.

Job name	Current user	Server	Job type	Job status	Time entered system	Date entered system	Thread count
Qzdassinit		OS/400 TCP Data	Batch	Printer output	00:42:39	01/25/00	0
Qzdasrvsd		OS/400 TCP Data	Batch	Printer output	00:42:39	01/25/00	0
Qzdasoinit		OS/400 TCP Data	Batch	Printer output	14:45:38	01/28/00	0
Qzdasoinit		OS/400 TCP Data	Batch	Printer output	14:45:38	01/28/00	0
Qzdasoinit		OS/400 TCP Data	Batch	Printer output	14:45:38	01/28/00	0
Qzdasoinit		OS/400 TCP Data	Batch	Printer output	15:13:51	01/27/00	0
Qzdasoinit	QUSER	OS/400 TCP Data	Batch	Active	06:30:38	01/31/00	1
Qzdasoinit	TEAM02	OS/400 TCP Data	Batch	Active	06:30:38	01/31/00	1
Qzdassinit		OS/400 TCP Data	Batch	Printer output	07:12:08	01/03/00	0
Qzdassinit	QUSER	OS/400 TCP Data	Batch	Active	15:05:59	01/28/00	1
Qzdasrvsd	QUSER	OS/400 TCP Data	Batch	Active	15:05:59	01/28/00	1

Figure 46. Finding the database server job (Part 2 of 2)

This display illustrates an advantage of using the Operations Navigator “servers” support to find a job, compared to using OS/400 5250-display based commands such as the Work with Subsystem Jobs (WRKSBSJOB), Work with Active Jobs (WRKACTJOB), or Work with Object Locks (WRKOBJLCK) commands.

The Operations Navigator interface lists the jobs based on their function. With the OS/400 commands, you need to understand what OS/400 subsystem the server jobs run in and the job name that identifies the server function. In our example, you need to know that the QZDASOINIT jobs do the database serving (in this case ODBC-based) work. You also need to look into the job logs of each active job to find the actual user ID (profile) using the job.

The OS/400 commands we discussed show equivalent jobs with the user ID as QUSER. QUSER is the user profile assigned by the system for pre-started Client Access database server jobs. The user profile name actually using the job is indicated in a job log message. The Operations Navigator interface examines the job log messages and shows the active user profile (TEAM02, in our example) if the pre-started job is currently in session with a signed on client.

### 1.8.1 Starting the SQL Performance Monitor

To run an SQL Performance Monitor, you need to:

- Define a new monitor
- Determine whether it’s going to be a *Detailed* collection or a *Summary* collection
- Specify the jobs to be monitored and the data to be collected for a Summary collection

We discuss the Detailed collection later in 1.8.1.2, “Detailed SQL Performance Monitor example” on page 70.

#### 1.8.1.1 Summary SQL Performance Monitor example

To start the SQL monitoring process, right-click **SQL Performance Monitors**, and select **New** as shown in Figure 47.

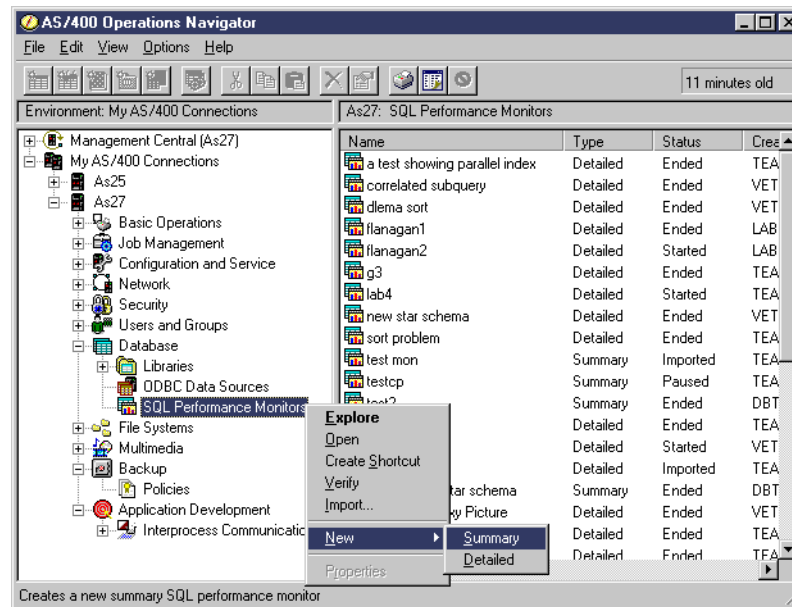


Figure 47. SQL Performance Monitor (Part 1 of 5)

Now select **Summary**. This brings up the New SQL Performance Monitor dialogue panel with three tabs: General, Monitored Jobs, and Data to Collect.

The General tab is shown in Figure 48.

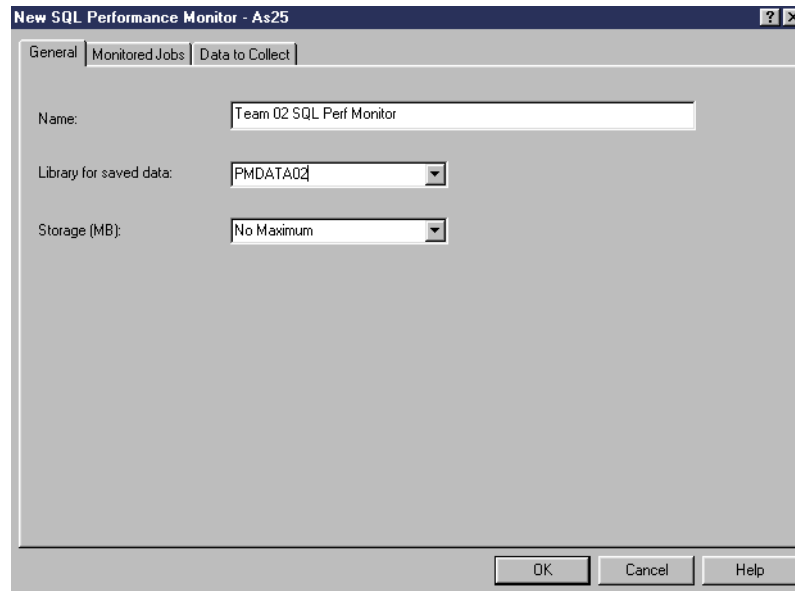


Figure 48. Starting a Summary SQL Performance Monitor (Part 2 of 5)

We entered the monitor name, the library name that is used to contain the collected data, and the amount of main storage allocated to the monitoring process.

Do *not* click the OK button yet. Monitoring all jobs will start if you have not selected specific jobs under the Monitored Jobs tab. Monitoring all jobs is not

recommended on a system with hundreds of active jobs because the monitoring process can degrade performance.

To specify which OS/400 jobs to manage, click the **Monitor Jobs** tab, which brings up the panel shown in Figure 49.

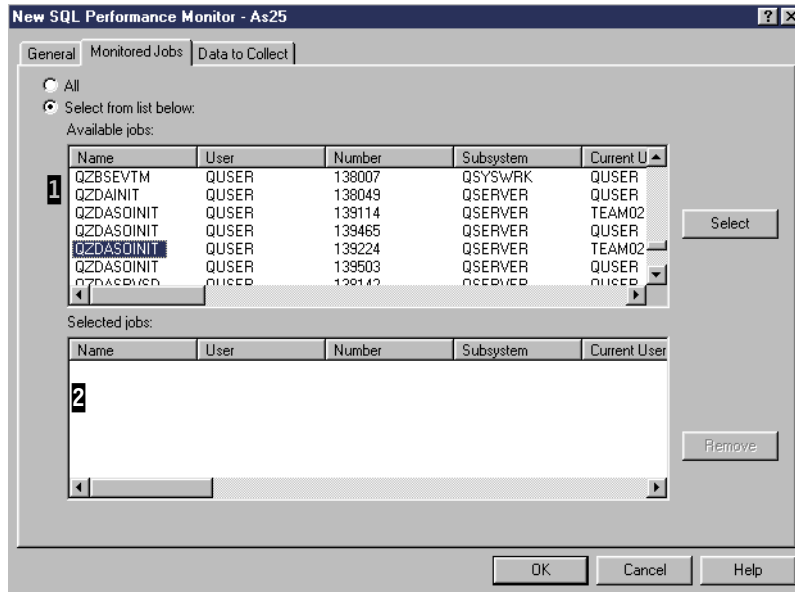


Figure 49. Starting a Summary SQL Performance Monitor (Part 3 of 5)

You can select to monitor all jobs or to select jobs from the Available jobs list pane shown in **1**. After you select a job and click the **Select** button, the job information is entered into the Selected jobs list pane **2**. You remove selected jobs by selecting a job in the Selected jobs pane and clicking the **Remove** button.

In this example, we scrolled down the active job names to display the ones shown in **1**. We select to monitor only job QZDASOINIT/QUSER/139224 with TEAM02 as the current user. We recommend that you monitor as few jobs as possible, because monitoring a large number of active jobs could impact normal productivity.

When you are finished selecting jobs, click the **Data to Collect** tab. This brings up the panel shown in Figure 50.

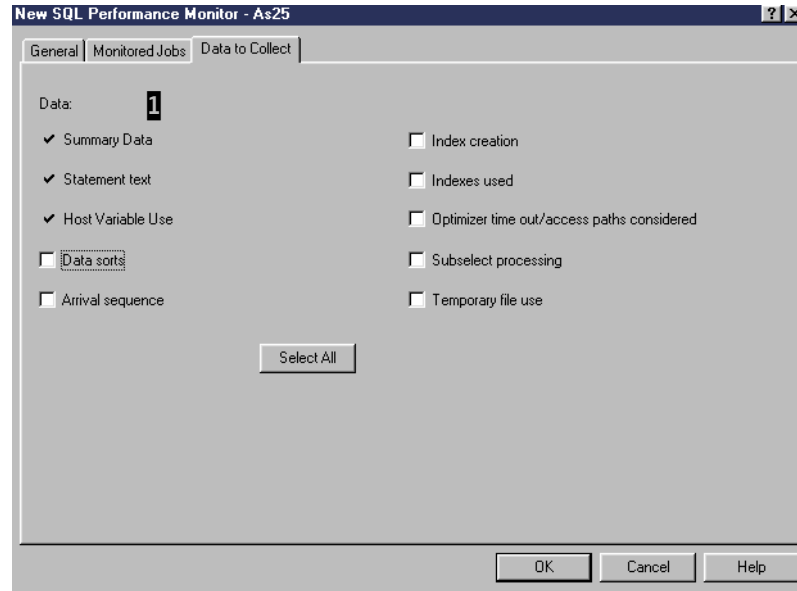


Figure 50. Starting a Summary SQL Performance Monitor (Part 4 of 5)

This panel shows three sets of SQL monitor data collected during every monitor collection period at **1**. You can specifically include other sets of data or simply click the **Select All** button. You should select *all*, unless you understand the application implementation in detail so that you need to collect only specific information.

When you are satisfied with your monitor collection specification, click the **OK** button to return to the original SQL Performance Monitor window, which shows the monitor status on the right pane in Figure 51.

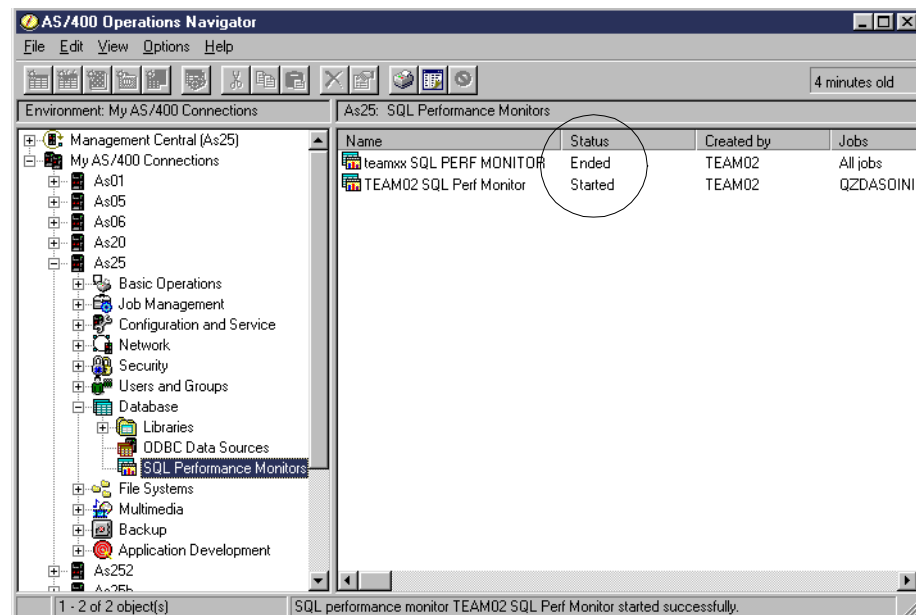


Figure 51. Starting an SQL Performance Monitor (Part 5 of 5)

In our example, we used Run SQL Scripts to run the SQL statement. This statement has a relatively complex WHERE clause as shown in Figure 52. Run SQL Scripts is discussed in more detail in 1.5, “Run SQL Script” on page 45.

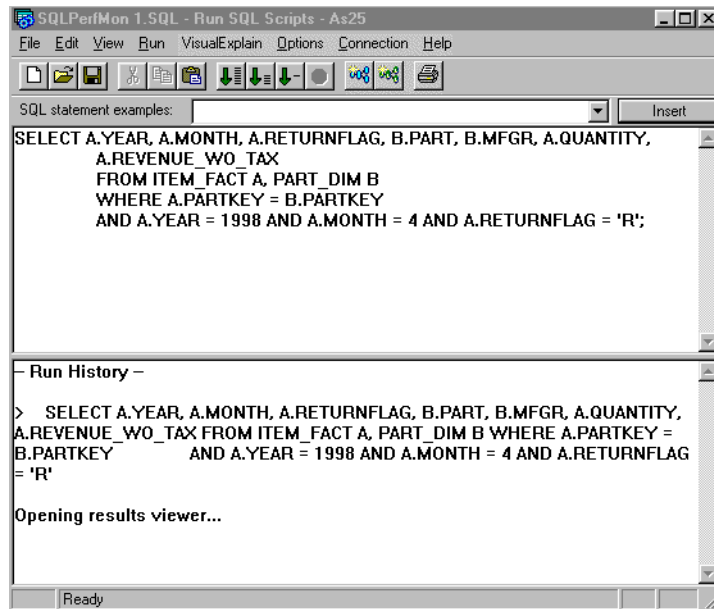


Figure 52. SQL Performance Monitor: SQL statement monitored

Operations Navigator Run SQL Scripts support uses Open Database Connectivity (ODBC) support. In our example figure, the SQL statement was already run based on the evidence of its appearance within the Run History pane. The message `Opening results viewer...` indicates the results of the SQL select statement has already been displayed to the Operations Navigator user.

The SQL Performance Monitor can monitor all SQL work performed on OS/400. In addition to Operations Navigator Run SQL Scripts jobs, other users of OS/400 SQL support would include a client workstation Visual Basic program accessing the OS/400 via ODBC, a client workstation Java applet accessing the OS/400 via Java Database Connectivity (JDBC), a local AS/400 program using embedded SQL in the RPG, COBOL, or C program, a local AS/400 program using the SQL CLI (Call Level Interface) in RPG, COBOL, C, or Java.

OS/400 also has a 5250 workstation-based SQL interface running under the Start SQL (STRSQL) command.

### 1.8.1.2 Detailed SQL Performance Monitor example

To start the SQL monitoring process, right-click **SQL Performance Monitors**, and select **New**. Select **Detailed** as shown in Figure 53.

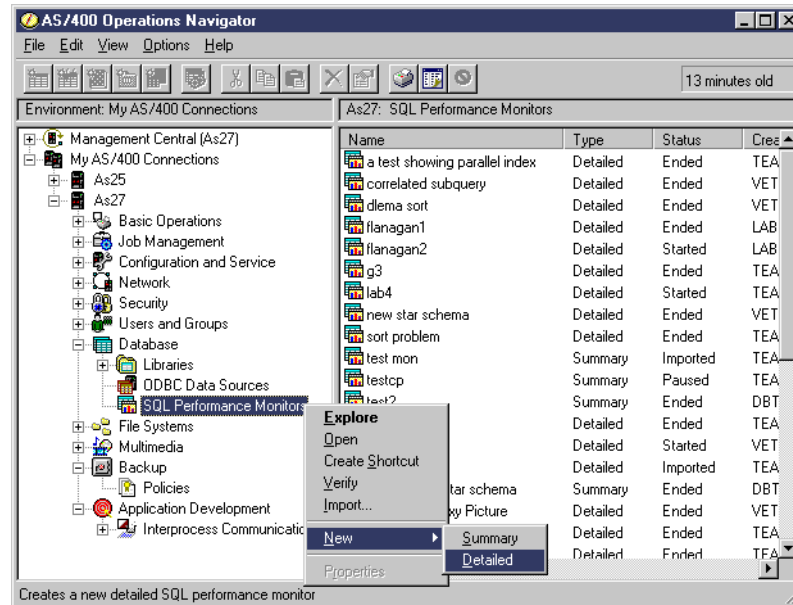


Figure 53. Starting a Detailed SQL Performance Monitor

Name the monitor. Select a library for the the collected data and proceed to select the jobs you want to monitor as previously documented in 1.8.1.1, “Summary SQL Performance Monitor example” on page 66.

## 1.8.2 Reviewing the SQL Performance Monitor results

The SQL Performance Monitor statistics are kept in main storage for fast recording, but need to be written to database files to use the Operations Navigator interface to review the results. You can have the statistics written to database files by either *pausing* or *ending* the monitor.

Right-click the active SQL Performance Monitor. A pop-up window appears that lists the Pause, End, and other monitor actions as shown in Figure 54.

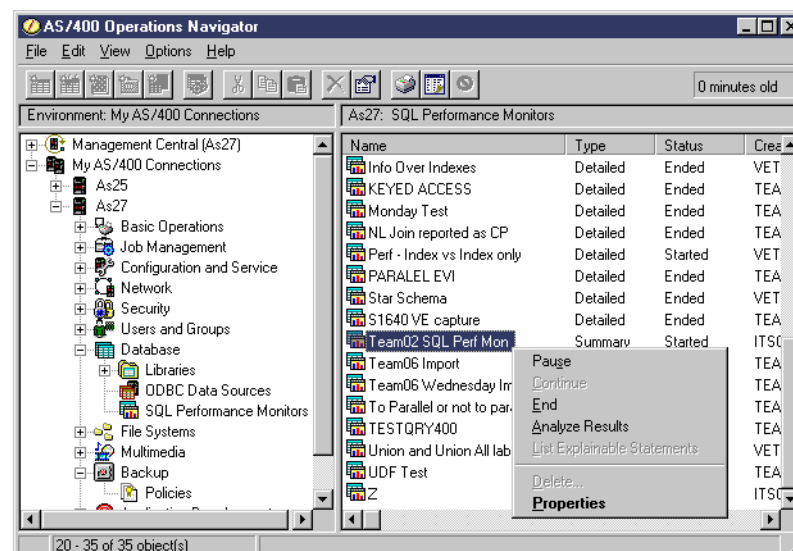


Figure 54. Managing the SQL Performance Monitor

The possible managing functions are:

- **Pause:** This stops the current collection of statistics and writes the current statistics into several database files or tables that can be queried by selecting the Analyze Results action. The monitor remains ready to collect more statistics, but requires the Continue action to restart collection.
- **Continue:** This restarts the collection of statistics for a monitor that is currently paused.
- **End:** This stops and ends the monitor and writes the current collection of statistics to the database files or tables.
- **Analyze Results:** This brings up a window with three tabs for selecting ways to look at (query) the collected statistics in the database files or tables:
  - Summary Results
  - Detailed Results
  - Composite View

We show an example of a Detailed Result query report in Figure 58 on page 74.

- **List Explainable Statements:** This opens a dialog listing the SQL statements for which the detailed SQL Performance Monitor has collected data and for which a Visual Explain diagram can be produced. See 1.8.2.3, “List Explainable Statements” on page 78, for an example.
- **Properties:** This brings up a window with three tabs representing the original monitor definition:
  - General
  - Monitored Jobs
  - Saved Data

An example of the Saved Data tab with the details for our monitor is shown in Figure 55.

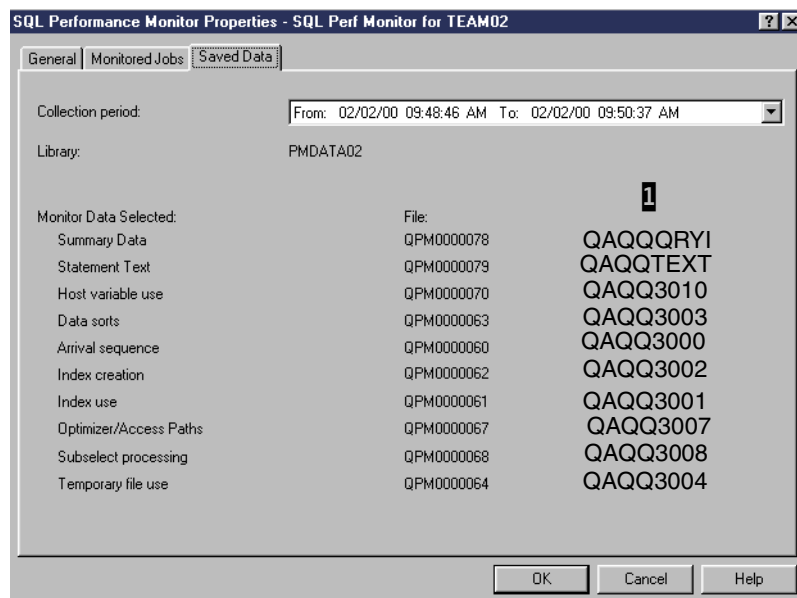


Figure 55. SQL Performance Monitor: Properties



Figure 55 shows the files or tables that correspond to the Data to Collect, specified in Figure 50 on page 69. The monitor files have generic names with sequence numbers as suffixes. We show the corresponding file or table name (under **I**) described in the memory resident database monitor documentation within the book *DB2 UDB for AS/400 SQL Programming*, SC41-5611.

The SQL Performance Monitor file name numeric suffix is updated when each monitor is started.

### 1.8.2.1 Analyzing a Summary SQL Performance Monitor

OS/400 provides many pre-defined queries to view the recorded statistics. You can select these queries by checking the various query types on the Analyze Results panels. To begin viewing the results, right-click the paused or ended monitor. Select **Analyze Results** from the pop-up window. Here we analyze results for a Summary Monitor.

Figure 56 shows the first results panel that groups queries according to three tabs:

- Summary Results
- Detailed Results
- Composite View

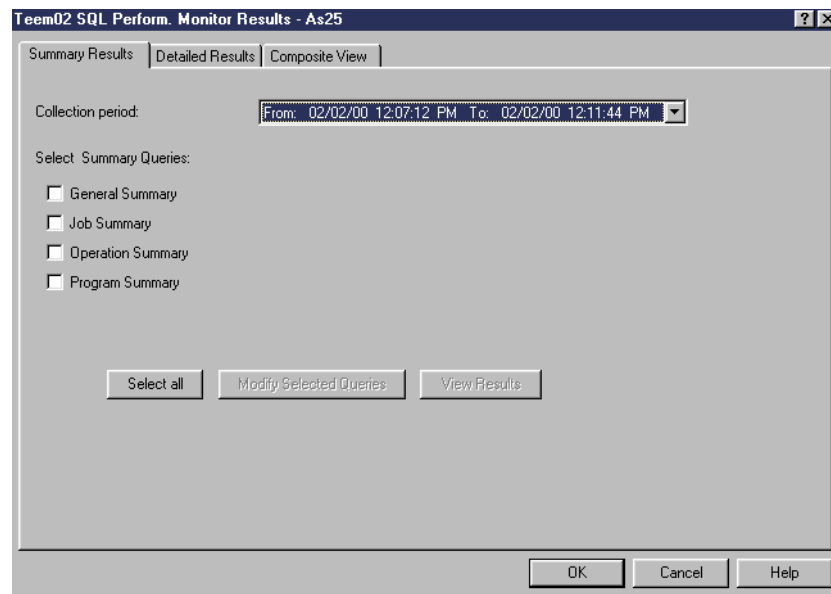


Figure 56. SQL Performance Monitor: Analyze Results - Summary Results queries

Table 2 on page 75 summarizes the various IBM-provided queries under each tab. You can select individual queries or use the Select All button. After selecting the queries you want to run, click the **View Results** button, which will be activated.

You can even choose to modify the pre-defined queries and run the new queries, by clicking the Modify Selected Queries button.

An in-depth discussion of using the SQL Performance Monitor results to improve performance is beyond the scope of this redbook. However, we show sample query results output for our SQL statement used in Figure 52 on page 70.

To obtain the query results shown in Figure 58, you must first select the **Detailed Results** tab on the Performance Monitor Results window shown in Figure 56. This brings up the Detailed Results panel shown in Figure 57.

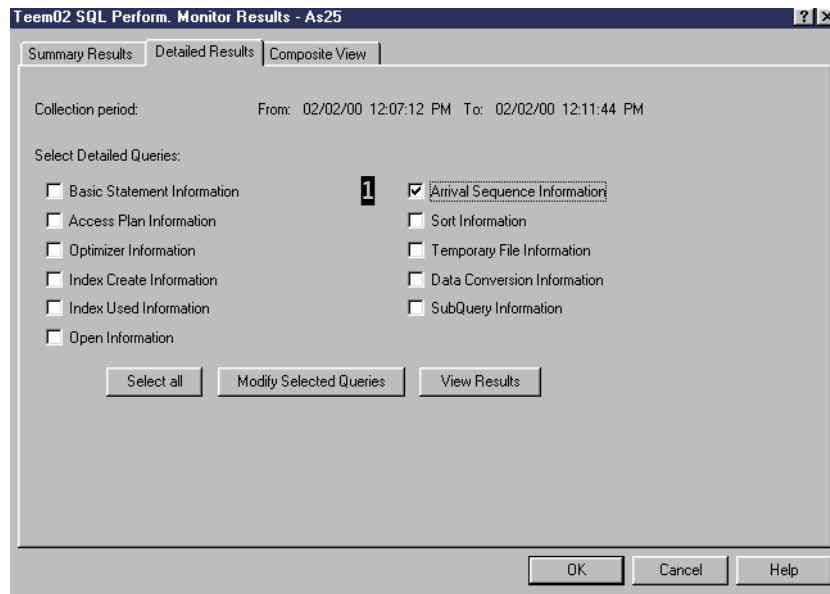


Figure 57. SQL Performance Monitor: Detail Results

You can select individual detail query reports, select *all* queries, and even modify the provided queries. When you are finished selecting the queries you want, click the **View Results** button.

The OS/400 query optimizer support includes an Index Advisor function. This support includes, when appropriate, a recommendation that a new index should yield improved performance. Columns that should be used in the index are listed. To view this detailed information, you must first select to view Arrival Sequence Information as shown at **1** in Figure 57. Click the **View Results** button to access a panel similar to the one shown in Figure 58.

	Time	Estimated Processing Time	Reason Code	Total Rows In Base Table	Estimated Rows Selected	Advised Index	Advised Index Keys	St
1	2000-02-02 12:10:46.795992	0.004	T1	100029	90	Yes	YEAR, MONTH, RETURNFLAG	1
2	2000-02-02 12:07:53.939024	0.002	T1	100029	100	Yes	YEAR, MONTH, RETURNFLAG	1

Figure 58. SQL Performance Monitor: Arrival Sequence Information

To view the information contained within Figure 58, we had to scroll to the right to find the columns Advised Index and Advised Index Keys shown at **1**. You can see at **2** that we compressed several columns in the results to make the index path information fit within the window.

A lab exercise can be downloaded to your AS/400 system on a PC workstation listed in the beginning of this chapter. The “Self study lab” can be used to familiarize yourself with the power of the SQL Performance Monitor, as well as most of the Operations Navigator Database support. It also includes tips on tuning SQL performance.

Table 2 summarizes the results queries grouped under the Summary Results, Detailed Results, and Composite View tab results categories.

Table 2. Summary SQL Performance Monitor: Analyze results queries

Query group or name	Description
<b>Summary Results Group</b>	<b>Group contains several views of summary information</b>
General Summary	Contains a row of information that summarizes all SQL activity
Job Summary	Contains a row of information for each job. Each row summarizes all SQL activity for that job
Operation Summary	Contains a row of summary information for each type of SQL operation
Program Summary	Contains a row of information for each program that performed an SQL operation. Each row summarizes all SQL activity for that program
<b>Detailed Results Group</b>	<b>Group contains several views of detail level information</b>
Basic Statement Information	Summarizes the activity for each unique SQL statement
Access Plan Information	Contains a row of information for each SQL statement that required the access plan (algorithm to find rows) to be rebuilt
Optimizer Information	Contains a row of optimizer information for each subselect in an SQL statement
Index Create Information	Contains a row of information for each SQL statement that required an index to be built
Index Used Information	Contains a row of information for each SQL statement that needed to use the index
Open Information	Contains a row of information for each table open activity for each SQL statement
Arrival Sequence Information	Contains a row of information for each select that required rows to be processed in arrival sequence order. If appropriate, includes column names for a recommended new index
Sort Information	Contains a row of information for each sort that an SQL statement performed.
Temporary File Information	Contains a row of information for each SQL statement that required a temporary result
Data Conversion Information	Contains a row of information for each SQL statement that required data conversion
SubQuery Information	Contains a row of subquery information
<b>Composite View</b>	<b>These reports join data from selected detail reports and includes summary data and SQL statement text</b>
Arrival Sequence	Contains the table scan data for monitored jobs
Data Sorts	Contains details of data sorts that monitored jobs performed
Host Variable Usage	Contains the host variable values used by monitored jobs
Optimizer time outs and access paths	Contains details of any query time outs in monitored jobs
Indexes Used	Contains details of how indexes are used by monitored jobs
Index Creation	Contains details of index creation by monitored jobs
Subselect Processing	Contains information about each subselect in an SQL statement

Query group or name	Description
Temporary File Use	Contains details of temporary files that monitored jobs created

### 1.8.2.2 Analyzing a Detailed SQL Performance Monitor

Most of the discussion in 1.8.2.1, “Analyzing a Summary SQL Performance Monitor” on page 73, also applies to a Detailed Monitor.

Figure 59 shows the first results panel that groups queries according to three tabs:

- Summary Results
- Detailed Results
- Extended Detailed Results

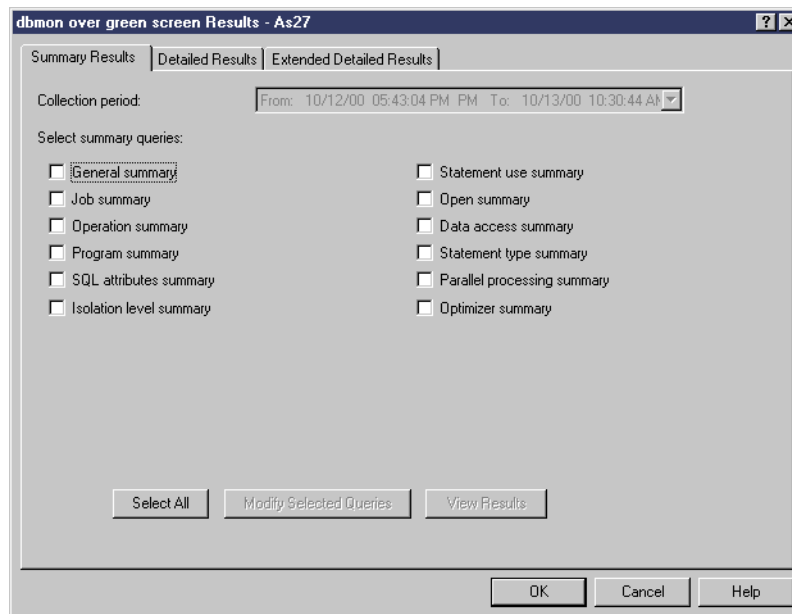


Figure 59. Detailed SQL Performance Monitor: Analyze Results - Detailed Monitor

For each of these options, you can run any of the pre-prepared queries or modify them for your own analysis.

Table 3 summarizes all the results queries from the Summary Results, Detailed Results, and Extended Detailed Results tabs.

Table 3. Detailed SQL Performance Monitor: Analyze result queries

Query group or name	Description
<b>Summary Results Group</b>	<b>Group contains several views of summary information</b>
General Summary	Contains a row of information that summarizes all SQL activity.
Job Summary	Contains a row of information for each job. Each row summarizes all SQL activity for that job.
Operation Summary	Contains a row of summary information for each type of SQL operation. Each row summarizes all SQL activity for that type of SQL operation.
Program Summary	Contains a row of information for each program that performed SQL operations. Each row summarizes all SQL activity for that program.

Query group or name	Description
SQL Attributes Summary	Contains a summary of the optimization attributes. This option is only available when you use a detailed SQL Performance Monitor.
Isolation Level Summary	Contains a summary of the number of statements that were run under each isolation level. This option is only available when you use a detailed SQL Performance Monitor.
Statement Use Summary	Contains a summary of how many statements are executed and the number of times they are executed during the performance monitor collection period. This option is only available when you use a detailed SQL Performance Monitor.
Open Summary	Contains a summary of how many statements perform an open the number of times they are executed during the performance monitor collection period. This option is only available when you use a detailed SQL Performance Monitor.
Data Access Summary	Contains a summary of the number of SQL statements that are read-only versus those that modify data. This option is only available when you use a detailed SQL Performance Monitor.
Statement Type Summary	Contains a summary of whether SQL statements are in extended dynamic packages, system-wide statement cache, regular dynamic, or static SQL statements. This option is only available when you use a detailed SQL Performance Monitor.
Parallel Processing Summary	Contains a summary of the parallel processing techniques used. This option is only available when you use a detailed SQL Performance Monitor.
Optimizer Summary	Contains a summary of the optimizer techniques used. This option is only available when you use a detailed SQL Performance Monitor.
<b>Detailed and Extended Detailed Results Group</b>	<b>Group contains several views of detail level information</b>
Basic Statement Information	Summarizes the activity for each unique SQL statement.
Access Plan Information	Contains a row of information for each SQL statement that required the access plan (algorithm to find rows) to be rebuilt.
Optimizer Information	Contains a row of optimizer information for each subselect in an SQL statement.
Index Create Information	Contains a row of information for each SQL statement that required an index to be built.
Index Used Information	Contains a row of information for each index that an SQL statement needed to use.
Open Information	Contains a row of information for each table open activity for each SQL statement.
Index advised information	Provides information about advised indexes. This option is only available when you use a detailed SQL Performance Monitor.
Governor time out information	Provides information about all optimizer time outs. This option is only available when you use a detailed SQL Performance Monitor.
Optimizer time out information	Provides information on any optimizer time outs. This option is only available when you use a detailed SQL Performance Monitor.
Procedure call information	Provides information on Procedure call usage. This option is only available when you use a detailed SQL Performance Monitor.
Hash table information	Provides information on any temporary has tables that were used. This option is only available when you use a detailed SQL Performance Monitor.

Query group or name	Description
Distinct processing information	Provides information about any DISTINCT processing. This option is only available when you use a detailed SQL Performance Monitor.
Table scan	Contains a row of information for each subselect that required records to be processed in arrival sequence order.
Sort Information	Contains a row of information for each sort that an SQL statement performed.
Temporary File Information	Contains a row of information for each SQL statement that required a temporary result.
Data Conversion Information	Contains a row of information for each SQL statement that required data conversion.
SubQuery Information	Contains a row of subquery information.
Row access information	Contains information about the rows returned and I/Os performed. This option is only available when you use a detailed SQL Performance Monitor.
Lock escalation information	Provides information about any lock escalations. This option is only available when you use a detailed SQL Performance Monitor.
Bitmap information	Provides information about any bitmap creates or merges. This option is only available when you use a detailed SQL Performance Monitor.
Union merge information	Provides information about any union operations. This option is only available when you use a detailed SQL Performance Monitor.
Group By information	Provides information about any GROUP BY operations. This option is only available when you use a detailed SQL Performance Monitor.

Although the items listed under the Detailed and Extended Detailed Results tabs have the same names and descriptions, the underlying queries are different. The Extended ones allow you a more complete understanding of the Optimizer choices. You can easily verify this by selecting the same item in both lists and clicking the **Modify Selected Query** button to have the SQL statement opened with the Run SQL Script center.

### 1.8.2.3 List Explainable Statements

The Explainable Statements for SQL Performance Monitor dialog lists the SQL statements for which a detailed SQL Performance Monitor has collected data and for which a Visual Explain graph can be produced.

To access this function, click **Operations Navigator->Database->SQL Performance Monitors**. Then you will see a list of the SQL Performance monitors that are currently on the system. Right-click a *detailed* SQL Performance Monitors collection and select **List Explainable Statements**, as shown in Figure 60.

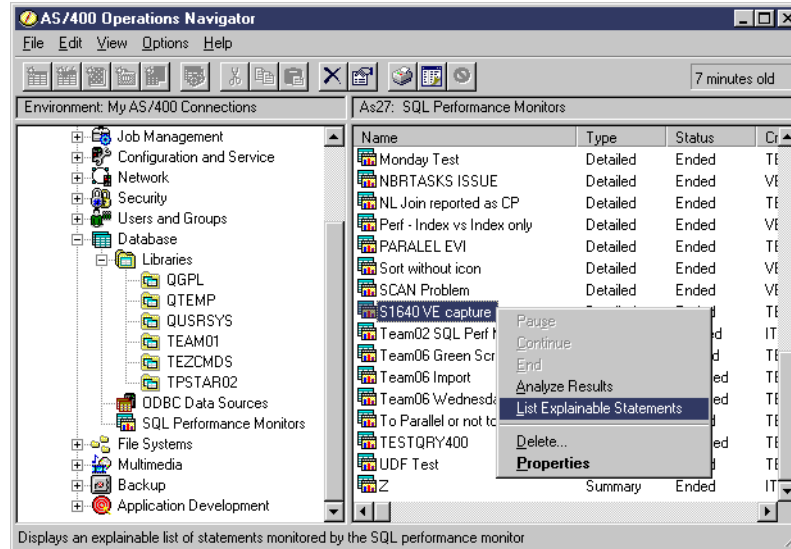


Figure 60. List Explainable Statements

When you select this item, you see the panel in Figure 61. The upper half of the panel displays the SQL statements monitored during the data collection session. Click to select the statement **1** you are interested in analyzing. The selected statement will appear in the lower half of the dialog.

Once the statement is in focus, it is possible to have it analyzed and explained. Click the **Run Visual Explain** button shown at **2**. Refer to 1.9, “Visual Explain” on page 85, for a detailed discussion on this tool.

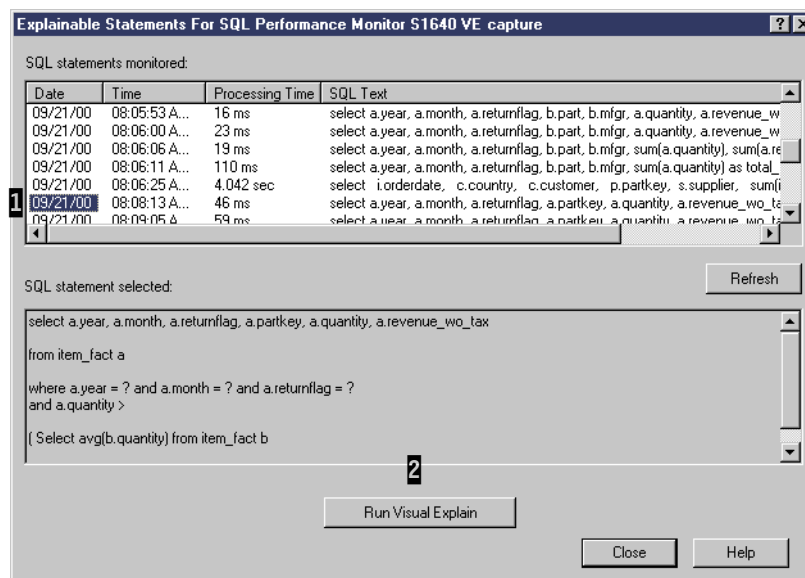


Figure 61. Using List Explainable Statements

As you may have already noticed, the set of database analysis options and tools provided by Operations Navigator are well interconnected and meant to be used together in an iterative fashion.

### 1.8.3 Importing data collected with Database Monitor

In V4R5, it is possible to import Database performance data collected with the more traditional green-screen interface tool, known as Database Monitor. You can start Database Monitor by using either the STRDBMON or STRPFMON STRDBMON(\*YES) CL commands on the AS/400 system. While it is beyond the purpose of this redbook to discuss the traditional collection methods, users will certainly be pleased to discover that this data can be analyzed with the simpler and more intuitive instruments made available with Operations Navigator, rather than with the traditional approach.

To import data collected with Database Monitor, select **Database->SQL Performance Monitor->Import** as shown in Figure 62.

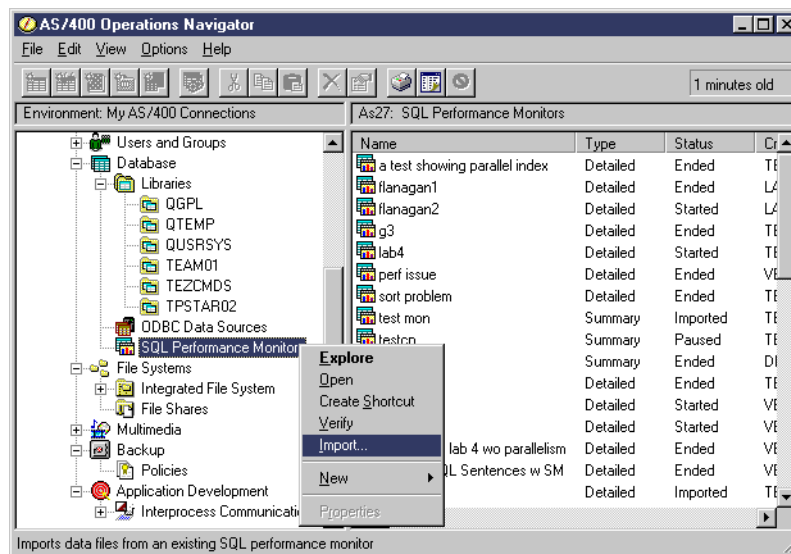


Figure 62. Importing data in SQL Performance Monitor (Part 1 of 2)

A panel appears like the example in Figure 63. On this display, you give a name to new monitor and specify the the library and file containing the collected data.

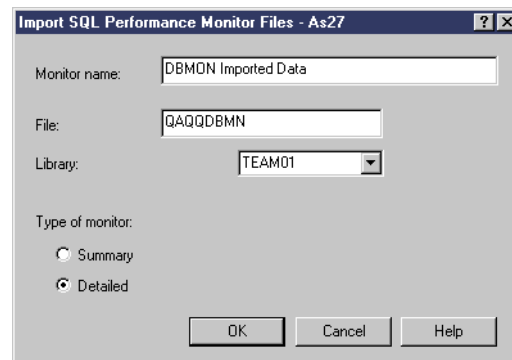


Figure 63. Importing data in SQL Performance Monitor (Part 2 of 2)

Due to some new fields being added to the Database Monitor file in V4R5, SQL Performance Monitor only fully supports importing and analyzing database performance data collected in V4R5. Data collected from earlier releases will not have all of the information needed by Visual Explain. The system imports data from earlier releases and converts the data to a V4R5 format. However, it can



only use default values for information that was not recorded at the earlier release, and full results cannot be guaranteed when using Visual Explain.

Please refer to 1.8.2, “Reviewing the SQL Performance Monitor results” on page 71, for a discussion on how to analyze the collected data.

Exit the Visual Explain window and the Explainable Statements window after you complete your analysis. Depending on your future needs for further investigation, you may either retain the performance data or delete it from the system at this time. To delete an SQL Performance Monitor collection, right-click the data collection you are interested in, and select **Delete**.

### 1.8.3.1 Importing performance data for Query/400

Query/400 is not included in the list of queries that can be monitored by the SQL Performance Monitor, even though debug messages can be used with Query/400 queries.

Because Query/400 queries are often blamed for poor performance, and sometimes even banned from execution during daylight hours, it was thought appropriate to provide guidance to bring Query/400 queries into the scope of Visual Explain.

There is no direct Query/400 to SQL command. However, the STRQMQRV CL command will run a query definition (object type \*QRYDFN) as an SQL statement, as long as the parameter ALWQRYDFN is set to either \*YES or \*ONLY.

To use this SQL statement with Visual Explain, either start an SQL Performance Monitor for this job before you issue the STRQMQRV command or use the native STRDBMON CL command to collect detailed data for the job. See 1.8.1, “Starting the SQL Performance Monitor” on page 66, for further information.

Alternatively, you can access the SQL statement by using the Current SQL for a Job option (obtained by right-clicking the database icon in Operations Navigator).

Here we document the actions you need to follow to import the database performance data collected for Query/400 using the STRQMQRV command as a workaround. For documentation on the differences between STRQRY and STRQMQRV, see:

<http://publib.boulder.ibm.com/pubs/html/as400/v4r5/ic2924/info/index.htm>

Use STRQMQRV as a search word.

In our example, we collect database performance data on a pre-existing Query/400 definition, named TESTQRY in library ITSCID41 using Database Monitor on the AS/400 system. We are going to create an SQL Performance Monitor named QRYIMPORT. We use this process:

1. Start the traditional green-screen tool to collect database performance data, Database Monitor, into a file named QAQQDBMN in library ITSCD41, using the STRDBMON CL command:

```
STRDBMON OUTFILE(ITSCID41/QAQQDBMN) TYPE(*DETAIL)
COMMENT('Collecting Data for Query/400 - to be Imported in SQL Monitor')
```

See Figure 64 on page 82 for a sample of this command. All traditional 5250 commands and activities documented here have been performed in the *same*

*working session.* Had it been otherwise, the STRDBMON command should point to the correct job (STRDBMON JOB (xxxxxx/USER/JOBNAME)).

```

Start Database Monitor (STRDBMON)

Type choices, press Enter.

File to receive output . . . . . > QAQQDBMN      Name
Library . . . . . > ITSCID41      Name, *LIBL, *CURLIB
Output member options:
Member to receive output . . . *FIRST      Name, *FIRST
Replace or add records . . . . *REPLACE  *REPLACE, *ADD
Job name . . . . . *                Name, *, *ALL
User . . . . .                               Name
Number . . . . .                               000000-999999
Type of records . . . . . > *DETAIL      *SUMMARY, *DETAIL
Force record write . . . . . *CALC      0-32767, *CALC
Comment . . . . . > Collecting Data for Query?400 -
                          to be Imported in SQL Monitor

                                                                    Bottom
F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys

```

Figure 64. Start Database Monitor

- Now that we are recording all the database activities for our current job, we can run the previously prepared Query/400 definition using Query Management Query, as shown in Figure 65.

```

Start Query Management Query (STRQMORY)

Type choices, press Enter.

Query management query . . . . . > TESTQRY      Name
Library . . . . . > ITSCID41      Name, *LIBL, *CURLIB
Output . . . . . *                *, *PRINT, *OUTFILE
Query management report form . . *SYSDFT   Name, *SYSDFT, *QMORY
Library . . . . .                               Name, *LIBL, *CURLIB

Additional Parameters

Relational database . . . . . *NONE
Connection Method . . . . . *DUW           *DUW, *RUW
User . . . . . *CURRENT           Name, *CURRENT
Password . . . . .                               Character value, *NONE
Naming convention . . . . . *SYS          *SYS, *SAA
Allow information from QRYDFN . > *YES      *NO, *YES, *ONLY

                                                                    More...
F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys

```

Figure 65. Start Query Management Query

Use the STRQMORY CL command with the ALLQRYDFN parameter set to \*YES to enable it to exploit the pre-existing query definition:

```
STRQMORY QMORY (ITSCID41/TESTQRY) ALWQRYDFN(*YES)
```

- After the query has finished executing, you will notice message QWM2704 posted into your job log, as shown in Figure 66.

This is an informational message documenting that no query management object was found. However, a query definition having the same name existed and it was used, as allowed by the ALLQRYDFN(\*YES) parameter.

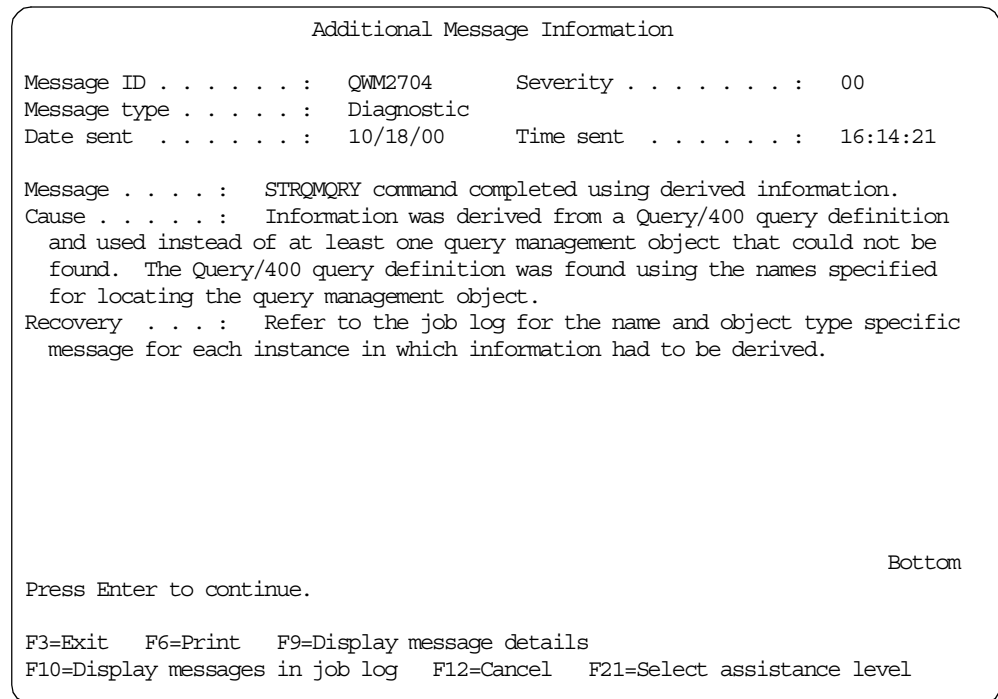


Figure 66. Message QWM704

- At this point, if you are finished collecting data, you can end the Database Monitor by using the CL command:

```
ENDDBMON
```

- Now you can use the SQL Performance Monitor tool in Operations Navigator to import and analyze the data collected. To do so, select **Database->SQL Performance Monitor**. Right-click **SQL Performance Monitor** and select **Import**. You see the dialog shown in Figure 67.

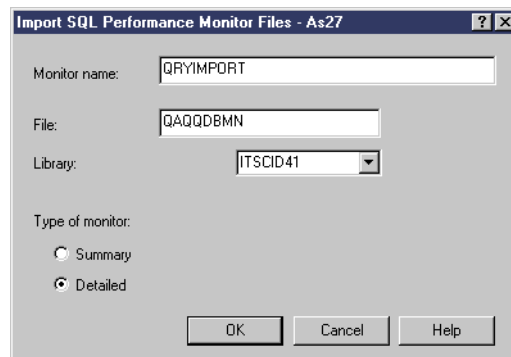


Figure 67. Importing data in SQL Performance Monitor

Here specify a name for the new Monitor you are creating and the file and library containing the previously collected data.

- The new monitor name is added to the list shown at the SQL Performance Monitor item. Right-click it and select **List Explainable Statements**, as shown in Figure 68.

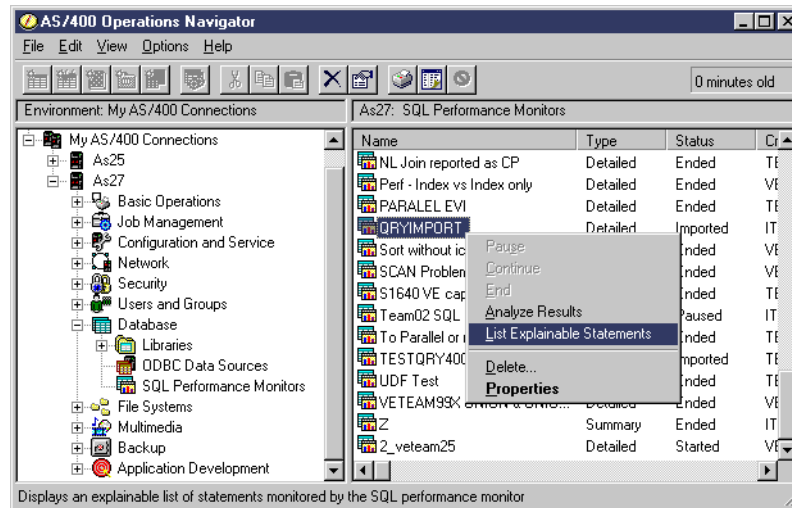


Figure 68. SQL Performance Monitor - List Explainable Statements

After you select this option, you are presented with the List Explainable Statements dialog (Figure 69). On this display, you can see the actual SQL statement generated by Query Manager to resolve the original Query/400 request.

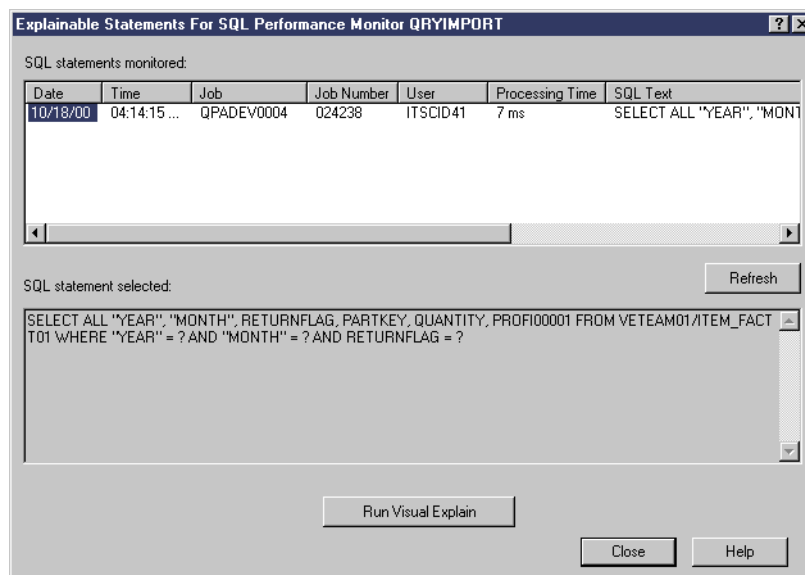


Figure 69. List Explainable Statements dialog

From here you can link into Visual Explain to analyze the database behavior during the execution of this query. For more information on this subject, refer to 1.9, "Visual Explain" on page 85.

---

## 1.9 Visual Explain

Visual Explain was added to the database component of Operations Navigator in V4R5. It provides a graphical representation of the optimizer implementation of a query request. The query request is broken down into individual components with icons representing each unique component. Visual Explain also includes information on the database objects that are considered and chosen by the query optimizer. Visual Explain's detailed representation of the query implementation makes it easier to understand where the greatest cost is being incurred.

The access plan details how the query is going to approach the retrieval of records from the tables involved in the query. The cost-based optimization uses statistics from the selection criteria and the tables being queried to generate a default cost. Then, for each index (until a time out), the optimizer materializes the index attributes and determines whether the index matches the selection criteria.

It then updates the selection criteria statistics and applies a cost to the index. If this cost is lower than the current best cost, the index is selected.

Visual Explain provides a graphical representation of the plan built by the Optimizer. It shows the tables accessed and any indexes used.

Where there is no index, it provides guidance as to whether an index could be beneficial, and if so, the criteria to use for index creation. Visual Explain can help you understand where the greatest overhead is being incurred.

Most queries use the SQL interface and generate an SQL statement, either directly (SQL Script Window, STRSQL command) or indirectly (QM/400).

Some other queries do not generate identifiable SQL statements (Query/400), and cannot be analyzed with the predefined reports of SQL Performance Monitor or Visual Explain. In this case, the name SQL as part of the SQL Performance Monitor is significant.

The statements that generate SQL and can be used with the SQL Performance Monitor include:

- SQL statements from the SQL Script Center
- SQL statements from the STRSQL command
- SQL statements processed by the RUNSQLSTM command
- SQL statements embedded into a high-level language program, such as Cobol, Java, or RPG
- SQL statements processed through an ODBC interface

The statements that do not generate SQL and, therefore, cannot be used with the SQL Performance Monitor include:

- Native database access from a high level language, for example, Cobol, RPG, etc.
- Query/400 access to the database
- OPNQRYF database access
- OS/400 Create Query API (QQQQRY)

However, it is sometimes possible to find workarounds that enable you to gather statistics for a product that does not normally generate SQL statements. Please refer to 1.8.3.1, "Importing performance data for Query/400" on page 81, for documentation on how to gather performance data for Query/400.

Table 4 shows the methods that are supported by Visual Explain at V4R5.

Table 4. Query access functions supported at V4R5M0

Optimizer access plan	Debug	Perf Monitor	Visual Explain
<b>Non-Keyed Access Methods</b>			
Table Scan	✓	✓	✓
Parallel Table Scan	✓	✓	✓
Parallel Pre-fetch	✓	✓	✓
Parallel Table Pre-load	✓	✓	✓
Skip Sequential with dynamic bitmap	✓	✓	✓
Parallel Skip Sequential	✓	✓	✓
<b>Keyed Data Access Methods</b>			
Key Positioning and Parallel Key Positioning	✓	✓	✓
Dynamic Bitmaps/Index ANDing ORing	✓	✓	✓
Key Selection and Parallel Key Selection	✓	✓	✓
Index-From-Index	✓	✓	✓
Index-Only Access	✓	✓	✓
Parallel Index Pre-load	✓	✓	✓
<b>Joining, Grouping, Ordering</b>			
Nested Loop Join	✓	✓	✓
Hash Join	✓	✓	✗
Index Grouping	✓	✓	✓
Hash Grouping	✓	✓	✓
Index Ordering	✓	✓	✓
Sort	✓	✓	✓
<b>Query Statements</b>			
Select	✓	✓	✓
Update	✓	✓	✓
Insert	✓	✓	✗
Delete	✓	✓	✓
Sub Query	✓	✓	✗
Union	✓	✓	✗

Optimizer access plan	Debug	Perf Monitor	Visual Explain
View Materialization	✓	✓	✗
<b>Operational Characteristics</b>			
Index Usage	✓	✓	✓
Index Advice	✓	✓	✓
Open Data Path Usage	✓	✓	✓
Work Management details	✗	✗	✓
<b>Notes:</b> ✓ - Supported for analysis with this method ✗ - Not supported for analysis with this method			

Be sure to load the latest V4R5 Database Group PTF to obtain the best functionality of Visual Explain.

Each of the methods shown in Table 4 provides assistance with the task of debugging queries and the analysis of queries to optimize their performance. None of the methodologies will actually carry out any changes for you. Their only purpose is to provide you with the necessary information to make an informed choice.

### 1.9.1 Navigating Visual Explain

The ease of use that Visual Explain brings to the table can disguise the fact that it is an advanced tool working for you in a highly technical area. Use Visual Explain to assist you with the task of enhancing query performance. Visual Explain cannot sort problems out for you. It can only help you identify and solve problems in a more effective way.

You still need to understand the process of query optimization, the different database access plans that can be implemented, and the effects of those plans on the system. You also need to have an understanding of the database that you are tuning, its use, and the impact of creating and changing indexes. For more information, refer to the manual *DB2 UDB for AS/400 Database Performance and Query Optimization*, which you can download from:

<http://publib.boulder.ibm.com/html/as400/infocenter.html>

Unlike the rest of the Database Tools provided by Operations Navigator, there is not a direct database item that lets you start Visual Explain. On the contrary, it has been linked in four different places, from where you can start Visual Explain (as you may have read in the previous discussion):

- Run SQL Script center (1.5.3.5, “Linking to the Visual Explain component” on page 56)
- Current SQL for a Job (1.7, “Current SQL for a Job” on page 62)
- SQL Performance Monitors (1.8.2, “Reviewing the SQL Performance Monitor results” on page 71)
- List Explainable SQL Statements (1.8.2.3, “List Explainable Statements” on page 78)

Any of these methods will take you to the Visual Explain panel shown in Figure 70. The Visual Explain graphics window is presented in two parts. The left-hand side of the display is called the “Query Implementation Graph”. This is the graphical representation of implementation steps of the SQL statement and the way that DB2 accesses the tables or indexes. The arrows indicate the order of the steps. Each node of the graph has a different image and represents an operation or values returned from an operation.

The right-hand side of the display has the Query Attributes and Values. The display is about the icon that has been selected on the graph. The default display first shown is for the final results icon. The vertical bar that separates the two sides is adjustable, and each side is individually scrolled and windowed.

The default settings cause the display to be presented with the final results icon (a checkered flag) on the left of the display, working toward the right of the display for the input to the query. Each of the icons on the display has a description and the estimated number of rows to be selected for each stage of the process.

When you click any of the icons, the right-hand description changes and presents the details that are known to the query for that part of the process. You may find it helpful to adjust the display to see more of the attributes and values.

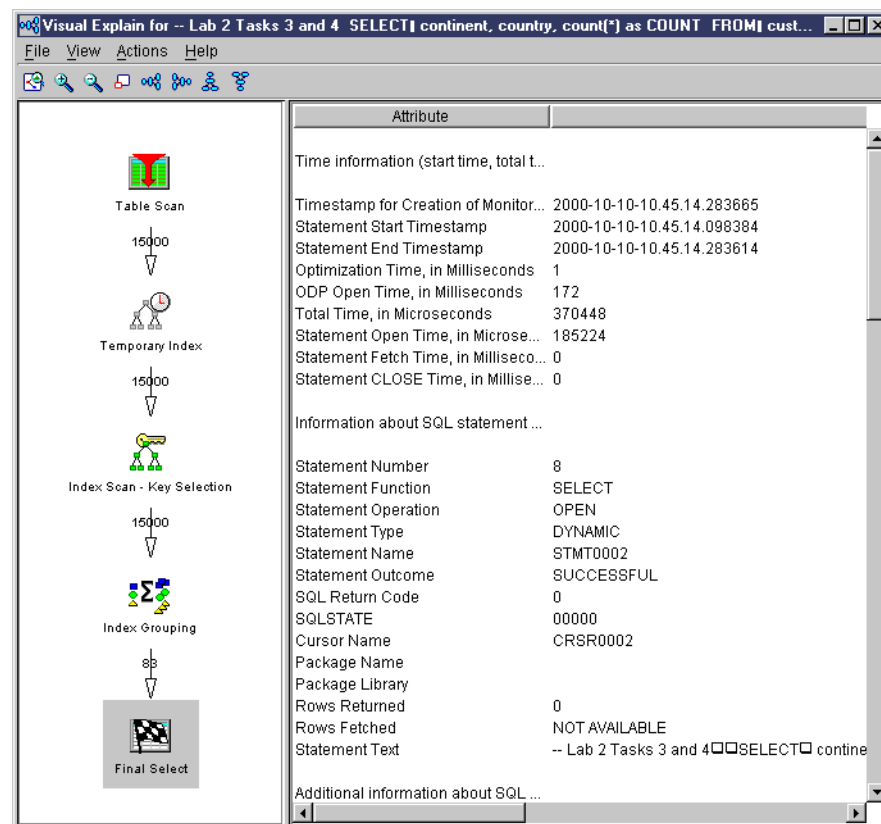


Figure 70. Visual Explain

If you right-click any of the icons on the display, you will find an action menu displayed. The action menu has options to assist with query information and can provide a shortcut to the table information to be shown in a separate window.

The following action menu items may be found selectively on different icons:



- **Table Description:** Displays table information returned by Display File Description (DSPFD).
- **Index Description:** Displays index information returned by DSPFD.
- **Create Index:** Creates a permanent index with the same characteristics as the temporary index.
- **Table Properties:** Displays object properties.
- **Index Properties:** Displays object properties.
- **Display Query Environment:** Displays the environment settings used during the processing of this query.

There are also flyover panels for many of the icons. By moving the mouse pointer over the icon, a window appears with summary information on the query step, as you can see in Figure 71.

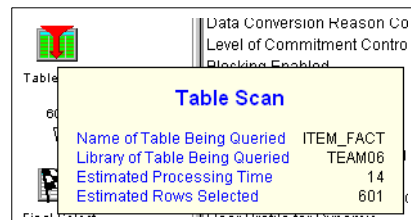


Figure 71. Summary information on query step

The Visual Explain tool bar (Figure 72) helps you navigate the displays. The first four icons (from left to right) help you control the sizing of the display. The left most icon scales the graphics to fit the main window. For many query implementations, this leaves the graphical display too small to be of value. The next two icons allow you to zoom in and out of the graphic image.

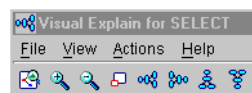


Figure 72. Visual Explain tool bar

The fourth icon (Overview) creates an additional window with the Visual Explain graphic shown on a reduced scale, as shown in Figure 73. This window has a highlighted area, which represents the part of the image that is currently displayed in the main window.

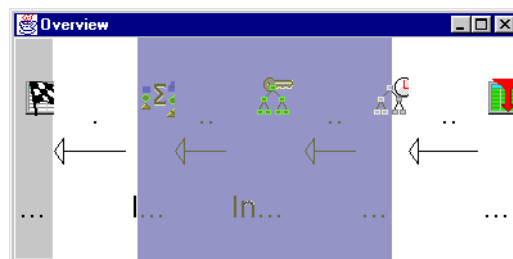


Figure 73. Visual Explain Overview window

In the Overview window, you can move the cursor into this highlighted area that is shown in the main window. The mouse pointer changes to enable you to drag the

highlighted area to change the section of the overall diagram that is shown in the main window.

The default schematic shows the query with the result on the left, working from right to left. The remaining four icons on the tool bar allow you to rotate the Visual Explain image. The icons are:

- Starting from the right, leading to the result on the left (default view)
- Starting from the left, leading to the result on the right
- Starting at the bottom, leading to the result at the top
- Starting from the top, leading to the result at the bottom

Try these icons to see which style of presentation you prefer. Be aware that each time you use Visual Explain, you will return to the default view.

### 1.9.2 Menu options

The menu options above the tool bar icons are File, View, and Actions. The File option allows you to close the window. The View options (Figure 74) generally replicate the tool bar icons. The additional options are:

- Icon spacing (horizontal or vertical) changes the size of the arrows between the icons.
- Arrow labels allow you to show/hide the estimated number of rows that the query is processing at each stage of the operation.
- Icon labels allow you to show/hide the description of the icons.

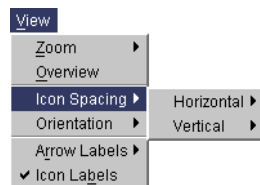


Figure 74. Visual Explain menu bar

The Actions menu item replicates features available on the display.

### 1.9.3 Visual Explain example

In this example, we are going to use Visual Explain to analyze an SQL statement previously collected with the SQL Performance Monitor tool, documented in 1.8.2.3, “List Explainable Statements” on page 78.

Click **Operations Navigator->Database->SQL Performance Monitors**. Then you see a list of the SQL Performance Monitors that are currently on the system. Right-click a *detailed* SQL Performance Monitors collection and select **List Explainable Statements**.

When you select this item, the panel shown in Figure 75 appears. In the upper half of the panel, all the explainable SQL statements are displayed that have been captured during the data collection session. Click to select the statement **1** you are interested in analyzing. This produces the selected statement in the bottom half of the dialog.

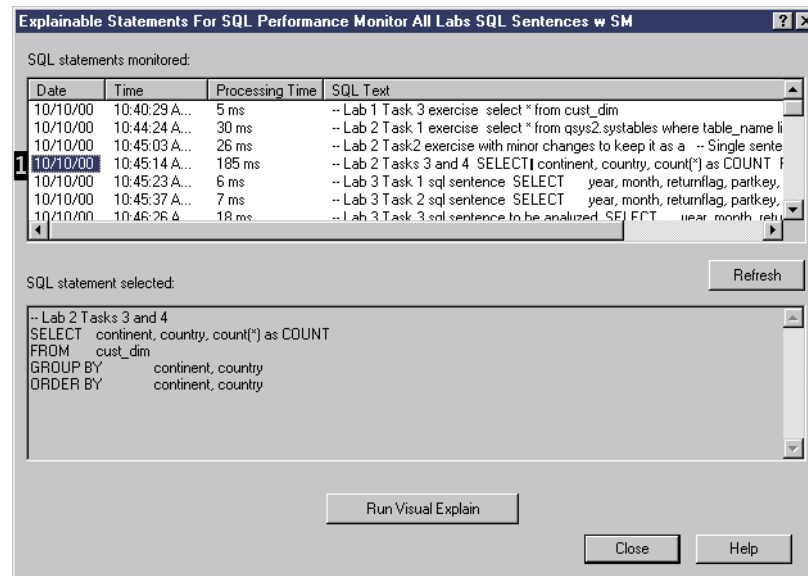


Figure 75. Starting Visual Explain

In our example, we are working on the following statement:

```
SELECT continent, country, count(*) as COUNT
FROM cust_dim
GROUP BY continent, country
ORDER BY continent, country
```

Once the statement is in focus, it is possible to have it analyzed and explained. Click the **Run Visual Explain** button at the bottom of the panel. Visual Explain is a new tool to V4R5 and is entirely written in Java. It may take a few seconds to start. The resources available in your PC affect the startup (processor feature and amount of memory available can greatly influence this).

In Figure 76 on page 92, you can see the Visual Explain panel for this statement. Here the icon view has been changed to *orient bottom*, using the right-most icon on the icon tool bar, to obtain a better picture for this screen capture. For a detailed discussion on this panel layout, its menu options, and icon bar, refer to 1.9, "Visual Explain" on page 85.

Looking at the icons in the left hand side of this panel, it is easy to see how the Optimizer implemented a plan to access database when the SQL statement was run.

Each icon documents a different step:

- Table Scan
- Temporary Index
- Index scan - Key Selection
- Index Grouping
- Final Select

When you click each of the icons, notice that the right-hand side of the panel changes to provide information on the selected step. In Figure 76 on page 92, we captured the screen while the Final Select icon was highlighted. Therefore, the information relates to the overall results.

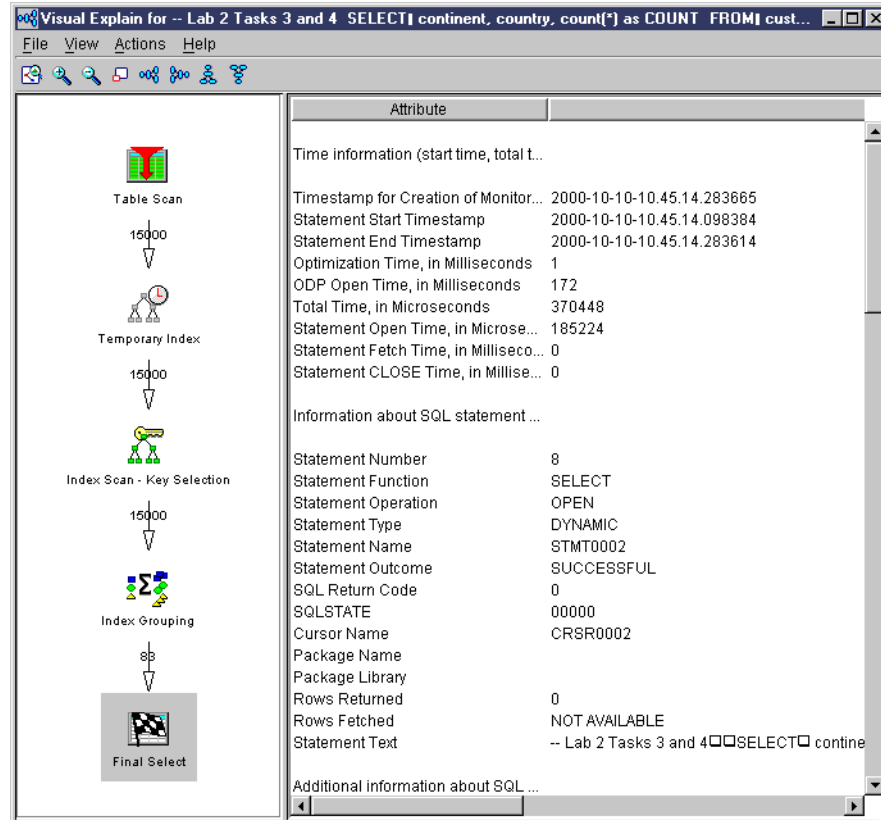


Figure 76. Visual Explain results

If you simply move the mouse pointer on any of these icons, you see a pop-up message box with summary information, as shown in Figure 77.

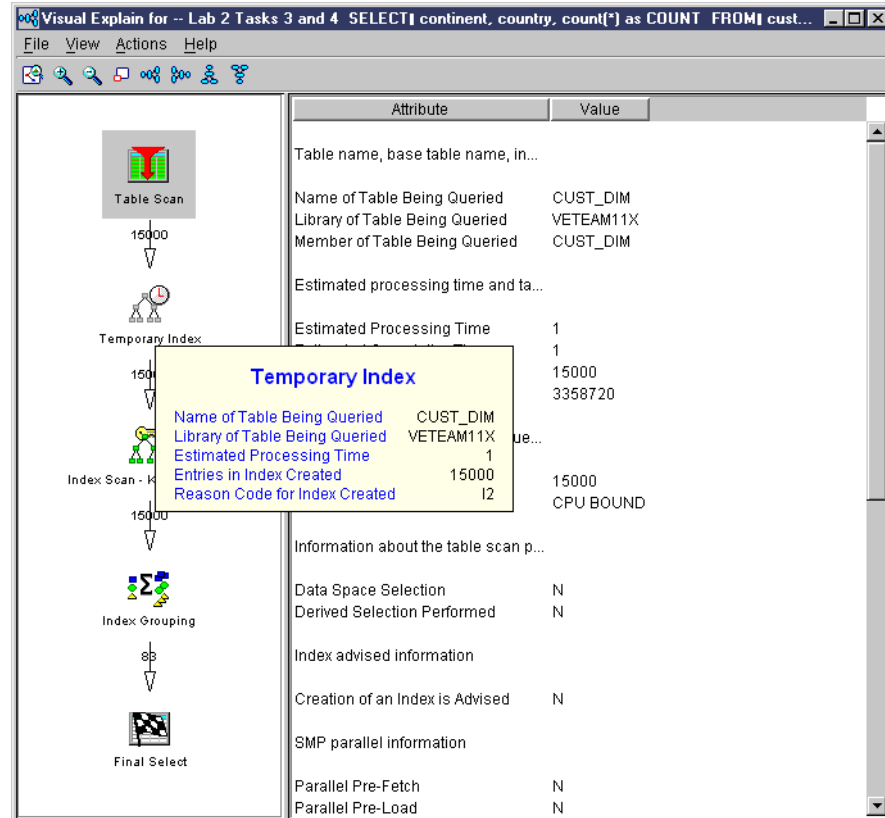


Figure 77. Visual Explain icon caption

Considering the SQL statement used in this example, both the GROUP BY and ORDER BY clauses can use the same index to resolve the grouping and ordering. If you are going to run this or a similar statement on a regular basis, it makes sense to create a permanent index.

Visual Explain is going to help with this task. When you right-click the icon representing the temporary index creation (Figure 78), you see a list of options.

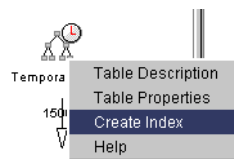


Figure 78. Index options

This list varies accordingly to the steps and icons involved. In our case, with the Temporary Index icon, we can select Create Index to permanently build an index with the same characteristics as the temporary one that was automatically built by the database to resolve this query. Table Description and Table Properties link you into the same functions that you obtain when you right-click a table and select the above mentioned items. For examples, refer to 1.3.3.3, “Table Description example” on page 29, and 1.3.3.4, “Table Properties example” on page 31.

Visual Explain is an easy and user-friendly interface to a complex activity – database tuning. Although this tool may make it easier for non-experts to

understand database access, some specialized skills are still needed to perform database tuning. An in-depth discussion on database tuning is well beyond the goals of this redbook.

#### 1.9.4 The Visual Explain icons

The icons that you may encounter on the Visual Explain query implementation chart are shown here:



The **Final Result** icon displays the original text and summary information of how the query was implemented.



The **Table Scan** icon indicates that all rows in the table were paged in and selection criteria was applied against each row. Only those rows meeting the selection criteria were retrieved. To place the result in a particular sequence, you must specify the ORDER BY clause.



The **Parallel Table Scan** icon indicates that a table scan access method was used and multiple tasks were used to fill the rows in parallel. The table was partitioned, and each task was given a portion of the table to work with.



The **Skip Sequential Table Scan** icon indicates that a bitmap was used to determine which rows would be selected. No CPU processing was done on non-selected rows and I/O was minimized by bringing in only those pages that contained rows to be selected.



The **Skip Sequential Parallel Table Scan** icon indicates that a skip sequential table scan access method was used and multiple tasks were used to fill the rows in parallel. The table was partitioned, and each task was given a portion of the table to work with.



The **Derived Column Selection** icon indicates that a column in the row selected had to be mapped or derived before selection criteria could be applied against the row. Derived column selection is the slowest selection method.



The **Parallel Derived Column Selection** icon indicates that Derived Field Selection was performed, and this processing was accomplished using multiple tasks. The table was partitioned, and each task was given a portion of the table to work with.

---



The **Index Key Positioning** icon indicates that only entries of the index that match a specified range of key values were paged in. The range of key values was determined by the selection criteria whose predicates matched the key columns of the index. Only selected key entries were used to select rows from the corresponding table data.



The **Parallel Index Key Positioning** icon indicates that multiple tasks were used to perform the key positioning in parallel. The range of key values was determined by the selection criteria whose predicates matched the key columns of the index. Only selected key entries were used to select rows from the corresponding table data.



The **Index Key Selection** icon indicates that all entries of the index were paged in. Any selection criteria, whose predicates match the key columns of the index, were applied against the index entries. Only selected key entries were used to select rows from the table data.



The **Parallel Index Key Selection** icon indicates that multiple tasks were used to perform key selection in parallel. The table was partitioned, and each task was given a portion of the table to work with.



The **Encoded Vector Index** icon indicates that access was provided to a database file by assigning codes to distinct key values and then representing these values in an array (vector). Because of their compact size and relative simplicity, encoded vector indexes provide for faster scans.



The **Parallel Encoded Vector Index** icon indicates that multiple tasks were used to perform the encoded vector index selection in parallel. This allows for faster scans that can be more easily processed in parallel.



The **Sort Sequence** icon indicates that selected rows were sorted using a sort algorithm.



The **Grouping** icon indicates that selected rows were grouped or summarized. Therefore, duplicate rows within a group were eliminated.



The **Nested Loop Join** icon indicates that queried tables were joined together using a nested loop join implementation. Values from the primary file were joined to the secondary file by using an index whose key columns matched the specified join columns.



The **Hash Join** icon indicates that a temporary hash table was created. The tables that were queried were joined together using a hash join implementation where a hash table was created for each secondary table. Therefore, matching values were hashed to the same hash table entry.



The **Temporary Index** icon indicates that a temporary index was created, because the query either requires an index and one does not exist or the creation of an index will improve performance of the query.



The **Temporary Hash Table** icon indicates that a temporary hash table was created to perform hash processing.



The **Temporary Table** icon indicates that a temporary table was required to either contain the intermediate results of the query, or the queried table could not be queried as it currently exists and a temporary table was created to replace it.



The **Dynamic Bitmap** icon indicates that a bitmap was dynamically generated from an existing index. It was then used to determine which rows were to be retrieved from the table. To improve performance, dynamic bitmaps can be used in conjunction with a table scan access method for skip sequential or with either the index key position or key selection.



The **Bitmap Merge** icon indicates that multiple bitmaps were merged or combined to form a final bitmap. The merging of the bitmaps simulates boolean logic (AND/OR selection).



The **DISTINCT** icon indicates that duplicate rows in the result were prevented. You can specify that you do not want any duplicates by using the **DISTINCT** keyword, followed by the selected column names.



The **UNION Merge** icon indicates that the results of multiple subselects were merged or combined into a single result.



The **Subquery Merge** icon indicates that the nested **SELECT** was processed for each row (**WHERE** clause) or group of rows (**HAVING** clause) selected in the outer level **SELECT**. This is also referred to as a *correlated subquery*.





The **Incomplete Information** icon indicates that a query could not be displayed due to incomplete information.

---



---

## Appendix A. Special notices

This publication is intended to help programmers, analysts, and database administrators to implement and manage DB2 UDB for AS/400. See the PUBLICATIONS section of the IBM Programming Announcement for OS/400 V4R5 and Client Access Express for Windows V4R5 for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785.


Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.


Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

e (logo)®   
 IBM ®  
 AFP  
 AS/400  
 AT  
 Current

Redbooks  
 Redbooks Logo   
 APPN  
 AS/400e  
 CT  
 DB2

DB2 Universal Database	Distributed Relational Database Architecture
DRDA	IBM ®
IBM Payment Server	Manage. Anything. Anywhere.
Netfinity	Network Station
OfficeVision	OfficeVision/400
Operating System/400	OS/2
OS/400	PartnerWorld
RACF	RS/6000
Service Director	SP
SP1	SP2
System/36	System/390
WebSphere	Wizard
XT	400
Lotus	cc:Mail
Lotus Notes	Domino
Notes	Tivoli
TME	NetView
Cross-Site	Tivoli Ready
Tivoli Certified	Planet Tivoli

The following terms are trademarks of other companies:

Tivoli, Manage. Anything. Anywhere., The Power To Manage., Anything. Anywhere., TME, NetView, Cross-Site, Tivoli Ready, Tivoli Certified, Planet Tivoli, and Tivoli Enterprise are trademarks or registered trademarks of Tivoli Systems Inc., an IBM company, in the United States, other countries, or both. In Denmark, Tivoli is a trademark licensed from Kjøbenhavns Sommer - Tivoli A/S.

C-bus is a trademark of Corollary, Inc. in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

PC Direct is a trademark of Ziff Communications Company in the United States and/or other countries and is used by IBM Corporation under license.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through The Open Group.

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

Other company, product, and service names may be trademarks or service marks of others.

---

## Appendix B. Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

---

### B.1 IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 107.

- *iSeries Handbook*, GA19-5486
- *IBM AS/400 Printing V*, SG24-2160
- *The System Administrator's Companion to AS/400 Availability and Recovery*, SG24-2161
- *AS/400 Internet Security: IBM Firewall for AS/400*, SG24-2162
- *DB2/400 Advanced Database Functions*, SG24-4249
- *Understanding LDAP*, SG24-4986
- *LDAP Implementation Cookbook*, SG24-5110
- *AS/400 TCP/IP Autoconfiguration: DNS and DHCP Support*, SG24-5147
- *Lotus Domino for AS/400: Installation, Customization, and Administration*, SG24-5181
- *DB2/400: Mastering Data Warehousing Functions*, SG24-5184
- *V4 TCP/IP for AS/400: More Cool Things Than Ever*, SG24-5190
- *AS/400 Client Access Express for Windows: Implementing V4R4M0*, SG24-5191
- *The AS/400 NetServer Advantage*, SG24-5196
- *Lotus Domino for AS/400: Integration with Enterprise Applications*, SG24-5345
- *IBM Firewall for AS/400 V4R3: VPN and NAT Support*, SG24-5376
- *AS/400 Internet Security: Implementing AS/400 Virtual Private Networks*, SG24-5404
- *Management Central: A Smart Way to Manage AS/400 Systems*, SG24-5407
- *DB2 UDB for AS/400 Object Relational Support*, SG24-5409
- *Developing Cross-Platform DB2 Stored Procedures: SQL Procedures and the DB2 Stored Procedure Builder*, SG24-5485
- *Lotus Domino for AS/400 R5: Implementation*, SG24-5592
- *AS/400 Internet Security: Developing a Digital Certificate Infrastructure*, SG24-5659
- *IBM Network Station Manager V2R1*, SG24-5844

The following redbooks can only be accessed online as softcopy publications at the redbooks Web site at: <http://www.redbooks.ibm.com>

At the site, enter the order number in the Search field and click **Go**. Then, click on the publication you wish to view.

- *AS/400 - IBM Network Station - Getting Started*, SG24-2153
- *Exploring NFS on AS/400*, SG24-2158

---

## B.2 IBM Redbooks collections

Redbooks are also available on the following CD-ROMs. Click the CD-ROMs button at [ibm.com/redbooks](http://ibm.com/redbooks) for information about all the CD-ROMs offered, updates and formats.

CD-ROM Title	Collection Kit Number
IBM System/390 Redbooks Collection	SK2T-2177
IBM Networking Redbooks Collection	SK2T-6022
IBM Transaction Processing and Data Management Redbooks Collection	SK2T-8038
IBM Lotus Redbooks Collection	SK2T-8039
Tivoli Redbooks Collection	SK2T-8044
IBM AS/400 Redbooks Collection	SK2T-2849
IBM Netfinity Hardware and Software Redbooks Collection	SK2T-8046
IBM RS/6000 Redbooks Collection	SK2T-8043
IBM Application Development Redbooks Collection	SK2T-8037
IBM Enterprise Storage and Systems Management Solutions	SK3T-3694

---

## B.3 Other resources

These publications are also relevant as further information sources:

- *HTTP Server for AS/400 Quick Beginnings*, GC41-5433
- *HTTP Server for AS/400 Webmaster's Guide*, GC41-5434
- *IBM Network Station Manager for AS/400*, SC41-0632
- *Ultimedia System Facilities Installation and Administration*, SC41-4540
- *Ultimedia System Facilities User Guide V3R6*, SC41-4541
- *Ultimedia System Facilities Programming*, SC41-4652
- *AS/400 National Language Support*, SC41-5101
- *Getting Your AS/400 Working for You*, SC41-5161
- *AS/400 Basic System Operation, Administration, and Problem Handling*, SC41-5206
- *Query Manage Use*, SC41-5212
- *AS/400 Security - Basic*, SC41-5301
- *OS/400 Security - Reference*, SC41-5302
- *Security - Enabling for C2*, SC41-5303
- *OS/400 Backup and Recovery*, SC41-5304
- *OS/400 Work Management*, SC41-5306
- *Distributed Data Management*, SC41-5307
- *Performance Tools/400 V4R2*, SC41-5340
- *AS/400 Communications Configuration*, SC41-5401
- *AS/400 Remote Work Station Support*, SC41-5402
- *AS/400 Communications Management*, SC41-5406

- *APPN Support*, SC41-5407
- *TCP/IP Configuration and Reference*, SC41-5420
- *TCP/IP Fastpath Setup*, SC41-5430
- *APPC Programming*, SC41-5443
- *Client Access Express for Windows - Setup*, SC41-5507
- *AS/400 Client Access Express for Windows ODBC User's Guide V4*, SC41-5509
- *DB2 UDB for AS/400 SQL Programming*, SC41-5611
- *DB2 UDB for AS/400 SQL Reference*, SC41-5612
- *Database Programming*, SC41-5701
- *Distributed Database Programming*, SC41-5702
- *Query Management Programming*, SC41-5703
- *DB2 Multisystem for AS/400*, SC41-5705
- *IFS (Integrated File System) Introduction*, SC41-5710
- *Integrated File System Introduction*, SC41-5711
- *DDS Reference*, SC41-5712
- *OS/400 CL Programming*, SC41-5721
- *OS/400 CL Reference*, SC41-5722
- *Client Access Express Host Servers*, SC41-5740
- *SQL Call Level Interface (CLI) to ODBC*, SC41-5806

The following publications are available in soft copy format from the AS/400 Information Center at: <http://www.as400.ibm.com/infocenter>

Once you reach this site, select **Online library**. Select a language, and click **GO**. Select a release. Select **Search or view**. Type the number of the book you need, and click **Find**.

- *HTTP Server for AS/400 Quick Beginnings*, GC41-5433
- *HTTP Server for AS/400 Webmaster's Guide*, GC41-5434
- *Query Manage Use*, SC41-5212
- *Distributed Data Management*, SC41-5307
- *APPN Support*, SC41-5407
- *APPC Programming*, SC41-5443
- *Distributed Database Programming*, SC41-5702
- *Query Management Programming*, SC41-5703
- *DB2 Multisystem for AS/400*, SC41-5705
- *IFS (Integrated File System) Introduction*, SC41-5710
- *Integrated File System Introduction*, SC41-5711
- *DDS Reference*, SC41-5712
- *OS/400 CL Reference*, Parts 1-4: SC41-5723, SC41-5724, SC41-5725, SC41-5726, or the entire book as SC41-5722

- *SQL Call Level Interface (CLI) to ODBC*, SC41-5806

From this same site, you can select **Database and File Systems-> Database management**. Under Database management, by selecting **DB2 Universal Database for AS/400 books online**, you can find a list of publications that contain additional information, including:

- *Distributed Data Management*, SC41-5307
- *DB2 UDB for AS/400 SQL Programming*, SC41-5611
- *DB2 UDB for AS/400 SQL Reference*, SC41-5612
- *Database Programming*, SC41-5701

This book describes database capabilities, primarily outside of SQL terminology. This includes physical files (correspond to SQL tables), logical files (correspond to SQL views), fields (correspond to SQL columns), records (correspond to SQL rows), file management, and file security.

---

## B.4 Referenced Web sites

These Web sites are also relevant as further information sources:

- To download a set of AS/400 database lab exercises, visit the Web site at:  
<http://www.as400.ibm.com/developer>  
Select **Education->Internet Based Offerings->DB2 UDB->Piloting DB2 UDB for AS/400 with Operations Navigator**.
- AS/400 NetServer home page: <http://www.as400.ibm.com/netserver>
- AS/400 Directory Services (LDAP): <http://www.as400.ibm.com/ldap>
- AS/400 Information Center and Technical Studio Web sites:  
<http://www.as400.ibm.com/infocenter>  
<http://www.as400.ibm.com/tstudio>
- AS/400 Operations Navigator Plug-in Support white paper:  
[http://www.as400.ibm.com/oper\\_nav/pluginwpaper.htm](http://www.as400.ibm.com/oper_nav/pluginwpaper.htm)
- AS/400 Operations Navigator Plug-In Support Web page:  
[http://www.as400.ibm.com/oper\\_nav/pluginpage.htm](http://www.as400.ibm.com/oper_nav/pluginpage.htm)
- Operations Navigator Plug-In Support, IBM Technical Studio document:  
<http://www.as400.ibm.com/tstudio/opsnav/plugin/pludex.htm>
- News/400 Web site (search for Operations Navigator articles):  
<http://www.news400.com>
- Midrange computing Web site (search for Operations Navigator articles):  
<http://www.midrangecomputing.com>
- Midrange systems Web site (search for Operations Navigator articles):  
<http://www.midrangesystems.com>
- MIDRANGE dot COM Web site (search for Operations Navigator articles):  
<http://www.midrange.com>
- Find a service packs link on the Client Access Web site:  
<http://www.as400.ibm.com/clientaccess>
- The AS/400 Information Center Web site is at:  
<http://www.as400.ibm.com/infocenter>



- Register for, and use, the iPTF facility at: <http://www.as400service.ibm.com>
- For details on implementing C2 level security on OS/400, refer to the IBM AS/400 in the Evaluated Product List (EPL) at the US Government Web site at: <http://www.radium.ncsc.mil/tpep/epl/epl-by-vendor.html>
- The IBM Web-only version of the security wizard that you can go through without actually changing the security setting on your system:  
[http://www.as400.ibm.com/tstudio/secure1/index\\_av.htm](http://www.as400.ibm.com/tstudio/secure1/index_av.htm)



## How to get IBM Redbooks

This section explains how both customers and IBM employees can find out about IBM Redbooks, redpieces, and CD-ROMs. A form for ordering books and CD-ROMs by fax or e-mail is also provided.

- **Redbooks Web Site** [ibm.com/redbooks](http://ibm.com/redbooks)

Search for, view, download, or order hardcopy/CD-ROM Redbooks from the Redbooks Web site. Also read redpieces and download additional materials (code samples or diskette/CD-ROM images) from this Redbooks site.

Redpieces are Redbooks in progress; not all Redbooks become redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

- **E-mail Orders**

Send orders by e-mail including information from the IBM Redbooks fax order form to:

	<b>e-mail address</b>
In United States or Canada	<a href="mailto:pubscan@us.ibm.com">pubscan@us.ibm.com</a>
Outside North America	Contact information is in the "How to Order" section at this site: <a href="http://www.elink.ibm.ibm.com/pbl/pbl">http://www.elink.ibm.ibm.com/pbl/pbl</a>

- **Telephone Orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	Country coordinator phone number is in the "How to Order" section at this site: <a href="http://www.elink.ibm.ibm.com/pbl/pbl">http://www.elink.ibm.ibm.com/pbl/pbl</a>

- **Fax Orders**

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	Fax phone number is in the "How to Order" section at this site: <a href="http://www.elink.ibm.ibm.com/pbl/pbl">http://www.elink.ibm.ibm.com/pbl/pbl</a>

This information was current at the time of publication, but is continually subject to change. The latest information may be found at the Redbooks Web site.

### IBM Intranet for Employees

IBM employees may register for information on workshops, residencies, and Redbooks by accessing the IBM Intranet Web site at <http://w3.itso.ibm.com/> and clicking the ITSO Mailing List button. Look in the Materials repository for workshops, presentations, papers, and Web pages developed and written by the ITSO technical professionals; click the Additional Materials button. Employees may access MyNews at <http://w3.ibm.com/> for redbook, residency, and workshop announcements.



## Index

### Symbols

\*FILE object 3

### A

access path 25  
 add column 31  
 Advised Index 74  
 alias 12, 28  
 ALTER TABLE 32  
 Analyze Results 73  
 analyzing SQL Performance Monitor results 73  
 ASP 11  
 auxiliary storage pool 11

### C

catalog 11  
 CCSID 32, 44  
 change properties caution 31  
 Change Query Attributes 7  
 check constraint 34  
 CL command 49  
 Coded Character Set Identifier 32, 44  
 collection 10, 11  
 command 26  
 commit 42  
 commit mode 42  
 commitment control 42  
 constraints 26, 33, 34  
 constraints tips 34  
 copy 28  
 create journal 22  
 Create Physical File 14  
 CREATE TABLE 3  
 CREATE VIEW 3, 5  
 CRTJRNRCV 26  
 CRTLF 3, 5  
 CRTPF 3, 32  
 cut 28

### D

data dictionary 11  
 data source translation 44  
 Database administration 1  
 Database functions 7  
 Database Library functions 8  
 debug mode 64  
 default libraries 42  
 delete 28, 35  
 delete column 31  
 delete rows 28  
 deleted record 14  
 Distributed Relational Database Architecture (DRDA) 48  
 DLTPCT 14  
 DRDA (Distributed Relational Database Architecture) 48, 49

### E

edit recovery for access path 25  
 Edit SQL 22  
 edit SQL 21  
 edit SQL tip 22

### F

field level authority 5  
 function, user defined 12

### I

Include Debug Messages in Job Log 57  
 Include Error Message Help in Run History 57  
 index 12, 25, 26, 33  
 insert rows 28

### J

JOIN statement 58  
 journal 11, 12, 22, 26  
 journal entry 23  
 journal example 22  
 journal internal entries 25  
 journal receiver 23, 24, 26  
 journals 12

### K

key constraints 33

### L

lab exercise 74  
 level check 31  
 libraries 9  
 library 10, 11  
 library name 51  
 library-based functions 12  
 LVLCHK 31

### M

maximum field data returned 43  
 maximum members 14  
 member size 14  
 Modify Selected Queries 73

### N

new journal receiver 35

### O

ODBC data source format parameters 43  
 ODBC data source server parameters 41  
 ODBC data sources (IBM provided) 40  
 Open 27, 35

**P**

performance collection files 45  
permissions 12, 26  
physical file 14  
procedure 12  
programs 12  
properties 12, 35

**Q**

QBATCH 50  
Query Optimizer 64  
quick view 27

**R**

referential constraints 33  
remote journal 23, 26, 35  
reorganize 28  
reorganize file/table 15  
REUSEDLT 14  
rollback 42  
Run History pane 47  
Run SQL Scripts 7, 10, 28, 45  
Running CL in SQL tips 51

**S**

self study lab 3  
SMAPP 25  
Smart Statement Selection 57  
SQL 70  
SQL collection 11  
SQL CONNECT 48  
SQL naming convention (operational difference) 44  
SQL software requirements 6  
SQL VIEW example 17  
starting and ending journaling 35  
starting the SQL Performance Monitor 66  
Stop on Error 57  
Submit Job 50  
swap receivers 35  
system naming convention, operational difference 44

**T**

table 5, 12, 26  
triggers 26, 33  
types 12

**U**

update row 28  
User Defined Functions (UDF) 12  
user defined types 12

**V**

view 12  
View Results button 73  
views 12

## IBM Redbooks review

Your feedback is valued by the Redbook authors. In particular we are interested in situations where a Redbook "made the difference" in a task or problem you encountered. Using one of the following methods, **please review the Redbook, addressing value, subject matter, structure, depth and quality as appropriate.**

- Use the online **Contact us** review redbook form found at [ibm.com/redbooks](http://ibm.com/redbooks)
- Fax this form to: USA International Access Code + 1 845 432 8264
- Send your comments in an Internet note to [redbook@us.ibm.com](mailto:redbook@us.ibm.com)

<b>Document Number</b>	SG24-5993-00
<b>Redbook Title</b>	DB2 UDB for AS/400: Database Administration with Operations Navigator V4R5
<b>Review</b>	
<b>What other subjects would you like to see IBM Redbooks address?</b>	
<b>Please rate your overall satisfaction:</b>	<input type="radio"/> Very Good <input type="radio"/> Good <input type="radio"/> Average <input type="radio"/> Poor
<b>Please identify yourself as belonging to one of the following groups:</b>	<input type="radio"/> Customer <input type="radio"/> Business Partner <input type="radio"/> Solution Developer <input type="radio"/> IBM, Lotus or Tivoli Employee <input type="radio"/> None of the above
<b>Your email address:</b> The data you provide here may be used to provide you with information from IBM or our business partners about our products, services or activities.	<input type="checkbox"/> Please do not use the information collected here for future marketing or promotional contacts or other communications beyond the scope of this transaction.
<b>Questions about IBM's privacy policy?</b>	The following link explains how we protect your personal information. <a href="http://ibm.com/privacy/yourprivacy/">ibm.com/privacy/yourprivacy/</a>





To determine the spine width of a book, you divide the paper PPI into the number of pages in the book. An example is a 250 page book using Plainfield opaque 50# smooth which has a PPI of 526. Divided 250 by 526 which equals a spine width of .4752". In this case, you would use the .5" spine. Now select the Spine width for the book and hide the others: Special>Conditional Text>Show/Hide>SpineSize(-->Hide:)>Set

Draft Document for Review January 9, 2001 4:45 pm

**59933spine.fm 113**



## DB2 UDB for AS/400: Database Administration with Operations Navigator V4R5

(0.2"spine)  
0.17"-->0.473"  
90-->249 pages

To determine the spine width of a book, you divide the paper PPI into the number of pages in the book. An example is a 250 page book using Plainfield opaque 50# smooth which has a PPI of 526. Divided 250 by 526 which equals a spine width of .4752". In this case, you would use the .5" spine. Now select the Spine width for the book and hide the others: Special>Conditional Text>Show/Hide>SpineSize(-->Hide:)>Set

Draft Document for Review January 9, 2001 4:45 pm

**59933spine.fm 114**





# DB2 UDB for AS/400

## Database Administration with Operations Navigator V4R5



**Discover how to manage DB2 UDB for AS/400 using Operations Navigator**

**Learn how to use the SQL Scripting Center to run SQL statements**

**Use the latest Visual Explain graphical tool for SQL performance analysis**

Operations Navigator offers a Windows-like graphical interface to configure, monitor, and manage the OS/400 environment. This redpiece gives you insight into the wide range of DB2 UDB for AS/400 database administration functions available through the AS/400 Operations Navigator graphical interface, which comes packaged with AS/400 Client Access Express for Windows V4R5M0.

This redpiece is an update of a Chapter 11 from the existing IBM redbook *Managing AS/400 V4R4 with Operations Navigator*, SG24-5646. It includes and updates all the information, graphics, and screens with the V4R5 features. Apart from the updates to Chapter 11 provided in this redpiece, the remainder of the redbook *Managing AS/400 V4R4 with Operations Navigator*, SG24-5646, continues to support the features of V4R4M0.

With the release of V4R5, Operations Navigator has been enhanced to improve the manageability of DB2 UDB for AS/400. The major additions to this update were DSPFD type of output for tables, views, and indexes. But perhaps the biggest enhancement is the addition of Visual Explain to the Operations Navigator database toolset. Visual Explain is a tool that graphically displays the query optimizer's implementation of a query request.

### **INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION**

### **BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:**  
[ibm.com/redbooks](http://ibm.com/redbooks)