A Merlon White Paper

258 Adelaide Street East # 404

Toronto ON

Canada

416-504-8727

www.merlon.com

info@merlon.com

NonStop SQL/MX Manageability Tools

The Case For SQLXPress

John Furlong

Project Leader

SQLXPress Development

Table of Contents

NonStop SQL/MX Manageability Tools

Introduction

Since you are reading this, you are almost certainly trying to decide which tool you should choose to manage your NonStop SQL/MX databases.

You have probably realized already that using only command line tools is not a very satisfactory approach. The advantages of a Graphical User Interface (GUI) are just too compelling to be ignored. (Otherwise the world would still be using DOS instead of Windows.)

You have three choices:

1. The GUI tools that are shipped as part of SQL/MX: The SQL/MX Database Manager (MXDM), and the Visual Query Planner (VQP).
2. SQLXPress
3. Other third-part tools that support MX via ODBC / JDBC, for example DB Visualizer.

To The Reader

If you are a management type, you will probably want to skip the In-Depth Comparison of product features, it will make your eyes glaze over. Ask your technical folks for their opinion. The Summary of Conclusions is aimed at you.

If you are a technical type, you will get the most out of this paper if you have already tried SQLXPress for yourself. You can ask for an evaluation license by contacting us at info@merlon.com. The In-Depth Comparison section is aimed at you.

Summary of Conclusions

The bottom line:

- **MXDM / VQP:** There is no additional cost to use these (they are bundled with SQL/MX).

  They provide a minimal set of features with a so-so user interface. Many features are simply "do it with a GUI" where filling in a form takes the place of typing a command.

- **SQLXPress:** There is an additional cost. How much depends on your NonStop system(s) configuration. There are a number of pricing options available designed to suit specific customer's requirements.

  SQLXPress provides an extensive set of features, with a rich user interface. There are many value-added features that are either not available on the command line, or if available, are hard to use or are very labour-intensive.

  SQLXPress is the only NonStop SQL manageability tool that provides a complete audit of user activity.

- **Generic ODBC/JDBC based tools:** These tend to be priced at mass-market levels, usually a few hundred dollars per user.

  The downside to these tools is the lack of support for NonStop SQL specific extensions. For example no support for physical database extensions like file attributes, partitioning, file opens, file locks etc.; no support for Execution Plans; no support for MP etc.

  These generic tools are not really serious contenders due to their lack of NonStop SQL specific features.

A little more detail:

Here is a list of the criteria I have used to assess each option:

- **Cost:** Pretty obvious, how much does it cost?
- **Breadth:** This is about the scope of the product. What features are available?
- **Depth:** This is about how well a feature is implemented. Does it go above and beyond the fill-in-the-form GUI alternative to the command line?
- **User Interface:** Not all Graphical User Interfaces are created equal. A good GUI design will improve productivity, reduce training costs, and encourage user to use the product more effectively.

Here are the pros and cons of each choice:

MXDM and VQP

These comments are based on the SQL/MX 3.3 release.

Pros:

- **Cost:** The cost of these tools is "baked into" the cost of SQL/MX. You pay for them as part of the license fee for SQL/MX.
- **User Interface:** They both provide a Graphical User Interface which provides an advantage over command line tools such as MXCI. However, you'll also find User Interface under "Cons", below.

Cons:

- **Lack of Breadth:** There are many routine tasks that are not supported. For these, the user has to revert back to command line tools like MXCI, FUP etc

Some examples of features that are **not supported** include:

- Audit
- Update Statistics.
- Create / Alter / Drop / Rename support for Views, Triggers, Procedures and Sequences.
- Partition Management (Move / Split / Reuse / Drop etc.).
- Security Management (Grant / Revoke / Give).
- Utilities (DUP, RELOAD, Populate Index, List Locks, List Opens etc.)
- Ability to execute long-running operations (such as update statistics, modify table, populate index, etc) as a background task.
- Support For SQL/MP: If you are an MX-only shop, this is not an issue. However, if you have legacy SQL//MP databases, you can't manage them with MXDM. Also, VQP does not support SQL/MP.

- **Lack of Depth:** In my opinion, both MXDM and VQP provide what I call a "bare minimum" GUI implementation. Essentially, this is a GUI that, for the most part, provides a "fill in the form" alternative to typing a command. It lets you specify things using pick-lists, check-boxes, radio buttons, and all the other well-known input widgets. This is a good thing of course, it reduces the memory burden on the user, and avoids issues like misspelling names. However, there is a great opportunity for a GUI to enhance the user's productivity by providing a more imaginative interface that expands beyond the simple one-to-one correspondence between GUI tasks and the underlying SQL commands. For the most part MXDM and VQP have both missed this opportunity.
- **User Interface:** I included User Interface in the "Pros" list above because MXDM and VQP have Graphical User Interfaces that are inherently easier to use than a command-line interface. However, I have also included User Interface here in the "Cons" list because (in my opinion) its UI has a few shortcomings.
  - Features are not very "discoverable" for new users, for example:
    - Commands that are only available via right-click pop up menus.
    - The toolbar commands are represented by hard-to-understand icons, with no option to show toolbar button captions.
  - Lack of expert-level features to help make experienced users more productive. For example, keyboard shortcuts, or feature search.
  - One size fits all interface. The program makes use of tabs and windows with very little user control over how and when.
  - The interface lacks much in the way of user customization, providing a rigid program navigation system that can make it difficult to move between related program features.
  - Incomplete persistence of interface customizations can lead to confusion.

SQLXPress

These comments are based on the 3.41 release.

Pros:

- **Breadth:** SQLXPress supports virtually all of the features supported by MXDM and VQP, plus many more. See the In-Depth Comparison section on “Additional Feature in SQLXPress” for more details. Here is a non-exhaustive list as a teaser:
  - Complete audit of SQLXPress user activities.
  - Complete support for MX DDL statements.
  - Complete support for MP DDL statements.
  - Full support for partition management (MX and MP).
  - Partition Key data analysis.
  - Compare DDL.
  - Compare data.
  - Statistics Management (Display, update, copy, backup, restore statistics).
  - Visual Query Builder.
  - “No SQL required” data browser.
  - Data Import / Export.
- **Depth:** A feature has “depth” when it provides something extra to make the user more productive. It's about providing more than just a “fill in the form” alternative to typing in a command. An SQLXPress design imperative is to implement each feature in a way that goes beyond the simple replacement of commands with forms. Here are two examples that illustrate the idea:
  - The support for SQL name patterns is pervasive in SQLXPress. It's as easy to apply a command to one hundred (or even one thousand) objects as it is to a single object. The SQLXPress GUI applies the command to all of the objects that have names matching the supplied pattern.
  - SQLXPress supports the SHOWDDL command. Of course, you can select multiple objects using wild-card name patterns. The wizard will generate the CREATE and ALTER commands as required, and you can save the script to a file. (A PC file, a Guardian file, or an OSS file). The use case for this is so that you can easily re-create the SQL objects from the script, for example for disaster recovery. However, there are other use cases which are more likely. For instance, how about using SHOWDDL on a set of development tables so that you can use the script to generate a copy of the database for use by QA? To do this you are probably going to have to change the generated script in various ways. For example you might wish to:
    - Create the tables in a new schema.
    - Store the tables on a different set of physical disks.
    - Scale the extent size so the tables can hold more data.

Rather than have the user edit the script to make these changes, the SQLXPress SHOWDDL wizard will do it for you. You can specify schema name mappings, disk names for table locations, and extent scaling. The generated script will be ready to use to create the database clone. As before, you can save the script to a file, or you can go right ahead in the wizard and execute it directly. If a statement fails for some reason, the wizard will prompt you to either fix the statement by editing it, and retrying, or to skip it for now.

- **User Interface:** SQLXPress has a high-quality user interface designed to help both new and experienced users to be as productive as possible. Here are some highlights:
  - Fat client for good performance, but with web-browser-like look and feel for ease of use.
  - Features are easy to discover to help new users get up to speed quickly.
  - Expert-level features help experienced users get things done faster.
  - Flexible multi-tab, multi-window interface, which puts you in complete control of program behaviour.
  - Flexible program navigation to suit different user work styles.
  - A highly customizable user interface. You can configure:
    - Colours
    - Fonts
    - Toolbar Styles
    - Grid configuration including column visibility, column order, sorting, grouping, and filtering.
    - Styles, and page layout for printing subsystem.
    - Character sets.
    - On-screen page layouts.
  - All customizations persist between sessions so you can “set it and forget it”.
  - The help subsystem is embedded right into the user interface with context sensitive help synchronized to each page in the application.

Cons:

- **Cost:** There is an additional cost for SQLXPress. The cost depends on system configuration with small systems with a few CPUs costing less than larger systems with more CPUs. There are a number of pricing options available designed to suit specific customer's requirements.

Generic Third-Party Tools

There are numerous third-party SQL management tools available on the market. Its possible to use some of them with SQL/MX.

Pros:

- **Cost:** The mass-market tools have a modest cost, typically a few hundred dollars per user.
- **Multi-Database Support:** Many tools support all of the major merchant databases like SQL Server, Oracle, and DB2.

Cons:

- **Breadth:** Generic third-party tools support SQL/MX as a "generic database" via an ODBC or JDBC interface. The problem with this is that the most useful features are not supported via the ODBC/JDBC interface, and therefore can't be used with NonStop SQL.

  Features that **are** typically available include:

  - Browse SQL metadata.
  - Execute Query (aka SQL Whiteboard).

  Features that **are not** typically available: Pretty much everything else, including:

  - DDL operations – no support for NonStop SQL specific extensions like file attributes and partitioning.
  - Display of physical database information like files attributes, file opens, locks, TMF transaction information, disk space utilization, etc.
  - Anything to do with execution plans.
  - Utilities such as DUP, Reload, Modify Table, Populate Index etc.

Generic third-party tools are not a viable option for managing NonStop SQL.

In Depth Comparison of MXDM And SQLXPress

This is where we get a bit more technical.

The comparison is broken down as follows:

- **Breadth.** This compares the product feature sets,
- **Depth.** This compares the richness of implementation of various features.
- **User Interface.** This compares the design of the user interface.

Breadth

Features In Common

MXDM and SQLXPress both support the following features:

- **SQL/MX Metadata Display.** This allows you to display information about the SQL objects in their databases.
- **SQL/MX DDL Commands.** This feature gives you the ability to perform Data Definition Language commands. For example, CREATE, ALTER, DROP, RENAME, SHOWDDL.
  - **Note:** The current (3.3) release of MXDM only supports a subset of DDL commands. It **does not** support:
    - Alter, Create, Drop, Rename View
    - Alter, Create, Drop, Rename Sequence
    - Alter, Create, Drop MP Alias
    - Alter, Create, Drop Trigger
    - Create, Drop Procedure
    - Update Statistics
    - Give, Grant, Revoke.
  - SQLXPress supports all DDL commands.
- **SQL Whiteboard.** (SQLXPress calls this feature “Execute SQL Statements”). This feature allows you to execute dynamic SQL statements. You type in the command in a multi-line editor, and click an “Execute” button. The statement is then executed and the results are displayed.
- **MXCS Management Commands.** This allows you to configure the MXCS subsystem, and to display the status of the various MXCS components.
- **Launching The Remote MXCI Program.** This allows you to start the Remote MXCI program. This is a command line tool (that runs on your PC) which can be used to enter dynamic SQL statements.

These features are supported in both products, but that doesn't mean that they are equally effective. See the **Depth** section for details on how the SQLXPress implementation compares to MXDM.

Features in MXDM But Not In SQLXPress

- **View Sampled Statistics.** This feature samples the data corresponding to an existing statistics histogram. The histogram data and the sampled data are displayed. This can be used to determine whether the statistics are out of date.
- **SHOWDDL Permissions Option.** The output from the SHOWDDL command includes GRANT statements. SQLXPress does not include GRANT statements.
- **Copy Datasource.** The command creates a new MXCS datasource “like” an existing datasource.

Features in SQLXPress But Not in MXDM

- **Audit.** SQLXPress can be configured to audit user activities. There are a number of different audit levels that can be configured, from “None” to “Everything”. The audit subsystem supports **separation of duties** such that the security officer can configure and manage audit, but cannot use any other SQLXPress features. The audit subsystem has its own GUI client, and includes comprehensive reporting facilities.
- **SQL/MP support.** SQLXPress provides full support for NonStop SQL/MP. Features that support MX, also support MP, with some obvious exceptions. For example, MP doesn't support triggers, so the SQLXPress trigger-related features do not apply to MP.
- **Background Task Support.** Sometimes you don't want to perform a task interactively. Maybe it will take a long time to complete, and you don't want to wait. Or, you might want to do it at an off-peak time so that it doesn't interfere with normal application processing. Its easy to come up with examples; update statistics, populate index, move partition, dup tables, etc. SQLXPress includes extensive support for initiating and monitoring background tasks:
  - The SQLXPress GUI schedules potentially long-running operations to run in the background.
  - You can submit custom tasks written in TACL the Merlon Scripting Language, or any other command line interface.
  - Background Task Management integrates with NetBatch (or the Merlon Batch Monitor).
  - Separate Task Manager GUI to manage and monitor background tasks.
  - Schedule tasks to run immediately, or in the future.
  - Schedule tasks to run repeatedly at a specified interval (for example every Sunday at midnight).
- **Merlon Scripting Language (MSL).** The MSL scripting language is ideal for developing scripts to implement routine database maintenance tasks. MSL scripts are processed by the MSCRIPT program which can be run interactively from TACL, or in the background via the Task Manager. Here are some of the main features of MSL:
  - The MSL syntax is based on the ANSI SQL 92 standard for Persistent Stored Modules (SQL/PSM) . It is similar to stored procedure languages used by other database systems, such as MySQL, and DB2. It includes support for SQL statements, cursors, host variables, assignment statements, control statements like If-Then-Else, Case, and While, etc.
  - MSL is designed to shorten the traditional edit-compile-link-run process. To run a script, you simple invoke the MSCRIPT program, passing the name of the script as a parameter.
  - MSL scripts run in the Guardian environment.

- The MSCRIPT program has a built-in symbolic debugger which you can use to debug scripts.
- Both NonStop SQL/MP and SQL/MX are supported, even from within the same script.
- You can execute SQLCI and MXCI utilities and commands from within a script.
- MSL has a number of extensions designed to enhance its usefulness as a scripting (as opposed to stored procedure) language. For example, terminal IO, disk file IO, run program, and so on.

- **Environments.** An Environment is a named collection of SQL objects – i.e. catalogs, schemas, tables, views, sequences etc.

  Objects must be registered to be included in an environment. When you register an object, you assign a synonym to it. The synonym is used to refer to the object instead of the object's SQL name. (Synonym names look like DEFINE names, you use them for both MX and MP objects). The idea here is that you can use the same synonym in multiple environments to refer to a different instance of the "same" table. For example you can set up environments for Development, QA, Staging, and Production. In each environment you register a different instance of the ORDERS table and assign the synonym =ORDERS to each. When you switch between environments, you use =ORDERS to refer to the orders table rather than the SQL object name.

  When you open an environment, only the objects registered in that environment are shown in the User Interface.

- **Visual Query Builder (VQB).** The VQB offers an alternative to the "SQL Whiteboard" for building SQL queries. It allows you to create queries "graphically" rather than by entering the query directly as text. (Actually, as it turns out, it can do both. It has a Builder tab and an SQL tab. The Builder tab lets you create and edit queries graphically, while the SQL tab lets you edit the query text directly. This is a "two-way" facility. Changes made in one tab are reflected in the other).

  However, what makes VQB really interesting is that it saves queries in a NonStop SQL database on the NonStop server. This allows queries to be easily shared with other users. In particular, users of the **XPressView** program can run queries developed using the VQB.

  VQB supports run time parameters. When you run a query, you are prompted to supply a value for each parameter.

- **XPressView.** XPressView is a GUI program that comes as part of SQLXPress. It has its own installer, and is intended for use by non-technical users. XPressView provides read-only access to NonStop SQL databases. You can view data from tables, views, and queries built using the VQB. You do not need any knowledge of SQL to use XPressView.

- **"No-SQL Required" Data Browser / Editor.** This is another alternative to the SQL Whiteboard. It allows you to display data from SQL tables, views, and queries built using the VQB. In addition, you can insert, update, and delete rows from tables (and some views). Some features:
  - Bi-directional cursor emulation for table browsing. You can go to the first, last, next, or previous row. Also, you can jump to any row in the table by entering some or all of the key values and clicking Find; the nearest rows are displayed, and from here you can browse backwards and forwards.
  - Rows from the table are cached locally in PC memory. A "smartcache" keeps track of the most recently viewed rows. Older rows are automatically discarded as required to make room for newer rows.
  - Filter rows.
  - Order by primary key, cluster key, or an index.
  - Select which columns to display.
  - Select the order in which columns are displayed.

- Display / edit character data in hex.
- Display / edit character data using a substitute character set (for example. Windows code page 1252 instead of ISO88591).
- Formatted data output (for example display LARGEINT values as timestamps converted to LCT).
- All of the various customizations such as filters, order by, column selection, formatting and so forth are persisted. The next time you view the table, the customizations are automatically applied.
- All of this is supported without you having to enter any SQL statements.

- **Execution Plan Management.** SQLXPress maintains a database of query execution plans. (Of course, both MX and MP plans are supported). Plans can be extracted into the database from a number of sources:
  - Static embedded queries in C / C++ / Cobol programs.
  - Queries from an MX Module File Cache.
  - Queries from the MXCS EMS statistics log.
  - Queries built using the Visual Query Builder.
  - Queries saved via the Visual Query Tuner.

  For MX programs, the database also keeps track of the modules used by a program, and the SQL tables used by each module. This information is used by the Browse Object Relationships feature.

  The exec plans database can store multiple "snapshots" for a query. This allows you to detect changes in the plan over time.

  Once execution plans have been captured in the database you can do a number of interesting things with them.
  - Display execution plans in various ways, including graphical plan diagrams.
  - Compare plans between query snapshots.
  - Query the exec plan database for "problem" plans.
- **Browse Object Relationships.** This facility is used to display the dependencies between SQL objects, programs, and modules. Direct and indirect relationships are displayed, as well as "Using" and "Used By" relationships. (MXDM does have some support for object relationships, but its fairly limited in comparison to SQLXPress).
- **Partition Management.** Both SQL/MX and SQL/MP have extensive support for managing table and index partitions. The partition management operations can be performed in "off-line" or "on-line" modes. SQLXPress supports of of the available partition management commands, for example add, drop, move, merge, and split. Partition management operations aer launched via wizards in the SQLXPress client, and are performed as background tasks.

  Some partition management operations, such as split or one-way move, require you to specify the first key value of the "split point". For example, if you want to split a partition into two equal parts, you need to specify a first key value such that half of the rows have keys less than the value, and half have keys larger than the value. This is quite difficult to do. SQLXPress includes a **partition key analysis** feature that solves this problem. It analyzes the distribution of partition key values within a partition and lets you select a percentage of rows before and after the split point, 50-50 say,or 67-33. It then fills in the first key values based on the selected ratio.
- **Statistics Management.** Of course, SQLXPress supports the Update Statistics command. In addition, it provides facilities to backup statistics before doing an update statistics. You can undo the effect of an update statistics by performing a restore from from a previous backup. There is

also a wizard that will copy statistics from one set of tables to another set. This can be useful for tuning queries based on production rather than development statistics.

- **Security Management.** SQLXPress supports all security-related commands such as grant / revoke, give, etc. The bulk-mode grant/revoke wizard is particularly effective. It allows you to grant or revoke privileges to many objects with a single command. SQLXPress also supports managing MX security administrators.
- **Import / Export.** SQLXPress supports the import of data into NonStop SQL tables from CSV files, XML files, Enscribe files, and other databases (for example Oracle or MySQL). You can export data from NonStop SQL to CSV files, XML files, NonStop SQL databases, or other databases. The various import/export wizards support filtering, column selection, and data reformatting. To be clear, the import/export feature is aimed at small to medium sized tables, and is suitable for ad-hoc data movement. This is not a high-throughput data replication type feature.
- **Server Monitoring.** The server monitoring feature allows you to monitor key system performance metrics including CPU busy, memory usage, TMF activity, SQL processes, disk IO rates, and disk space utilization.
- **Database Reports.** A comprehensive set of reports are available for both MX and MP. Examples:
  - DDL locks
  - File Locks by Object
  - File Locks by Transaction (includes deadlock detection)
  - File Opens
  - Unreclaimed Data
  - Incomplete DDL Operations
  - Out of Schema References
- **Duplicate Tables.** Duplicates a set of tables and their dependent objects. Includes support for view-defining query re-write to refer to target tables.
- **Reload Tables and Indexes.** Perform an online reload of table or index partitions.
- **ShowDDL For Missing Tables.** Finds tables that are defined in one schema, but not in another. Generates DDL to create the missing tables.
- **Compare DDL.** Compares the logical structure, physical structure, and security settings for two sets of tables.
- **Compare Data.** Compares the data in two sets of tables. Reports on differences.
- **Create MXCS Service:** Creates a new instance of an MXCS Association Server.
- **Extract MXCS Statistics From EMS.** Extracts MXCS statistics events from an EMS log.
- **Display MXCS Statistics.** Displays extracted statistics.
- **Advanced MXCS Datasource Configuration.** Support for additional datasource attributes for XA Broker and Microsoft Access. Also offers improved support for EVARS for CQD and SET.
- **XPressEdit.** A fully-featured text editor with special features to support NonStop SQL. We don't expect this to replace a developer's favourite text editor, but its very handy when working with MSL scripts for example. Main features:
  - Support for Guardian EDIT files, OSS files, and PC files.
  - Multi-file editing.
  - Synchronized file editing.
  - Color-coded syntax highlighting for numerous languages, including:

        - NonStop SQL (both MX and MP), The Merlon Scripting Language (MSL), C / C++, Cobol, Java, Tal, and Pascal.
    - Auto completion for NonStop SQL, including table, column and function name lookup.
    - Auto completion for MSL.
    - Support for embedded SQL in MSL, C, and COBOL.
    - Text folding.
    - Macros.
    - Code Templates.
    - Configurable keyboard shortcuts.
    - Search and replace.
    - Bookmarks.
    - Markers.
- **Extents Calculator.** A utility that calculates the file extent sizes required for a file based on sizing parameters like maximum number of records, records size etc.
- **PSTATE.** Runs the PSTATE utility on a process – lists detailed information on the internal state of a process, including opened files
- **Manage Process.** Supports process management commands including Alter Priority, Suspend, Activate, Status, and Stop.
- **SCF INFO.** Displays information about disk volume configuration.
- **SCF STATUS.** Displays information about disk volume status, including DP2 cache, and MX buffer space consumption.
- **Find SQL Objects.** Searches for SQL objects based on an object name pattern.
- **Parallel Index Load file editor.** An easy-to-use editor for SQL/MP parallel index load (PIL) configuration files. Creates and alters PIL files.
- **Printing Subsystem.** Supports printing from almost anywhere in the application. This is much more than simply printing a screen image. For example, a grid may have hundreds of rows, only some of which are displayed on the screen. When printed, all of the grid rows are printed. Main features:
    - Print any “page” in the application.
    - Customizations that have been applied to the screen display are also applied to the printed data - for example grid row filtering, grouping, sorting, column selection etc.
    - User-defined print styles that can be applied to any report. Print style editor allows user to create new styles, and edit existing styles. Styles provide control over page orientation, margins, headers and footers, scaling, fit to page, etc.
    - Report Layout designer allows user to customize page layout for reports.
    - Output can be sent to local or LAN-attached printers.
    - Export reports to PDF file.

Depth

In general, I believe it is fair to say that for features implemented by both SQLXPress and MXDM, SQLXPress typically provides a richer implementation. I refer to this as the "depth" of the implementation.

I have selected a few examples to illustrate the point.

SQL Whiteboard

This is the term that MXDM uses. SQLXPress calls this the "Execute SQL Statements" page. This features allows you to enter an SQL query, execute it, and view the results. I'll refer to the feature as simply "the whiteboard" to avoid confusion.

Both implementations have a fairly similar look and feel. There are three main "panels":

- **Statement:** Provides a multi-line text editor which is used to enter SQL statements.
- **Results:** Displays the results from executing a statement.
- **Previous Statements:** Provides a means to reuse previously executed statements.

Note that the actual names used for these panels is a little different in each product, but I'll use these names to simplify the discussion.

Both products allow you to customize the layout of the three panels. This allows you to devote screen real estate to each panel to suit your needs. In the User Interface section I'll compare how SQLXPress and MXDM implement this feature.

The Statements Panel

Both products offer a way to specify a default schema, which is used to resolve partially qualified table names.

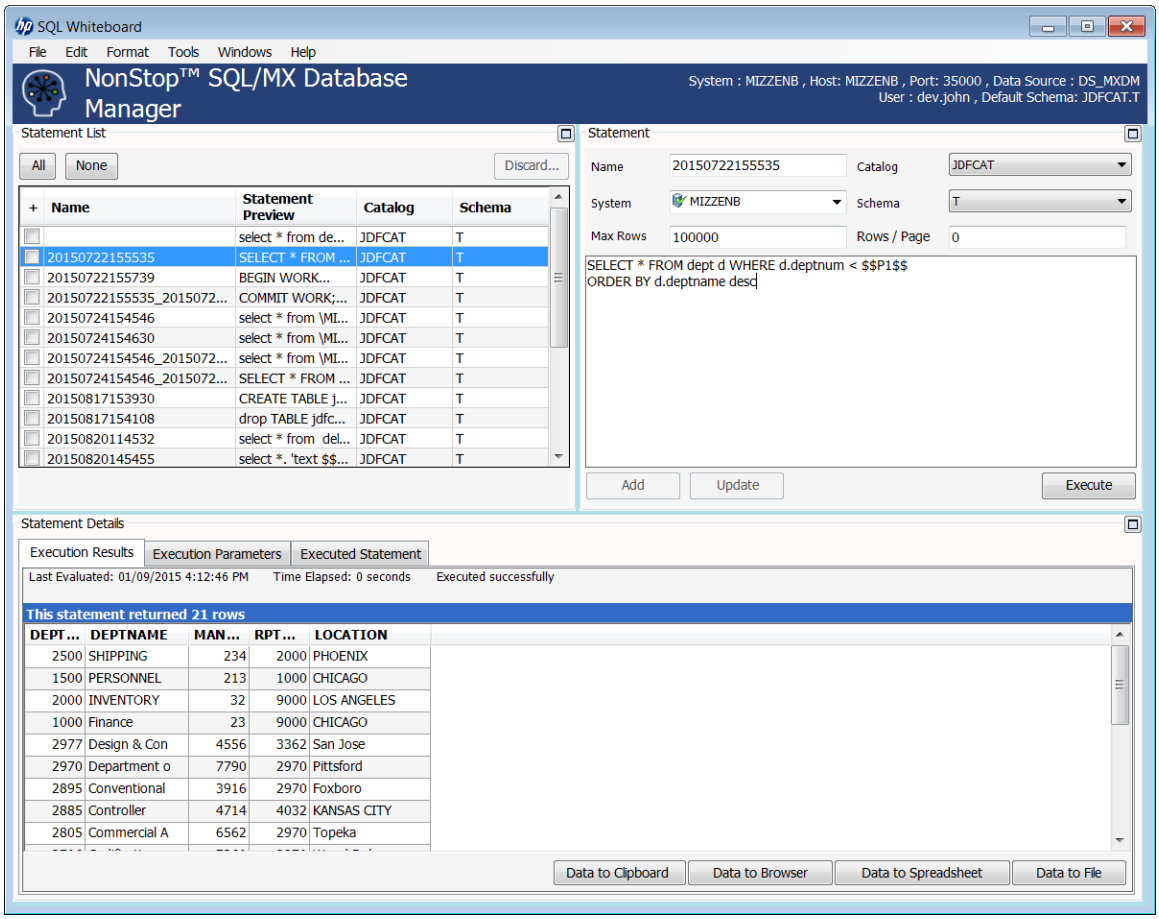
MDXM also provides fields that allow you to limit the number of rows returned by a query, and the number of rows cached in memory.

SQLXPress provides a drop-down list of the tables in the default schema which is used to select a table name for some commands.

Entering the query text.

MXDM: You enter an SQL statement, which can wrap over multiple lines. The standard Windows editing commands (cut, copy, paste etc) can be used. There is no other assistance available to the user. This is marginally better than entering statements via the command line (for example using MXCI).

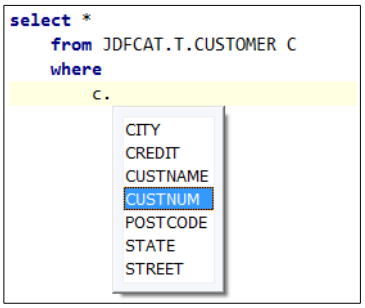
The whiteboard supports a curious form of syntax highlighting. Syntax highlighting is only applied when you perform the Highlight Syntax command from the Format menu. This applies color-coding to SQL keywords (some of them at least). If you subsequently edit the text, the highlighting is not updated as you type, so if you delete the S from the front of SELECT, the rest is still highlighted. Executing the statement seems to remove all highlighting. Since the point of highlighting is to allow the user to easily spot typing errors, this approach seems less than ideal (to me at least).



MXDM SQL Whiteboard

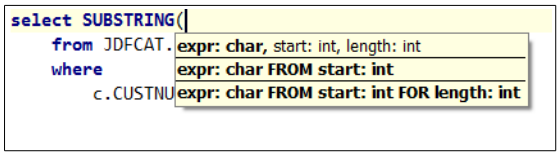
SQLXPress: The basic process of text entry is pretty much the same as with MXDM, i.e. a multi-line text editor that supports the common editing commands. However, there are a number of additional features that make the job much easier:

- Real-time syntax highlighting: SQL keywords, numbers, quoted strings, comments etc are color-coded as you type. This makes it easier to spot typing errors and correct them on the fly.
- Auto-completion: This feature helps by providing a list of choices from which you can select, instead of having to type everything. For example:



  Here the user has typed c, followed by a dot. The text editor recognizes that c is the alias for the CUSTOMER table, and pops up a list of its column names. You can use the keyboard, or the mouse, to select a column name, and the name will be pasted into the statement after the dot. In a similar way, a popup list can be used to select table names, or SQL function names. Note that you can "force" a popup list to appear by using the Ctrl + Space shortcut.

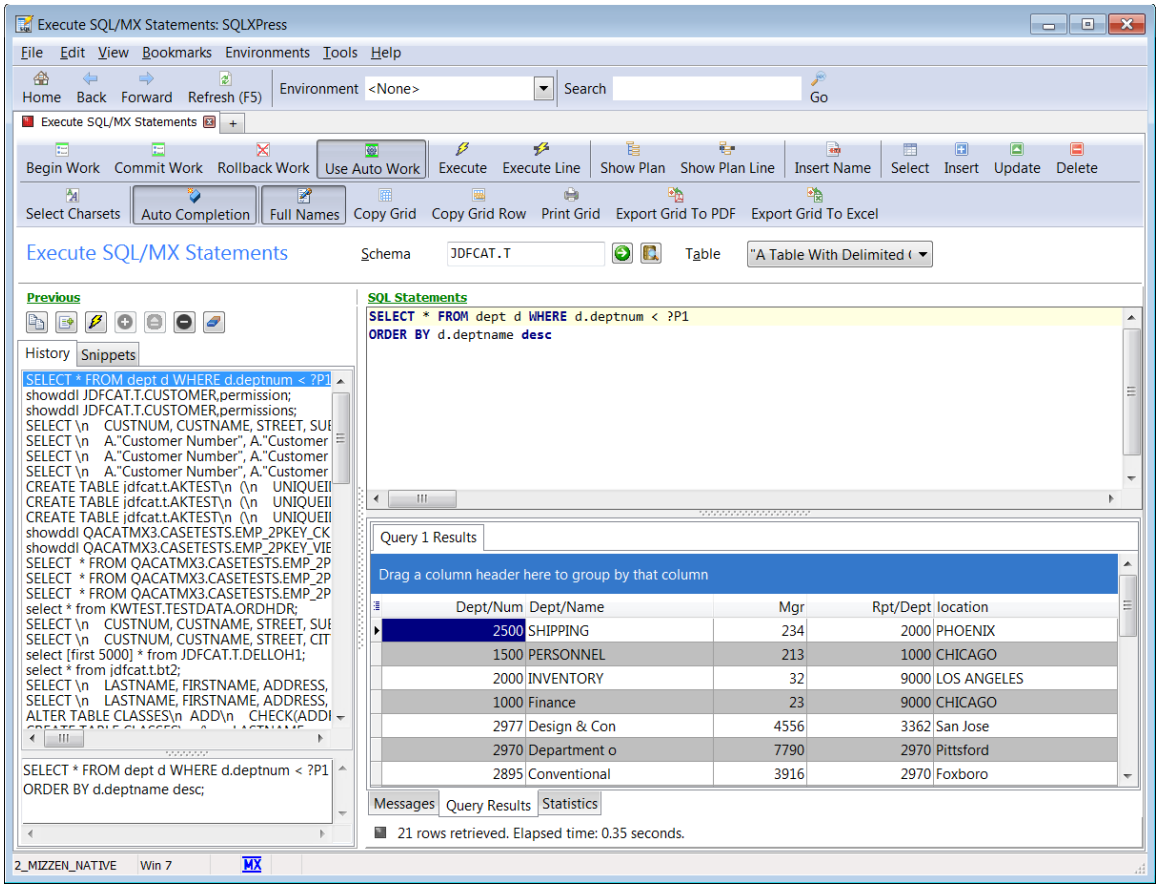
- Function Parameter Hints: This feature reminds the user of the parameters used by the various SQL built-in functions:



  Here, the user typed SUBSTRING (or more likely picked it from a popup list) followed by a an open parenthesis. This triggers the editor to show a hint that lists the parameters taken by the SUBSTRING function. In this case there are three different possibilities, all of which are shown).

- DML Statement generation: You can generate a SELECT, INSERT, UPDATE, or DELETE statement by selecting a table from a dropdown list, and clicking the appropriate toolbar button. A statement is generated and pasted into the text editor. The statement can be executed as-is, or can be edited as required.
- Transaction control statements: Both products implement auto-work, which relieves you from having to explicitly begin and commit transactions, each statement is automatically enclosed in a begin work-commit work pair. You can also choose to explicitly begin and commit transactions. This is useful when you want to execute multiple statements as part of the same transaction. In MXDM you have to type the BEGIN WORK and COMMIT WORK statements. In SQLXPress,

there are toolbar buttons (and context menu commands) to perform these statements, without having to type them out.



SQLXPress Execute SQL Statements

Executing a statement:

MXDM: You can execute one statement at a time. You click the Execute button to execute an SQL statement. If no text is selected in the Statement text editor all of the text is considered as part the the statement. If some text is selected, only the selected text is used.

If there are multiple statements entered into the whiteboard, you must select the exact text for a single statement before it can be executed.

SQLXPress: You can execute one or more statements. You have a number of options:

- Click the Execute button (or press F6) to execute one or more statements. Multiple statements must be separated with semi-colons (;). If no text is selected, all of the text is used, otherwise only the selected text is used.
- After the statement text, type a semi-colon followed by Enter. The statement is executed; In fact, only the statement immediately preceding the ; is executed, other statements in the editor are ignored. At any time you can place the cursor after a ; and press Enter to execute a statement. If you want to enter an additional statement without executing the previous statement, simply terminate the first statement with a ; then type a space. You can then press Enter to start the next statement on a new line.

- Click the Execute Line button (or press F7) to execute the text on the current line in the editor.

Cancelling a query:

MXDM: You can stop a query by clicking Cancel. For a SELECT, any rows that have already been fetched are discarded.

SQLXPress: You can stop a query by clicking Stop. For a SELECT, any rows that have already been fetched are displayed in the Results panel.

A word about parameters:

MXDM: To include a "parameter" in a DML statement, you must type $$ followed by a parameter name, followed by $$. This is used in place of the standard SQL syntax for parameters (a ? optionally followed by a name). If you paste a query with SQL parameters into the text editor, you will have to rewrite the query to use this double-dollar notation. Likewise, if you create a query in the whiteboard and want to save it in a script, or paste it into a program, you will have to rewrite it to use the correct SQL syntax.

When you execute the query you are prompted to supply a value for each parameter. The query text is then altered to replace the $$ enclosed text with the values as entered. In fact, SQL parameters are not actually used, text substitution is performed before the query is sent to the server for execution.

SQLXPress: The standard SQL syntax is used for parameters. Both named and unnamed parameters are supported. In fact if you type a host variable name, the whiteboard will translate the query text to use a parameter name instead, this makes it easier to deal with static SQL statements that have been cut and pasted from a program.

You are prompted to supply a value for each parameter when the query is executed.

Query Execution Plans.

For DML statements, SQLXPress supports displaying the execution plan from the whiteboard page. Simply click the Show Plan (or Show Plan Line) button.

MXDM does not support execution plan display.

The Results Panel

The results panel displays the results of executing a statement.

MXDM: The Results Panel contains one or more tabs, depending on the statement that was executed, and the outcome of the execution. Possible tabs include:

- Execution Results: The outcome of executing a statement. For a SELECT, there is a grid showing the rows returned by the query. For other types of statements this is a text panel.
- Executed Statement: The text of the statement that was executed.
- Execution Parameters. For a query with parameters, the name and values used when the query was executed.

SQLXPress: The Results Panel contains three tabs:

- Execution Results. For SELECT and SELECT-like statements (for example SHOWDDL) the rows returned from the query are displayed in a grid. If more than one SELECT was executed, there is a grid for each one.
- Messages. Feedback on the outcome of each statement.
- Statistics. A grid that displays the execution statistics for the statement. These are per-table statistics that include records read, records used, number of messages etc. If more than one statement was executed, there is a grid for each one.

The Previous Statements Panel

Both products provide a "previous statements" facility. The idea is that you get a record of previously executed statements. You can copy a previous statement into the Statements panel, then execute it. This saves you from having to re-type the statement.

There are implementation differences between the products:

MXDM: There is a list of previously executed statements. For each statement the list contains the statement name, text, and default schema.

When you execute a statement it is automatically added to the end of the list.

When you click on an entry in the list, its text is copied to the statements panel. You can then click Execute to rerun the statement. In this case the query is not added to the list, since it is already on the list. Curiously, if you type in a statement that is already on the previous statements list, then execute it, the statement is added of the end of the list, resulting in a duplicate entry.

You can explicitly add a statement to the list without executing it first. You can also edit an existing statement, and update the list to reflect the changes.

SQLXPress: There are actually two lists in the SQLXPress Previous panel, the History list and the Snippets list.

When you execute a statement, a copy is inserted at the top of the History list. If the statement already appears in the list then a new entry is not created, instead its existing entry is moved to the top of the list. So, the history list contains the previously executed statements with the most recently executed statements at the top of the list, with no duplicates. When the list grows to 100 entries, adding a new entry causes the least recently used entry to be removed from the list. You can also explicitly delete entries from the History list.

Unlike MXDM, when you click on an entry in the list, the statement is not copied to the statements panel. Instead, its text is displayed in a text panel which appears below the Previous panel. This allows you to see the complete text of the statement. The reason that the text is not immediately copied to the statements panel is that the panel may already contain other statements that you don't want to overwrite. You have the option to overwrite the statements panel contents with the selected statement in the list, or to insert the statement from the list.

The Snippets list is used to store pieces (snippets) of SQL statement text that you might want to re-use. (In fact you can use this list to store complete SQL statements if you wish). The difference between the Snippets list and the History list is that management of the Snippets list is completely manual. You can add, delete, or update snippets in the list, and older snippets are not automatically deleted.

ShowDDL

In this section we will compare the depth of implementation for the SHOWDDL function.

Use Cases

Before proceeding it is worthwhile to consider the question “What are the use cases for SHOWDDL?”

The SHOWDDL command reverse engineers the SQL metadata definition of an object to produce the equivalent SQL CREATE statement.

Armed with the generated CREATE statement, you could re-create the “same” object. So, one use case is to re-create an SQL object that has been lost, for example as a result of a system disaster, or because it was inadvertently dropped.

Another use case is to use SHOWDDL to generate a CREATE statement that can be used to create a new instance of an SQL object, with the same name, but on a different system (Expand node). In this scenario, we need to eliminate references to the original system name in the generated DDL, either by omitting the system name, and relying on the default name on the new system, or by replacing the system name explicitly. This applies to the location of the SQL object, and its partitions (if any). You can imagine for example doing a SHOWDDL on the production system and using the generated script to create a test database on a development system. There are similar issues with disk volume names if the source and target systems have different disk configurations.

A third use case is to use SHOWDDL to generate a CREATE statement that can be used to create an instance of an SQL object, with a different name, perhaps on the same system. For example, using SHOWDDL in the development environment to create an equivalent database in the testing environment.

MXDM does not address the second or third use cases. You would have to manually edit the generated DDL script to handle these use cases.

SQLXPress does address all of these use cases.

Here is a description of each implementation.

MXDM

When you select an SQL object in the main window a set of tabs is displayed. One of these tabs is the DDL tab, which shows the DDL for the selected object. The limitation here is that you can only select one object at a time. If you wish to generate the DDL for multiple objects, you must use the ShowDDL command, which is available by right-clicking on a node in the databases tree.

The ShowDDL command opens a dedicated window. On the left is a tree representing the SQL object name space. On the right is a text panel. The database tree has a check box for each node. To select a node, you check the check box. That node, (and its children, if any) are selected. You can uncheck individual child nodes to exclude them. So, for example, you could check the Tables node for a schema to select all of the tables in the schema.

Aside: It turns out that the ShowDDL command is unique within MXDM in its ability to operate on multiple objects. Other commands operate on one object at a time.

To show the DDL, you select the object(s) in the tree, then click Load. The text panel is cleared and the DDL for the selected nodes is displayed. There is also an Append command which does not clear the text panel, but appends the DDL to the existing text.

You can only select objects within a single schema. If you try to select objects from multiple schemas, an error is displayed. To work around this you can select objects from one schema at a time, and use the Append command.

The DDL is generated by sending a SHOWDDL command to the NonStop server, the PRIVILEGES option is used so that the generated DDL includes GRANT statements.

Once the DDL has been displayed in the text panel, you can save it to a file on the PC.

SQLXPress

In a similar way to MXDM, when you select an individual SQL object, you can view its DDL from the DDL tab in its object properties page. If you need to generate the DDL for multiple objects you must use the ShowDDL wizard. The following description is for the MX version of ShowDDL, SQLXPress also implements ShowDDL for SQL/MP – even though MP itself doesn't have a ShowDDL command.

The ShowDDL wizard leads you through the steps required to generate the required DDL.

Step 1. **Selection.** In this step, you select SQL objects. There is a field which is used to enter an object name, and a list of the already selected objects. You type in the name of an object, then click Add to add the object to the list. Obviously, it would be very tedious if you had to type the name of each individual object. There are a couple of mitigations for this. You can enter a object name pattern containing wildcard characters. When you click Add, all of the names that match the pattern are added to the list. Also, there is a Browse button, when clicked an object selection dialog is opened. This contains a tree representing the SQL object name space, you can select an object name using the tree. Again, in the Browse dialog, you can edit the name of a selected object to include wildcard characters.

You can select objects from multiple schemas if required.

> Aside: This technique of selecting multiple objects is commonly used for many commands in SQLXPress.

Step 2. **DDL Options.** In this step, you are given a number of options that control how the DDL is generated. In particular you can control how LOCATION attributes are generated. You can chose to omit location altogether, or if it is included to omit either the system name, and/or the file name parts of the location. This allows the generated DDL to be used on a different system.

Step 3. **Advanced Options.** These options also control how the DDL is generated.

Substitute Schema Names: This option allows you to “map” object names from one schema to another schema. The idea here is to allow you to “clone” SQL objects in a source schema into a target schema. A mapping is defined for each source schema into a corresponding target schema. If the target schema does not exist, the generated DDL will include a CREATE SCHEMA statement for the target schema. To be clear, the selected objects can be located in more than one schema, so there would be a mapping for each source schema. This mapping not only applies to the names of the selected objects, but also to object names found in the definitions of views and triggers.

Scale Extent sizes: This option allows you to control how primary and secondary extent sizes are generated. By default, the extent sizes are those defined in the metadata. However, you can chose to scale the sizes, up or down, by a scale factor. For example, make the extents ten times larger, or five times smaller.

Specify Location Volume Names: This option allows you to specify alternative disk volume names for table, index, and partition locations. This is useful when the generated DDL is to be used on a different system, or a different set of disk volumes within the same system.

Step 4. **Substitute Schema Names:** If you chose to use substitute schema names, you use this step to add the schema name mappings.

Step 5. **Scale extent Sizes:** If you chose to scale extent sizes, you use this step to specify the scale factor.

Step 6. **Partition Location Volumes:** If you chose to use substitute location volumes, you uses this step to pick the volume names. Names from the list are used round-robin and assigned to the location attribute for tables, indexes, and partitions.

Step 7. **Generate Script:** This step generates the script. The script can be formatted as an MXCI script, or an MSL script. After the script has been generated, you can save it to a PC file, a Guardian file, or an OSS file. If the script was generated using substitute schema names, you can execute the script directly from within the wizard.

Working With Multiple SQL Objects

There are cases when the database administrator is required to perform the same action on multiple SQL objects, for example on all the tables in a schema. Here are some examples:

- GRANT privileges to tables and views.
- UPDATE STATISTICS on tables.
- Set the CLEARONPURGE attribute for tables.
- GIVE tables to another user.

In MXDM (with the sole exception of ShowDDL) commands apply to a single object. You must go through the steps required to apply the command once per object. This can become quite time consuming.

In SQLXPress, you select the objects, then go through the steps to perform the command once. SQLXPress then applies the command to each of the selected objects in turn.

Object references in SQLXPress can include wildcard characters, making it easy to select many objects at once.

User Interface

In this section we will compare the user interface of each product.

Summary of conclusions:

- SQLXPress has a more flexible user interface.
- The SQLXPress user interface is easier to learn for beginners.
- The SQLXPress user interface is more customizable, and is better at remembering user preferences. This makes intermediate level user more productive.
- The SQLXPress user interface offers more advanced features for expert users.

User Interface Overview

MXDM

The MXDM main window is used to navigate to the various features in MXDM. There is exactly one instance of the main window. Other task-specific windows are opened in response to user actions as required. For example, if you execute the Create Table command, a window containing the Create Table wizard is opened.

On the left there is "sidebar", it has two "areas", one for Database, and one for Connectivity. The Database area is used to work with SQL objects, the Connectivity area is used to work with the MXCS subsystem.

Database Area

On the left there is a treeview that allows you to navigate the SQL object name space. At the top level, there is a Catalogs folder which has the set of MX catalogs as children. Each catalog has a child Schemas folder, each schema has a set of folders for Tables, Views etc. The tree reflects the hierarchy of MX objects. When you select a node in the Databases tree, information related to the node is displayed in a set of one or more tabs on the right hand side of the window. A splitter control can be used to resize the panels containing the sidebar on the left, and the information tabs on the right.

There is a panel above the Databases tree that contains "Favourites". These are references to nodes in the Databases tree. You can drag a node from Databases to the Favourites panel to add a Favourite. Subsequently, you can click on a favourite to navigate directly to that node in the tree.

Connectivity Area

Once again there is a treeview on the left, this time representing MXCS Services and Datasources.

You select a node in the tree and the corresponding information is displayed in the information panel on the right hand side.

The Connectivity area also has a favourites panel which acts in the same way as the Databases favourites.

This is a fairly conventional interface for SQL management tools.

In general, the information shown in a tab in the "information" panel can be "cloned in a window". A supplementary window is opened showing the contents of the tab. The supplementary window operates somewhat like a main window, except it does not have a sidebar, so you cannot use it issue DDL commands or navigate to other objects.

There is a Window Manager command which allows you to switch between MXDM windows. You can also use the Windows taskbar, or the Alt+Tab shortcut to switch between MXDM windows.

SQLXPress

SQLXPress works much like an Internet Browser. The main window displays “pages”. You navigate from page to page using hyperlinks. You can return to a previous page by clicking a “back” link.

Pages fall into three main categories:

- A Topic page contains links to other pages. The SQLXPress Welcome page is an example.
- An Information page displays information, for example the Tables in a Schema page shows summary information about all of the tables in a schema.
- A Task page allows you to perform some type of action, for example create a table, or execute a query.

The SQLXPress main window is displayed after you log on. Initially, it displays the SQLXPress “home” page. By default, the SQLXPress Welcome page is the home page, but you can configure any page to be the home page.

A typical page has three panels. On the left is the Navigation panel, on the top is the History panel, on the right is the page content panel.

- The navigation panel contains hyperlinks to other pages.
- The History panel contains links to previously visited pages. (These links are sometimes called “breadcrumbs”).
- The page content panel varies depending on the function of the page.

When you start SQLXPress, a main window is displayed with a single tab which displays the home page. It is possible to navigate from here to any area of the program, and to perform any of the supported tasks.

You will notice that there is no databases tree. This is a bit unusual for an SQL management tool. It turns out that SQLXPress offers two navigation approaches, by object, or by task.

Navigation by Object

On the Welcome page, there is a link to Browse Catalogs. Clicking on this link takes you to the Catalogs page. This has a grid that summarizes the catalogs that are visible from this system.

On the Catalogs page, the navigation panel has a link to Schemas in This Catalog. You select a catalog from the grid and click the Schemas in This Catalog link to go to the Schemas page. On the Schemas page there are links to Tables in This Schema, View in This Schema, and so on. In this way you can browse the SQL object name space. In the various pages that you encounter, there are links to tasks that are related to the objects displayed on the page. For example in the Tables in a Schema page there are links to Create Table, Alter Table, Grant / Revoke Privileges, and so on.

Using this approach you browse the SQL object name space, and can initiate commands related to the objects you encounter.

Navigation by Task.

On the Welcome page, there is a link to Manage The Database. Clicking on this link takes you to the Database Administration topic page. Here there are links to pages such as Create Object, Alter or Manage Objects, Manage Security and so on.

Following the links, you will end up at a page that performs some sort of task, like create table, split partition, or whatever. At this point you must select the object you wish to work on. You can enter the name directly, or click a Browse button to select an object using a Browse Object dialog. Browse Object displays a databases tree from which you can select an object.

Using this approach you browse through the various tasks that can be performed, then select the object.

Working on More Than One Thing at a Time

At any time, you can chose to create a new tab in the main window (just click the "Plus" button at the right of the tabs list). Each tab operates independently – like a Browser). In addition, you can create additional main windows (select New Window from the File menu). Each window operates independently, and each can have multiple tabs. Note that some commands automatically open a page in a new tab.

Bookmarks

You can "bookmark" pages. When a page is bookmarked, an entry is added to the Bookmarks menu. You can jump directly to a page by selecting it from the Bookmarks menu.

Comparison

MXDM supports a single main window from which you can navigate to the various SQL and MXCS objects. Supplementary windows are opened in support of some function, for example create table, or SQL whiteboard. You can clone some information tabs into a supplementary window.

SQLXPress supports multiple main windows. You can explicitly open additional tabs in each window. Windows and tabs operate independently of each other.

SQLXPress supports navigation by object, or by task.

The MXDM favourites feature and the SQLXPress Bookmarks feature are similar. The main difference is subtle. In MXDM, a favourite is a reference to a tree node, whereas in SQLXPress a bookmark reference is to a page.

Here is an example to illustrate:

- Suppose that in MXDM, you add a table as a favourite. When you click on the favourite, the databases tree selects the table. From there you can perform commands like create index, rename, and alter on that table.
- In SQLXPress you bookmark pages. So, you can bookmark a page to alter a table, another to Grant / Revoke on the same table, another to create an index on the same table. The table in the bookmark is the table that was selected at the time you added the bookmark. When you subsequently jump to the bookmark, the default table on the page is set from the bookmark, but you can change this if desired.

The User Experience

Beginners aren't stupid, but they are busy. They want to become proficient quickly. They need some hand-holding to get going. Once they become more familiar with the product they become intermediate-level users.

Intermediate-level users understand the scope and purpose of the product. They may not remember exactly how to use a particular feature, so they need hints to get them on track. An intermediate user knows how to use reference material, the online help is for them. An intermediate user will have a working set of features that he uses most often. He needs to get to them without the product getting in the way.

Some users will use the product so much that they eventually become experts.

Expert users tend to use more of the product features, and value faster access to them. They want shortcuts to everything.

A good user interface caters to all three levels of user.

MXDM

Beginners are able to easily navigate the Database and MXCS object spaces, using their respective treeviews. Selecting a node in the treeview displays the corresponding information.

Learning to perform other tasks is less obvious. Only a few commands are available using the Main menu. Some commands are available via buttons on the information panel for an object, but many are only available via context (right-click) menus, making them much less “discoverable” for beginners.

The toolbar on the main window mainly replicates main menu commands, with a few extras for navigating in the treeview. The big problem with the toolbar is that it only shows small buttons with icons. The icons are hard to interpret and remember for beginners.

The Favourites feature is useful for Intermediate users. This allows you to jump to a node in the tree with a single click.

There are a number of ways to customize the user interface:

- There are many places where information is presented in a grid. You can hide columns, and can change column order in the grid. Data can be sorted on more or more columns.

  Unfortunately, these customizations are not persistent. When you navigate away, or close the client, the customizations are lost and must be re-applied the next time you visit that area of the product. This severely limits their usefulness.

- The SQL Whiteboard and Sampled Statistics windows contain several panels. The size and position of the panels can be arranged by the user, and the layout persists.

  However, there are some drawbacks in the implementation.

  First of all, the size of the window itself does not persist. When the window is next opened, the panels may not be fully visible within the window. You may need to re-size the window to allow full access to the panels. (I managed to “lose” the SQL Whiteboard Execute button in this way. I opened the SQL Whiteboard window and couldn't execute any queries until I figured out that the Execute button was off the edge of the window – there are no scrollbars to give you a hint.)

  When moving or resizing a panel, the other panels do not automatically adjust. You must arrange each panel's size and position individually, a fairly tedious process.

You are allowed to move a panel so that it obscures the other panels in the window, which is not helpful.

When you re-size the window itself , the panels do not automatically adjust, except when the default layout is used.

Compare this with the SQLXPress Execute SQL statements page for example, which also has multiple panels. When you re-size one of the panels, the others are automatically adjusted so that they do not overlap. The panels always fill the entire area of the page. Also, when the page is re-sized the panels are automatically adjusted. Of course, the panel layout persists.

- Options. There are a few options you can use to control the look and feel of the applications. You can control whether the Databases and Connectivity areas are visible. You can configure Remote MXCI options, and you can control whether system metadata tables show up in the Databases area. That' it.

The MXDM online help provides reference information. It uses the standard Windows Help program which operates as an independent window. There is no context-sensitive help. Help can be invoked from anywhere in the MXDM program by pressing F1. This launches the Help program with the “About This Guide Topic” selected. From there you must navigate to the help topic of interest.

There is no support for printing information in MXDM.

MXDM does not offer much in the way of advanced features for expert users. Keyboard shortcuts are few and far between. The Favourites feature provides limited help in getting things done faster.

SQLXPress

Beginners are able to easily navigate to all areas of the product. The Welcome page contains hyperlinks to the main product areas. Each link has a text caption indicating the target. Most pages also have hyperlinks in the navigation panel (again with text captions).

The Main menu commands together with on-page hyperlinks make all of the product features readily “discoverable”.

The search facility allows the beginner to discover product features based on a keyword search.

The main window has a toolbar with a few buttons which are mostly concerned with navigation. Many pages in the user interface have additional on-page toolbars that are relevant to that particular page.

You can choose to have toolbars with icons only, or with icons and text captions. For beginners, using icons and captions offers an easy way to learn the function of toolbar buttons. Intermediate and expert users may choose to use toolbars with icons only, which can save screen real-estate (although this is much less of a concern with modern, large screen monitors).

There are context (right-click popup) menus almost everywhere. Right-click menus offer an additional way to access commands that are also available on a toolbar, or via links in the navigation bar.

The Bookmarks menu is useful for intermediate users. You can bookmark any page in the application. The page (and the object it refers to) are saved in the bookmark. To return to the page simply select it from the Bookmarks menu. You can also select any page in the application to be the Home page – the selected page is the first page that is displayed when the application starts. You can jump to the Home page at any time by clicking Home on the toolbar.

The intermediate user has lots of ways in which he can customize the product to suit his way of working.

- Grids are used throughout the product to display information. You can customize grids in various ways, for example:
  - Group by one or more columns
  - Sort by one or more columns
  - Change column order
  - Show or hide columns
  - Show or hide grid lines
  - Filter rows

  Customizations persist, so you do not have to re-apply them each time.

  When it makes sense, grid customizations apply to all instances of a particular page type. For example if you have the Tables in a Schema page open in two tabs, each pointing to a different schema, hiding a column in one of the pages, causes the column to be hidden in the other page also. However, applying a filter to table name in one page does not automatically apply the same filter in the other page since it may not be appropriate (although, if you want to apply the same filter it is easily accomplished using a simple cut and paste operation).
- Some pages have multi-panel layouts (for example the Execute SQL Statement page). You can easily adjust the layout to suit your needs and the custom layout is persistent.
- All of the pages in the application can be printed. The printing subsystem offers a report layout editor to control the report format, and a print style editor to control the page setup. You can create custom print styles, which are persistent.
- Options. There are many options available that control how the user interface operates:
  - Confirmation Prompts. These allow you to specify whether you should be prompted to confirm certain actions, for example deleting a query, or exiting with multiple windows open.
  - File transfer options. This allows you to specify a method for transferring files between the PC and the NonStop server.
  - Defaults. You can select defaults for Guardian subvolume, MX schema and MP catalog. These are used to resolve partially qualified names. They are specified on a system (Expand node) by system basis.
  - Remote MXCI options. These are used to configure the Remote MXCI program.
  - Character Sets. By default, SQLXPress uses the SQL-defined character sets to display and validate character data. You can override these with other characters to match the actual character sets used by your application. For example, your application may use ISO8859-15 (which includes the Euro symbol) instead of ISO8859-1.
  - Fonts. By Default, SQLXPress uses the standard Windows fonts for proportional and fixed width text. You can customize the font typeface and size if the defaults are not suitable for your use.

    - Other options. There is a miscellaneous set of options to control other program behaviours. For example, which version of Excel file format to use when exporting data to a spreadsheet, whether to show help embedded in a page, or as a separate page, whether to include SQL object names in search results, and so forth.

- Appearance. The Edit Appearance tool lets you customize the look and feel of the user interface. You can choose a custom color scheme or choose from a set of standard schemes. You can also control the appearance of menus, buttons, edit fields, links etc. The Appearance settings are associated with your logon session. If you have multiple sessions (for example, one per Expand node) you can use a different Appearance for each. For example, using a different color scheme gives you a visual cue as to which system you are logged on to.

The help subsystem is an important resource for the intermediate user.

The SQLXPress help system is comprehensive, easy to use, and unobtrusive.

- Every page in the application has a help topic.
- To view help for a page you can click on the Help link in the navigation panel, or press the F1 key.
- The help topic for a page is displayed within the page itself. There is a Help panel on the right hand side of the page. You can adjust the width of the help panel by simply dragging a splitter control. This arrangement is especially suited to modern, wide-screen monitors. (There is an option to display Help in its own page instead of in-page if so desired).
- In-page help is synchronized with the active page in the application. As you navigate to other pages, the help panel is automatically updated to show the help for each page.
- In many pages there are in-page hyperlinks that pop up a brief hint about how things work on the page. For example, in the Execute SQL Statements page there are hyperlinks that display hints about keyboard shortcuts, and how the history lists work.
- The standard Windows help program can be launched from the Help menu, but with a twist. Instead of launching help as a separate program in its own window, it is embedded into the application in its own tab. This makes switching between Help and the application much easier, you don't have to fuss with sizing the Help window, and the Help window does not overlay the application Window.
- The embedded Help program includes access to all help topics – not just page-specific help. It has all of the features of the standard Windows HTML help viewer including an index and a search facility.

The expert user wants shortcuts for everything. He knows what the product can do, and does not want to be held back by the user interface elements that are there to help beginners. He just wants to “get 'er done”.

SQLXPress supports the expert user in several ways:

- Bookmarks can be used to jump directly to any page in the application.
- Any page can be used as the Home page.
- Keyboard shortcuts are everywhere. The expert can use the keyboard instead of the mouse to get things done faster. For example, there are keyboard shortcuts for:

    - Navigation. All hyperlinks in the navigation and history panels have keyboard shortcuts.

  - All of the Window and Tab management commands.
  - Input widgets. Many input widgets have Alt+<letter> shortcuts, for example the Include File Details checkbox in the Tables in a Schema page can be toggled by pressing Alt+D.
  - Continue and Go Back in all wizards.
  - Right-click (context) menus can be accessed via the keyboard.
  - All the standard Edit commands.
  - Various page-specific shortcuts.
- There is a Shortcuts command under the Help menu that lists the various keyboard shortcuts making them easy to learn.
- The Search facility provides quick access to pages in the application, and SQL objects.
  - Search is accessible via the main toolbar.
  - You can type in one or more search keywords.
  - The soundex algorithm is used to match results with the keywords, so its not necessary to spell keywords exactly.
  - Match exact, and match all options are provided.
  - The search results include links to pages that match.
  - For matching SQL object names a link to the object properties page is included in the results.
  - Results are sorted by relevancy.
  - You can easily select previous search criteria and re-execute the search.

In Depth Comparison of VQP And SQLXPress

The HP Visual Query Planner (VQP) is a Windows application that is shipped as part of SQL/MX. It uses an ODBC connection to connect to SQL/MX.

The Visual Query Analyst (VQA) is a program developed by the HP Advanced Technology Centre (ATC) which is available for download from the ATC utilities web page. The VQA is written in Java and connects to SQL/MX via a JDBC connection.

Curiously, the VQP and VQA programs are almost identical. The user interface is a little bit different, and VQA has a couple of features not available in VQP, but their operation is pretty much the same. The discussion in this section is generally applicable to both VQP and VQA. Major differences are noted in the discussion.

SQLXPress ships with a number of different Windows client programs.

The main SQLXPress client supports displaying execution plans graphically from both the Execute SQL Statements page, and the Visual Query Builder.

In addition, there is a separate client program called the Visual Query Tuner (VQT) which is dedicated to query tuning. It has a rich set of features to support the query tuning work-flow. You can launch the VQT from the Windows Start menu, or from within the main SQLXPress client.

In the rest of this section we will mostly compare the HP VQP with the SQLXPress VQT. We will also cover the features of VQA that are not supported in VQP.

The comparison is broken down as follows:

- **Breadth.** This compares the product feature sets,
- **Depth.** This compares the richness of implementation of various features.
- **User Interface.** This compares the design of the user interface.

Breadth

Features In Common

The following features are available in both HP VQP and SQLXPress VQT.

- Display execution plan summary.
- Display execution plan in a treeview.
- Print execution plan treeview.
- Display plan operator details.
- Display Control Query Shape statement that reflects the execution plan.
- Execute Control Query Shape statement.
- Execute Control Query Default statements.

The following additional features are available in both HP VQA and SQLXPress VQT.

- Execute query, and display compile and execute times.
- Display visual plan diagram.

Features in HP VQP, but not in SQLXPress VQT

- Get execution plan for a statement in a compiled module.
- Save plan to PC file.

Features in SQLXPress VQT but not in HP VQP or VQA

- Full support for SQL/MP execution plans.
- Auto-tuner analyzes query and suggests changes to improve performance.
- Record query tuning experiments as a set of snapshots for a query.
- Save snapshots in the SQLXPress execution plans database (on the NonStop server).
- Execute statements containing parameters.
- Support for DEFINEs.
- Import query from MX module, MP program, Visual Query Builder
- Captures per-table execution statistics.
- Performance charts compare execution statistics from multiple snapshots.
- Display access paths for the tables in a query.
- Display histogram and partition statistics for tables in a query.
- Display configuration information for disk volumes.
- Display status information for disk volumes (including DP2 cache statistics and MX buffer statistics).
- Display table and index properties (including DDL).
- Update Statistics (both MX and MP).
- Create Index.
- Populate Index.
- Drop Index.

Depth

I'd like to start by talking about the work-flow of the query-tuning process.

Query tuning is about answering the question “How do I make this query run faster?”.

In query-tuning, it is assumed that the query itself is the problem to be worked on. In other words, the query is not running slowly because the system doesn't have enough capacity, or the disk process is not configured correctly. (These are valid problems that might need to be addressed, but they are not what query tuning is about).

Before you start to tune a query, you need to establish a baseline that records the current state of affairs. If you don't have a baseline, how can you tell if you are making any progress?

So, the first order of business is to save off the baseline data:

- The query text.
- Any CONTROL statements that are being used with the query.
- The execution plan.
- The performance characteristics.

For convenience, I'll call this set of data a “query snapshot”.

Once you have the baseline you can start to try some tuning experiments.

A tuning experiment is where you try changing something about the query, and test the effect of the change. Did things improve, or did you make it worse?

Here is a list of things you can change to try to improve the performance of your query:

- Update table statistics.
- Add or replace an index.
- Change the query text.
- Use CONTROL statements to influence the plan generated by the compiler.

Once you have made one or more changes, you need to create a new query snapshot that captures the details of the tuning experiment.

You can tell if you have made any progress by comparing the new execution plan and query performance statistics with the baseline.

This is an iterative process. Some experiments will make things worse, so you will abandon the changes. Others will improve things, so you'll want to keep the changes, and perhaps perform more experiments seeking further improvements.

I wanted to describe the query tuning work-flow first so that I could compare how each product supports it.

Recording Query Snapshots

How do the products help you save data about each tuning experiment (snapshot)?

HP VQP

- The VQP allows you to save the query text and execution plan to a file on your PC.
- Any CONTROL statements that were used are not saved. (The VQA does a better job here as it will save CONTROL QUERY DEFAULT and CONTROL QUERY SHAPE statements in the snapshot file).
- You can open a previously saved plan, this will load the query text and the plan. CONTROL statements are not loaded. (The Root node details for the plan contain information on the CONTROL statements that were used when the plan was generated, but you would need to manually re-enter them if you wish to use them for your next experiment). (VQA will reload CONTROL statements.)
- To keep a history of tuning experiments, you need to establish a naming convention for query snapshot files, each snapshot is saved in a separate file, with a user-specified name.
- VQP does not collect query performance data, and so the snapshot does not include any. You must manually collect this, for example using MXCI, and manually record the data for each snapshot. (Note that in VQA, you can execute a query – although parameters are not supported, and it will display some basic performance information, like compile time, execution time, and number of rows returned. This data is not saved in the query snapshot.)

SQLXPress VQT

- The VQT saves query snapshots in a NonStop SQL database.
- When you take the first (baseline) snapshot, you are prompted for a query name. The snapshot is identified by the query name, plus the time when the snapshot was taken. When you take subsequent snapshots, they are automatically saved under the same query name plus the snapshot time.
- You can open a previously saved query by name, which gives you immediate access to all of its snapshots.
- You can easily access any snapshot of the query you are working on by clicking the First, Last, Prev, or Next buttons. Alternatively, you can select a snapshot from a drop-down list.
- When you take a snapshot, the VQT saves the following:
  - The query text.
  - All CONTROL statements.
  - Parameter values used when running the query.
  - File names mapped by DEFINES in the query.
  - Plain text description of the query.
  - Plain text comments about the snapshot.
  - The execution plan.

    - Execution statistics including compile time, execution time, number of rows returned, and per-table statistics including records accessed, records used, number of messages, and message bytes.

- A single snapshot can include multiple execution statistics samples. The query is run multiple times, and the statistics saved from each run. The idea here is to eliminate cache bias. A query will get better response time if the data it requires is cached in memory as it won't require physical IOs to fetch the data. When comparing query response times you want to eliminate differences due to data being cached or not. When you run the query two or three times in quick succession, the second and third executions will benefit from the data most likely being in cache.

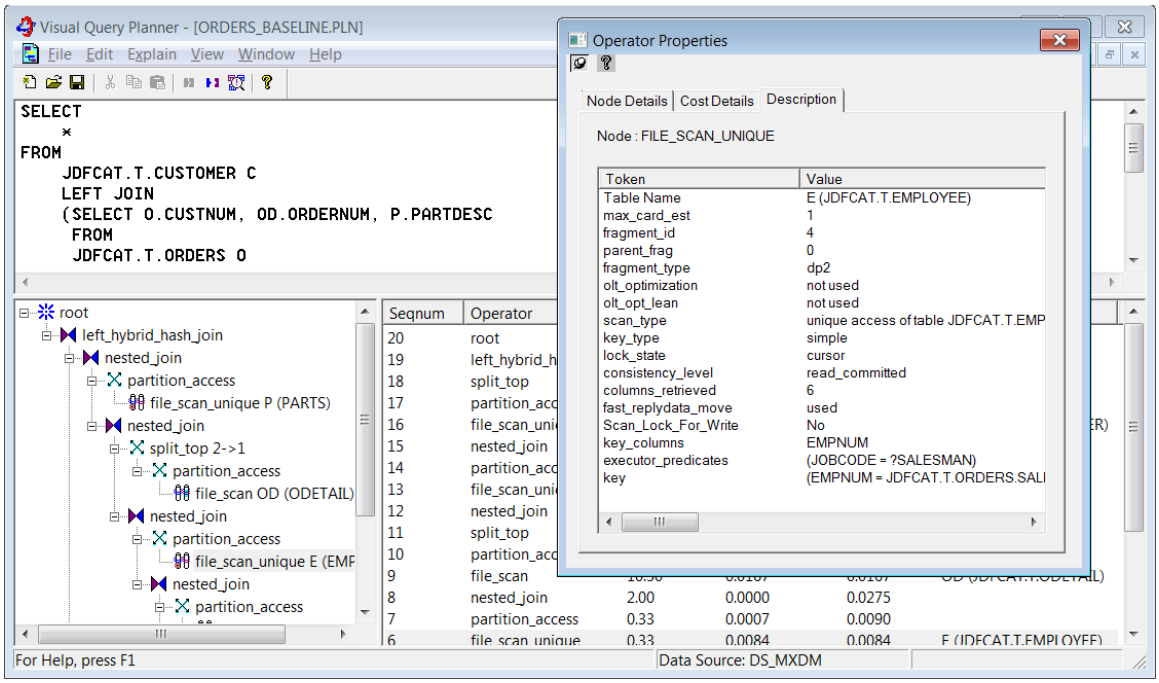
Comparison

- Managing snapshots with the HP VQP (and VQA) is a more labour-intensive process than with SQLXPress VQT.
- With VQP the user must explicitly Save As each snapshot, carefully choosing file names to help organize the set of snapshots. Navigating back to a previous snapshot involves opening the appropriate file name.
- With VQT, after the baseline snapshot has been taken, snapshot names are assigned automatically. Navigating between snapshots is supported directly from the toolbar with no need to use a file open dialog.
- VQT allows you to document a snapshot by entering explanatory comments. VQP does not offer this.
- A VQT snapshot captures more information than a VQP snapshot, in particular it captures performance metrics.
- VQT supports parameters and DEFINES, VQP does not.

Viewing Execution Plans

HP VQP

- The VQP displays a grid summarizing the operators in the plan, and a treeview showing the operators organized in a tree structure.
- The Properties command opens a popup window that shows the properties of an operator.
- The VQA displays the plan in the same way, but also includes a plan diagram which is displayed in a popup window.



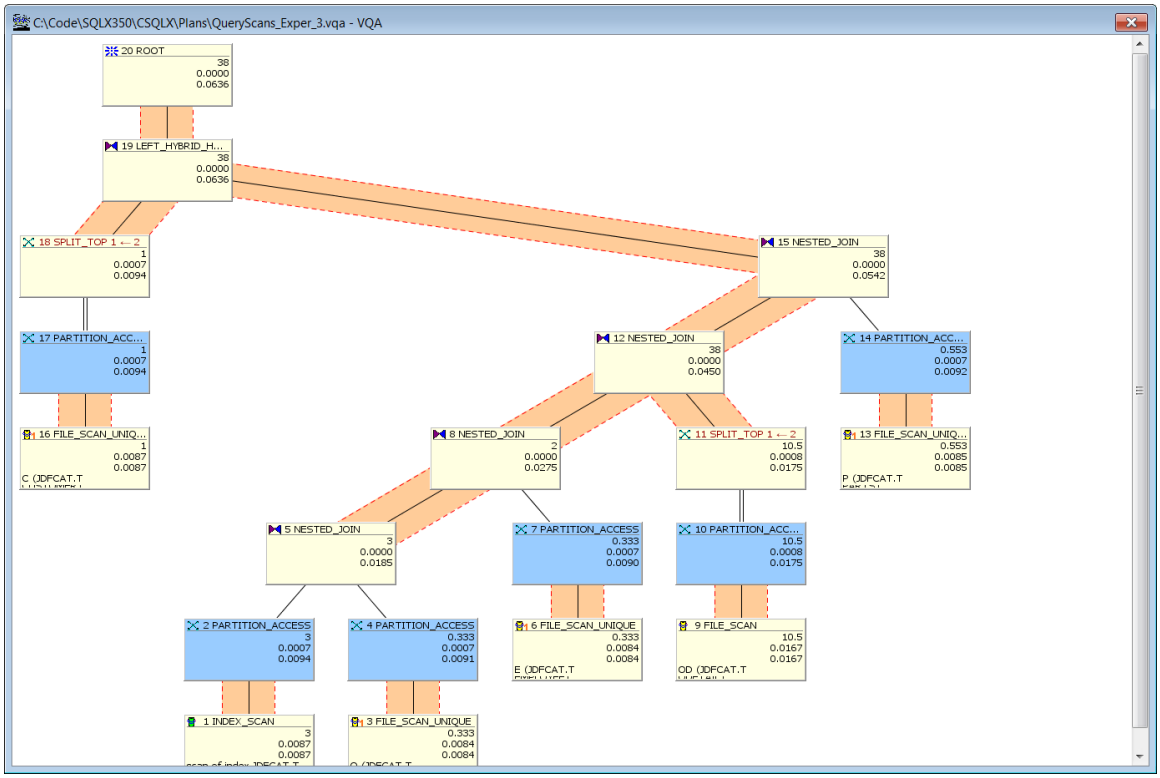
The VQP Main Window

The VQP main window has an editor at the top in which you enter the query text. The plan treeview is on the lower left, and the operator summary grid is on the lower right. To view operator property detail you open the Operator Properties window.

The plan tree is not particularly intuitive, and can be hard to understand, especially for more complex plans with a large number of operators.

The VQA offers a plan diagram that is more intuitive (There is an example on the next page).

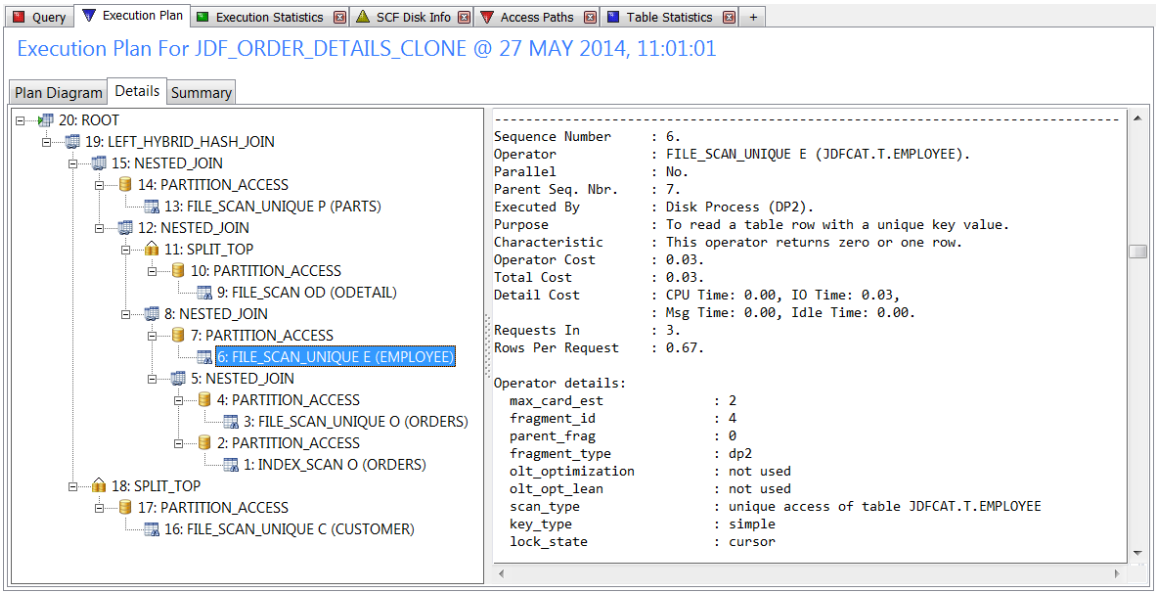
The plan tree is displayed in a popup window. The root operator is shown at the top of the tree, with the left child down and to the left, and the right child down and to the right. This is easier to understand than the treeview used in the VQP. The downside is the lack of detail shown for the operators.



The VQA Plan Tree Window

SQLXPress VQT

- The VQT shows three different views of the execution plan, each in its own tab.
- The Summary tab has a grid listing summary data for each plan operator.
- The Details tab contains a treeview outline of the plan, and a text panel listing the details of each operator. The text is synchronized with the treeview, when you select a node in the treeview, the plan text is scrolled to show the corresponding operator text.
- The Plan Diagram tab shows a plan tree diagram.

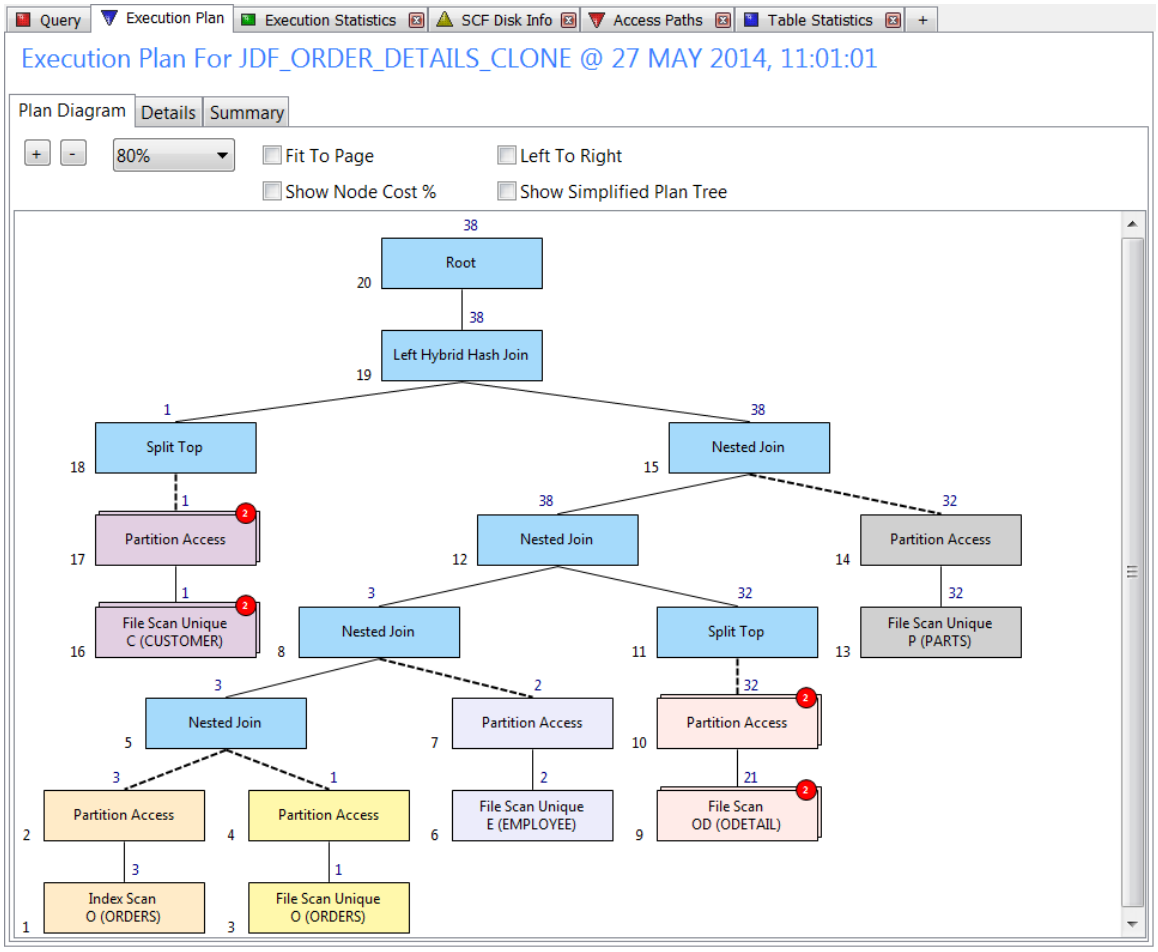


VQT Plan Details Tab

The Plan Diagram is much more intuitive, and also shows lots of details. (There is an example on the next page).

On the plan diagram, you can view complete operator details simply by moving the mouse over an operator. A panel is displayed showing the operator details. Simply move the mouse away and the panel is dismissed.

The Show Simplified Plan option displays the plan diagram without certain node types (like ESP_EXCHANGE, and PARTITION_ACCESS). This improves the readability without losing key information - the diagram has other ways of denoting plan fragment boundaries, and degrees of parallelism.



VQT Plan Diagram

Comparison

Both the HP VQP and the SQLXPress VQT show a summary grid listing the plan operators.

They both show the shape of the plan in a treeview.

The VQP Operator Details popup window is a bit awkward. You have to manage its position and size, and either have it stay on top of the main window, or keep switching between it and the main window. It shows operator details in three tabs. The VQA does a slightly better job, the Properties Window show all of the operator details in a single panel.

The VQT has a more elegant solution in which the operator details are displayed in a text panel that is synchronized to the treeview.

The VQP does not offer a plan diagram, although the VQA does. VQA plan diagrams do not offer direct access to operator details.

The VQT also offers plan diagrams. VQT plan diagrams provide direct access to all operator details.

Tuning Experiments - Making Changes

How do the products support making changes that will help improve performance?

HP VQP

- **Statistics:**
  - View table statistics: Not supported. Use MXDM.
  - Update Statistics: Not supported. Not supported by MXDM either. Use the command line (MXCI).
- **Access Paths:**
  - View access paths for tables in the query: Not supported. For each table in the query, use MXDM to view the access paths.
  - Create Index: Not supported. Use MXDM.
  - Drop Index: Not supported. Use MXDM.
  - Populate Index: Not supported. Use MXCI.
- **Change Query Text:** Supported. Use Query text editor to make changes.
- **Use Control Statements:** Supported. Use Control Query Shape window to issue CQS, it also supports CQD. (Actually the VQA is better at this).

SQLXPress VQT

- **Statistics:**
  - View table statistics: Supported. Table statistics page shows histogram and partition statistics for the tables in the query
  - Update Statistics: Supported. Use Update Statistics wizard to schedule Update Statistics to run as a background task. Can be run immediately, or at a scheduled time.
- **Access Paths:**
  - View access paths for tables in the query: Supported. View Access Paths page shows the available access paths for each table in the query.
  - Create Index: Supported. Use Create Index wizard.
  - Drop Index: Supported. Use Drop Index page.
  - Populate Index: Supported. Use Populate Index wizard to schedule the operations as a background task.
- **Change Query Text:** Supported. Use Query text editor to make changes. (Supports auto-completion and syntax highlighting.)
- **Use Control Statements:** Supported. You can enter CQD, CQS, and CONTROL TABLE statements.

Comparison

The HP VQP allows you to change the query text, and has some support for CONTROL statements. It offers no help with statistics or access paths. In contrast, the SQLXPress VQP helps in all four areas.

Tuning Experiments - Measuring Progress

The first thing you want to know after performing a tuning experiment is “Does the query run faster?”.

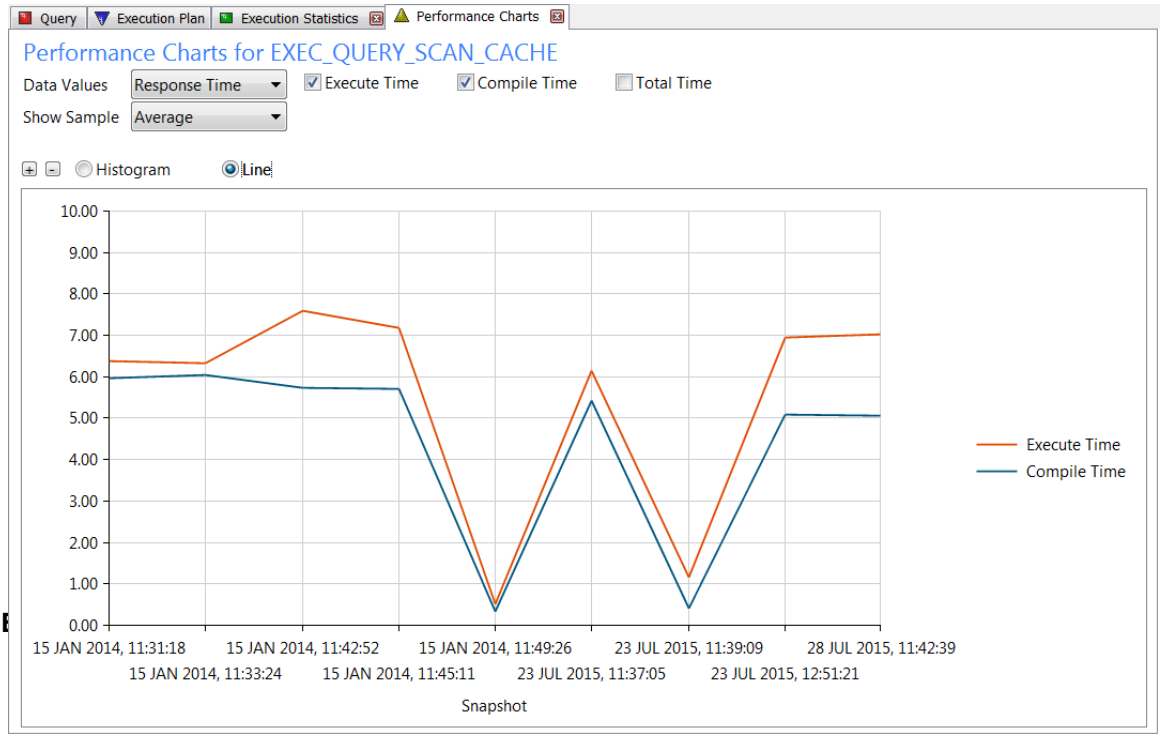
How do the products allow you to measure performance differences?

HP VQP

- The VQP does not support any performance metrics.
- The VQA allows you to run the query, it displays the query results and shows compile time and execution time. Unfortunately, it does not support parameters. If you have parameters in your query, you must rewrite it to use literals instead. However, a query rewritten to use literals may not have the same execution plan as the query with parameters, so the response time statistics may not be reliable. The VQA does not save the response time data when you save the plan, so you will need to record them manually.
- Neither the VQP nor the VQA show execution statistics. These are very useful in understanding the performance characteristics of the query. For example, they can tell you whether a tuning experiment resulted in the query reading more records than before. You can use the command line (MXCI) to display execution statistics. You must then record them for later reference.

SQLXPress VQT

- The query compile time and execution time is captured as part of the snapshot, and is displayed in the Execution Statistics tab.
- Per-table execution statistics are also captured.
- Multiple statistics “samples” can be taken for the same snapshot (to eliminate cache bias).
- The View Performance Data page displays charts comparing the performance metrics for each snapshot. Here is an example:



First E

VQT Performance Charts

Comparison

The HP VQP and VQA do not support the recording and comparison of performance metrics. You must revert to using MXCI to capture execution statistics.

The SQLXPress VQT automatically captures performance metrics when you take a snapshot. You can easily compare the performance of your various tuning experiments.

User Interface

HP VQP

The main window has three main areas:

- The text editor at the top of the window is used to enter the query text. The editor does not offer syntax highlighting or auto-completion.
- The plan treeview is shown on the lower left. It displays the plan tree when the user executes the Explain command.
- The plan summary is shown on the lower right. It lists summary information about the plan operators.

To view plan operator details, the user opens the Operator Properties window. This is a modeless dialog with three tabs. When the user selects an operator in the treeview or summary grid, the operator property details are shown in the Operator Properties window. The problem here is that clicking on an operator on the main window causes the Properties window to be hidden, and you must switch back to it to view the properties. As a workaround you can force the Operator Properties Window to stay on top of the main window – which of course can obscure some of the main window.

The Show Query Shape Command opens the Query Shape dialog. It shows the CQS statement corresponding to the current plan. You can edit the CQS statement, and click the Force Shape button to issue the modified CQS when you next Explain the query.

“Show Shape” is a bit of a misnomer for this window as it can also be used to execute CONTROL QUERY DEFAULT and CONTROL TABLE statements. Unfortunately, you can only execute one CONTROL statement at a time.

Like the Operator Properties window, the Show Shape window is an awkward to use modeless dialog.

A brief word about VQA. The layout of the main window is pretty much the same as the VQP. One advantage is that you can include CONTROL QUERY DEFAULT statements in the query text editor provided they appear one to a line, with a trailing semi-colon. By the way, you can't use the CQD short form. The VQA Show Shape facility is a bit tricky. You must first Enable Show Shape, the execute the Show Shape command. A Show Shape window appears with two text panels. The lower panel shows the shape from the latest explain. You can copy the CQS to the top panel and use this to force the shape on the next explain.

SQLXPress VQT

By default the VQT has three tabs on its main window:

- The Query tab is where you enter data about the query. This includes the query text, query description, snapshot comments, parameter values, define file mappings, and CONTROL statements.
- The Execution Plans tab displays the plan for the current snapshot.
- The Execution Statistics tab displays the performance metrics for the snapshot.

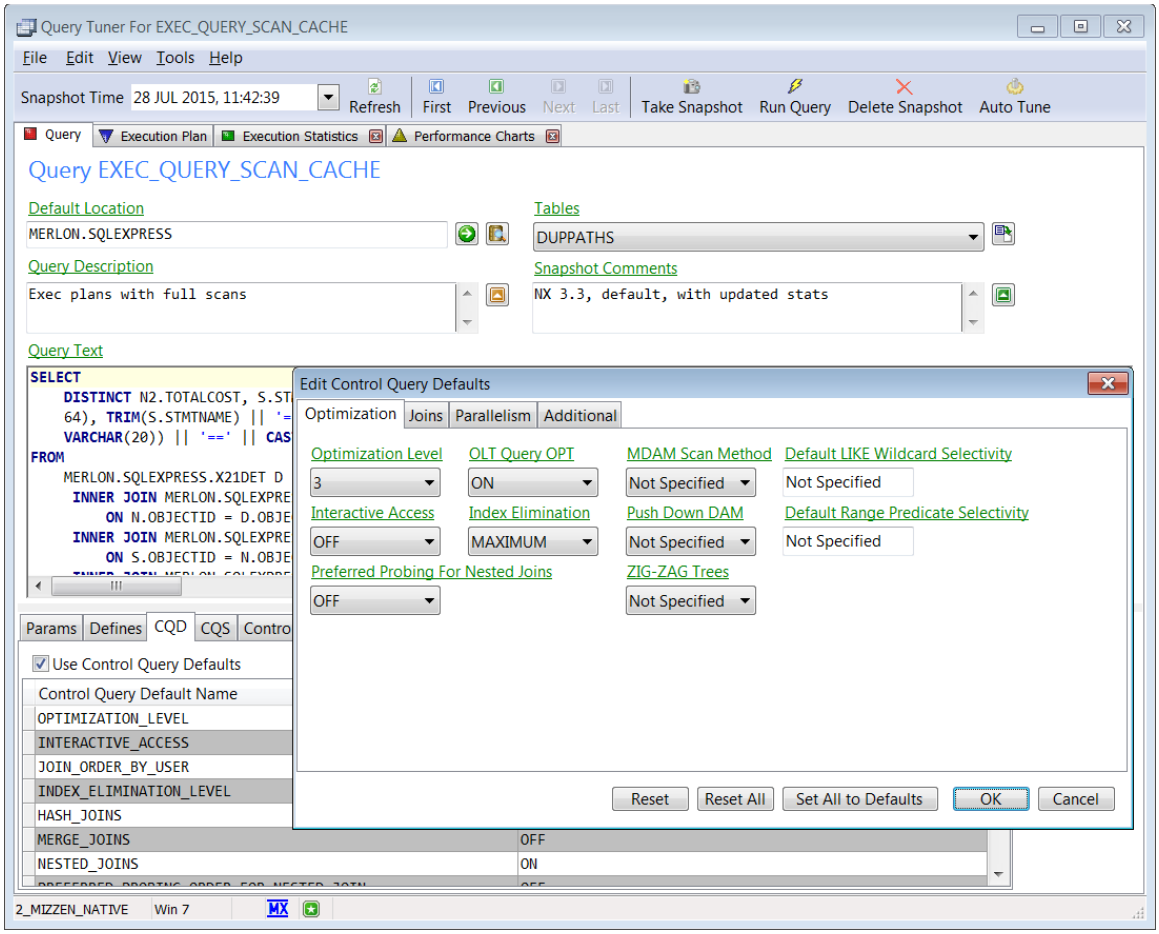
You can open these additional tabs:

- Performance Charts. Displays charts with performance metrics for the query snapshots.
- Access Paths. Lists the available access paths for the tables in the query.
- Table Statistics. Displays the histogram and partition statistics for the tables in the query.

The Query tab is where you enter data for the snapshot. The query editor supports auto-completion for table names, column names, and SQL functions. Syntax highlighting helps the user correct typing errors.

There are grids for entering parameters, defines, CQDs, and Control Table statements.

There is a special CQD editor which allows the user to specify CQDs without having to remember the syntax.



Editing CQDs

You can issue a show shape command to generate a CQS statement. You can edit the CQS and have it applied when you create the snapshot.

