HP NonStop SQL/MX Connectivity Service Manual for SQL/MX Release 3.2.1

HP Part Number: 663856-004
Published: August 2013
Edition: J06.15 and all subsequent J-series RVUs and H06.26 and all subsequent H-series RVUs

Legal Notice

The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

Export of the information contained in this publication may require authorization from the U.S. Department of Commerce.

OSF MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THE OSF MATERIAL PROVIDED HEREIN, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

OSF shall not be liable for errors contained herein or for incidental consequential damages in connection with the furnishing, performance, or use of this material.

Contents

About this document

This manual describes how to install and manage the HP NonStop™ SQL/MX connectivity service. This product enables applications developed for the Microsoft® Open Database Connectivity (ODBC) application programming interface and the Java Database Connectivity (JDBC) API to use HP NonStop SQL/MX to access HP NonStop SQL databases on an HP NonStop system.

Supported Release Version Updates (RVUs)

This manual supports J06.15 and all subsequent J-series RVUs and H06.26 and all subsequent H-series RVUs, until otherwise indicated by its replacement publications.

Audience

This manual is intended for all MXCS users, including system administrators and database administrators who install, configure, and manage MXCS.

You must be familiar with:

- Microsoft ODBC 3.5 API
- SQL/MX 3.x
- Microsoft Windows 98, Windows NT 4.0 (workstation or server), Windows 2000, Windows 2000 Server, Windows 2003, Windows XP, and Windows 7 if client computers are running one of these operating systems

New and Changed Information in This Edition

Changes to 663856-004 manual:

- Added SSL support in "Starting an MXCS Service manually" (page 20).

Changes to 663856-003 manual:

- Updated the Additional Information section and added a warning in "Starting an MXCS Service manually" (page 20).
- Updated the note in "Configuring MFC from the MXCS mode" (page 49).
- Added a note in "Design of MFC" (page 49).
- Updated "Benefits of MFC" (page 51).
- Added a note under "Migrating from SQL/MX Release 2.3.4 to SQL/MX Release 3.2" (page 24).

Notation Conventions

Hypertext Links

Blue underline is used to indicate a hypertext link within text. By clicking a passage of text with a blue underline, you are taken to the location described. For example: This requirement is described under Backup DAM Volumes and Physical Disk Drives.

General Syntax Notation

This list summarizes the notation conventions for syntax presentation in this manual.

UPPERCASE LETTERS. Uppercase letters indicate keywords and reserved words. Type these items exactly as shown. Items not enclosed in brackets are required. For example:

```
MAXATTACH
```

lowercase italic letters. Lowercase italic letters indicate variable items that you supply. Items not enclosed in brackets are required. For example:

```
file-name
```

computer type. `Computer type` letters within text indicate C and Open System Services (OSS) keywords and reserved words. Type these items exactly as shown. Items not enclosed in brackets are required. For example:

```
myfile.c
```

italic computer type.`Italic computer type` letters within text indicate C and Open System Services (OSS) variable items that you supply. Items not enclosed in brackets are required. For example:

```
pathname
```

[ ] Brackets. Brackets enclose optional syntax items. For example:

```
TERM [\system-name.]$terminal-name
```

```
INT[ERRUPTS]
```

A group of items enclosed in brackets is a list from which you can choose one item or none. The items in the list can be arranged either vertically, with aligned brackets on each side of the list, or horizontally, enclosed in a pair of brackets and separated by vertical lines. For example:

```
FC [ num ]
```

```
[ -num ]
```

```
[ text ]
```

```
K [ X | D ] address
```

{ } Braces. A group of items enclosed in braces is a list from which you are required to choose one item. The items in the list can be arranged either vertically, with aligned braces on each side of the list, or horizontally, enclosed in a pair of braces and separated by vertical lines. For example:

```
LISTOPENS PROCESS { $appl-mgr-name }
```

```
{ $process-name }
```

```
ALLOWSU { ON | OFF }
```

| Vertical Line. A vertical line separates alternatives in a horizontal list that is enclosed in brackets or braces. For example:

```
INSPECT { OFF | ON | SAVEABEND }
```

... Ellipsis. An ellipsis immediately following a pair of brackets or braces indicates that you can repeat the enclosed sequence of syntax items any number of times. For example:

```
M address [ , new-value ]…
```

```
[ - ] {0|1|2|3|4|5|6|7|8|9}…
```

An ellipsis immediately following a single syntax item indicates that you can repeat that syntax item any number of times. For example:

```
"s-char…"
```

Punctuation. Parentheses, commas, semicolons, and other symbols not previously described must be typed as shown. For example:

```
error := NEXTFILENAME ( file-name ) ;
```

```
LISTOPENS SU $process-name.#su-name
```

Quotation marks around a symbol such as a bracket or brace indicate the symbol is a required character that you must type as shown. For example:

```
"[" repetition-constant-list "]"
```

Item Spacing. Spaces shown between items are required unless one of the items is a punctuation symbol such as a parenthesis or a comma. For example:

```
CALL STEPMOM ( process-id ) ;
```

If there is no space between two items, spaces are not permitted. In this example, no spaces are permitted between the period and any other items:

```
$process-name.#su-name
```

Line Spacing. If the syntax of a command is too long to fit on a single line, each continuation line is indented three spaces and is separated from the preceding line by a blank line. This spacing distinguishes items in a continuation line from items in a vertical list of selections. For example:

```
ALTER [ / OUT file-spec / ] LINE
[ , attribute-spec ]…
```

Related Documentation

This manual is part of the HP NonStop SQL/MX library of manuals.

NonStop SQL/MX customer library

The manuals in the SQL/MX customer library are listed here for your convenience.

- **Introductory guides**

SQL/MX Comparison Guide for SQL/MP Users	Describes differences between NonStop SQL/MP and NonStop SQL/MX databases.
SQL/MX Quick Start Guide	Describes basic techniques for using SQL in the SQL/MX conversational interface (MXCI). Includes information about installing the sample database.

- **Installation guides**

SQL/MX Installation and Upgrade Guide	Describes how to plan for, install, create, and upgrade a SQL/MX database.
SQL/MX Management Manual	Describes how to manage a SQL/MX database.
NSM/web Installation Guide	Describes how to install NSM/web and troubleshoot NSM/web installations.

- **Reference manuals**

SQL/MX Reference Manual	Describes the syntax of SQL/MX statements, MXCI commands, functions, and other SQL/MX language elements.
SQL/MX Messages Manual	Describes SQL/MX messages.
SQL/MX Glossary	Defines SQL/MX terminology.

- **Connectivity manuals**

SQL/MX Connectivity Service Manual	Describes how to install and manage SQL/MX Connectivity Service (MXCS), which enables ODBC and other connectivity APIs to use NonStop SQL/MX.
SQL/MX Connectivity Service Administrative Command Reference	Describes the SQL/MX Administrative Command Library (MACL) available with the SQL/MX conversational interface (MXCI).

ODBC/MX Driver for Windows	Describes how to install and configure HP NonStop ODBC/MX for Microsoft Windows, which enables applications developed with ODBC API to use NonStop SQL/MX.
SQL/MX Remote Conversational Interface (RMXCI) Guide	Describes how to use SQL/MX Remote Conversational Interface to run the RMXCI commands, and SQL statements interactively or from script files.
HP NonStop MXDM User Guide for SQL/MX Release 3.2.1	Describes how to use the NonStop SQL/MX Database Manager (MXDM) to monitor and manage the SQL/MX database.
HP NonStop JDBC Type 2 Driver Programmer's Reference	Describes the NonStop JDBC Type 2 Driver functionality, which allows Java programmers to remotely develop applications deployed on client workstations to access NonStop SQL/MX databases.
HP NonStop JDBC Type 4 Driver 3.0 Programmer's Reference	Describes the NonStop JDBC Type 4 Driver functionality, which allows Java programmers to remotely develop applications deployed on client workstations to access NonStop SQL/MX databases.

- **Migration guides**

SQL/MX Database and Application Migration Guide	Describes how to migrate databases and applications to NonStop SQL/MX, and how to manage different versions of NonStop SQL/MX.
NonStop NS-Series Database Migration Guide	Describes how to migrate NonStop SQL/MX, NonStop SQL/MP, Enscribe databases and applications to HP Integrity NonStop NS-series systems.

- **Data management guides**

SQL/MX Data Mining Guide	Describes the SQL/MX data structures and operations for data mining.
SQL/MX Report Writer Guide	Describes how to produce formatted reports using data from an SQL/MX database.
DataLoader/MX Reference Manual	Describes the features and functions of the DataLoader/MX product, a tool to load SQL/MX databases.

- **Application development guides**

SQL/MX Programming Manual for C and COBOL	Describes how to embed SQL/MX statements in ANSI C and COBOL programs.
SQL/MX Query Guide	Describes how to understand query execution plans and write optimal queries for an SQL/MX database.
SQL/MX Queuing and Publish/Subscribe Services	Describes how NonStop SQL/MX integrates transactional queuing and publish/subscribe services into its database infrastructure.
SQL/MX Guide to Stored Procedures in Java	Describes how to use stored procedures that are written in Java within NonStop SQL/MX.

- **Online help**

SQL/MX Database Manager Help	Contents and reference entries from the SQL/MX Database Manager User Guide.
Reference Help	Overview and reference entries from the SQL/MX Reference Manual.
Messages Help	Individual messages grouped by source from the SQL/MX Messages Manual.
Glossary Help	Terms and definitions from the SQL/MX Glossary.

NSM/web Help	Context-sensitive help topics that describe how to use the NSM/web management tool.
Visual Query Planner Help	Context-sensitive help topics that describe how to use the Visual Query Planner graphical user interface.

The NSM/web, SQL/MX Database Manager, and Visual Query Planner help systems are accessible from their respective applications. You can download the Reference, Messages, and Glossary online help from the HP Software Depot, at http://www.software.hp.com. For more information about downloading online help, see the *SQL/MX Release 3.2.1 Installation and Upgrade Guide*.

Publishing History

This new edition is a revision of the manual originally titled *NonStop Tuxedo System Supplement to NonStop TS/MP D42.00 Manuals.*

Part Number	Product Version	Publication Date
640327-001	HP NonStop MXCS 3.0	February 2011
663856-001	HP NonStop MXCS 3.1	October 2011
663856–002	HP NonStop MXCS 3.2	August 2012
663856–003	HP NonStop MXCS 3.2.1	February 2013
663856–004	HP NonStop MXCS 3.2.1	August 2013

HP encourages your comments

HP encourages your comments concerning this document or this product. We are committed to providing products and documentation that meet your needs. Send any errors found or suggestions for improvement to **docsfeedback@hp.com**.

Include the document title and part number of the document.

1 Overview of MXCS

HP NonStop SQL/MX Connectivity Services (MXCS) enables applications developed for the Microsoft® Open Database Connectivity (ODBC) application programming interface (API) and the Java Database Connectivity (JDBC) API to use SQL/MX to access a NonStop SQL database on a NonStop server.

MXCS is managed by NSM/web, which is a web-based graphical user interface (GUI) management tool for configuring and monitoring the server components and data sources. For information on NSM/web, see the *HP NonStop SQL/MX Release 3.2.1 Installation and Upgrade Guide* and the *NSM/web online help*.

Topic

Architecture

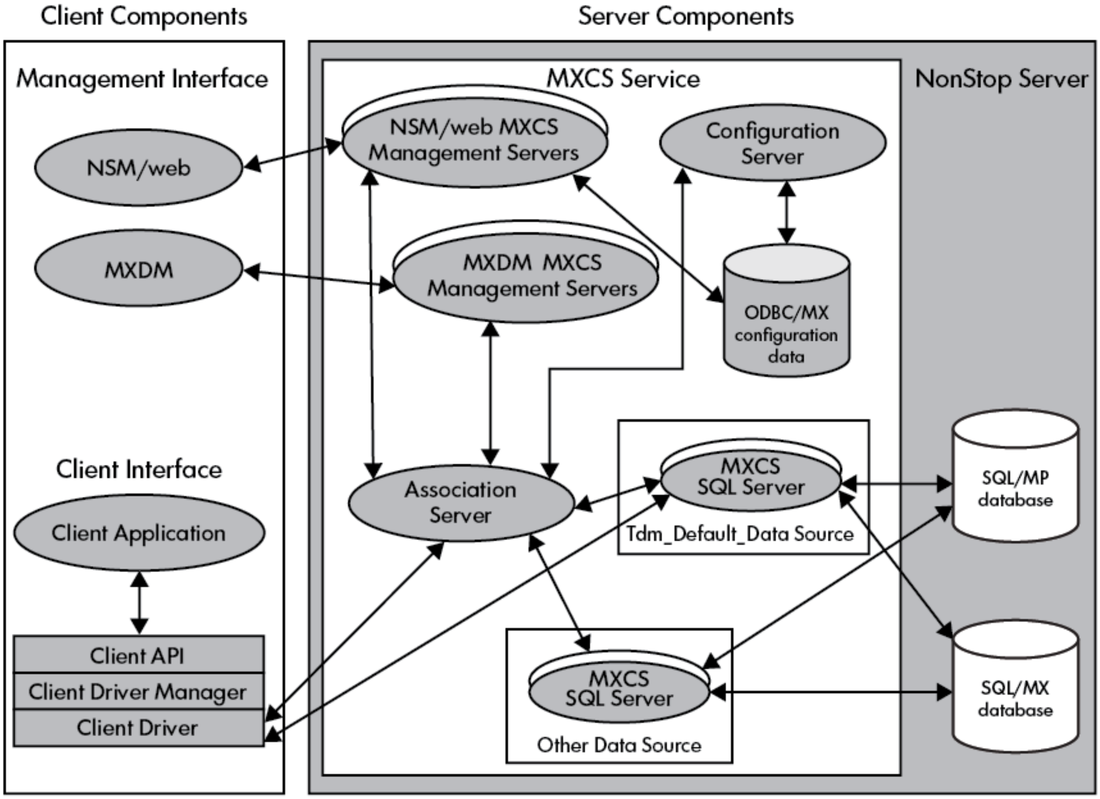
Components

Server components are installed on NonStop systems. The following are the server and client components:

- Association server: Starts and manages the MXCS server processes (MXOCFG and MXOSRVR processes) and associates a client application with a specific MXCS SQL server.
- Configuration server: Manages the MXCS configuration data.
- MXCS SQL Server: Provides access to a NonStop SQL database. It is a MXOSRVR process.
- NSM/web: Manages and configures MXCS service. Requires an installed web server. NSM/web is not a server component because it does not run in the host. It is a client component.

  Starting with SQL/MX Release 3.2, a new tool, called NonStop SQL/MX Database Manager (MXDM), is available. The new tool enables you to view and manage MXCS objects. For information on viewing and managing the MXCS objects, such as MXCS Services, data sources, and MXCS user permissions, see the *HP NonStop MXDM User Guide for SQL/MX Release 3.2.1*.

Figure 1 MXCS Architecture - Interactions Among Three Components



Data Sources

A datasource is a logical grouping of MXOSRVRs. The data source definitions include the basic configuration performed on the data source. MXCS requires data source definitions on the NonStop server, and for ODBC clients on the client workstation.

Server Data Sources

Server data sources reside on a NonStop server. Each server data source represents a pool of SQL MXCS servers that share the same NonStop SQL context. Use NSM/web or MXCI in MXCS mode (MACL) to add, remove, configure, and monitor server data sources. See "Managing MXCS Server Data Sources" (page 27).

Default Data Source

MXCS provides a preconfigured, default server data source on the server called TDM_Default_DataSource. You can stop and start the TDM_Default_DataSource using NSM/web, but you cannot delete it. The TDM_Default_DataSource has these functions:

- Clients can connect to TDM_Default_DataSource and access a NonStop SQL database with no special configuration required.
- When a client tries to connect to a data source in the NonStop system, the association server connects the client to TDM_Default_DataSource if one of the following conditions exists:
  - the data source name and the NonStop system data source name do not match (including the case).
  - the NonStop system data source does not exist.

NOTE: TDM_Default_DataSource must be in a started state for the client to be connected. If an ODBC client tries to connect to any server data source that is stopped, MXCS rejects the connection.

Client Data Sources

Client data sources are used for ODBC/MX connections and reside on a client workstation. Each client data source is a logical name that identifies the attribute values used to connect to the MXCS service and defines some local behavior for the connection. Use the Microsoft ODBC Data Source Administrator to add, configure, and remove ODBC/MX client data sources.

Connections

Connecting a Client to a Data Source

Figure 2 (page 14) shows the flow of data among the client and server components.

The MXCS association server starts and manages the MXCS SQL servers associated with MXCS data sources in the NonStop system. When a client requests a connection to a data source, the association server also associates the client with an available MXCS SQL server. The following scenario describes how a client application connects to an MXCS SQL server through the association server:

1. The client application sends, via an API call, a connection request to the MXCS service. The client specifies an IP address or the NonStop server Domain Naming Service (DNS) name and port number that is defined when the association server is started up. ODBC Clients connect to MXCS by connecting via a client data source, which contains the connection-specific information. JDBC clients specify the name of the Server Data source and other connection details as part of the connect API call.
2. The association server grants the connection request and associates the client application with an available MXCS SQL server for the data source. (If the association server does not find the requested data source name, it assigns a server associated with the default data source.) If all available MXCS servers already have a connection, the request is denied.

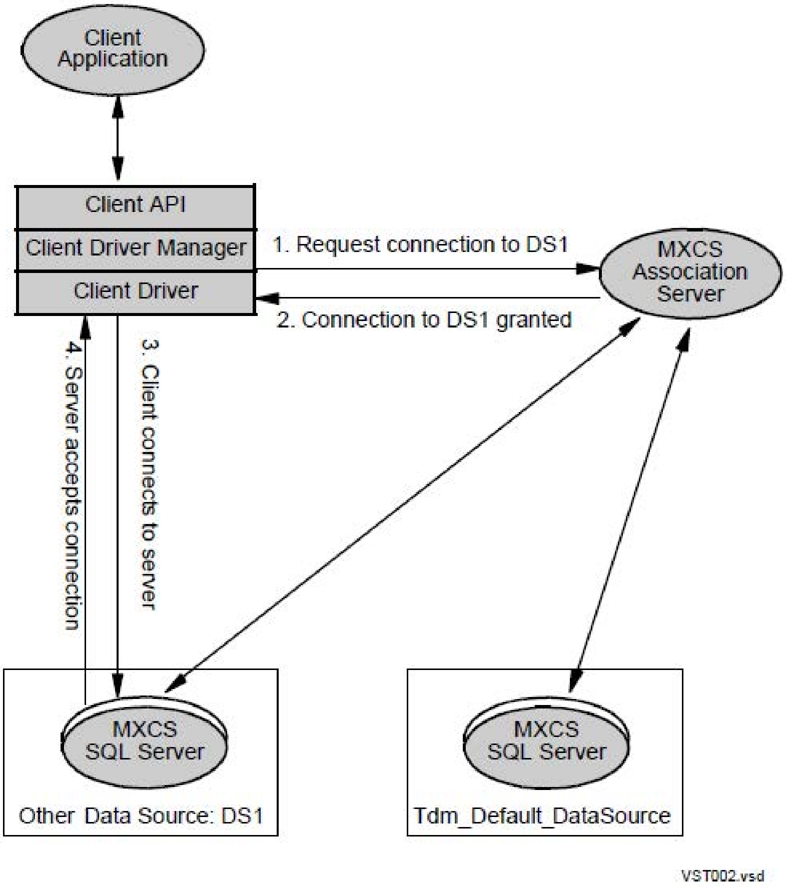
   NOTE: If a client tries to connect to a server data source that is stopped, MXCS rejects the connection.

3. The client application sends a connection request to the MXCS SQL server that was assigned by the association server. If the user information is not valid, the connection is rejected.

   NOTE: Starting with SQL/MX Release 3.2 or later, connecting to a data source takes less time.

4. When the client application disconnects, the MXCS SQL server notifies the association server that it is available for connections from other clients and returns to the server data source pool.

Figure 2 Connection Data Flow



Connection Mapping

MXCS maps the client data source name in an incoming connection request to the server data source name (names are case sensitive). If the name matches but the server data source is not started, the connection request is rejected. If the server data source name does not exist, the association server routes the connection to the default data source. If the default data source is not started, the connection request is rejected.

After a data source name is assigned, the association server uses any MXCS server that was previously used by the same user ID. If no such server is found, the association server assigns the first available server to the connection and reinitializes the user context to the one defined in the configuration for the data source name on the MXCS SQL server.

Security

Access authorization

Authorization for access to NonStop SQL databases uses authenticated Guardian user names. The MXCS SQL server program (MXOSRVR) follows SQL/MX rules for accessing SQL/MX objects (and SQL/MP rules for SQL/MP objects), and as such the SQL objects are subject to the normal SQL rules that govern access to those objects. For example, to read from an MX table, the user must have SELECT privilege.

Each MXCS SQL process runs under a Guardian user name that is used to authorize access to SQL objects.

For more information about SQL/MX privileges, see the *SQL/MX Reference Manual*.

User Authentication

MXCS uses a connection model in which users request a connection to a named service, known as the data source. To complete the connection, the user provides a user name and a password, which are passed in a secure, private packet to the MXCS server.

Two formats for the username are valid:

`group.user` or `safeguard-aliasname`

Because many client applications use only one-part user identification (such as on Windows and UNIX systems), you might need to add Safeguard aliases and use the `safeguard-aliasname` format when connecting to MXCS. For more information about user authentication, see the *Safeguard User's Guide*.

Passwords

Safeguard supports multiple alias and password pairs for a single Guardian username. This can improve performance because object caching reuses the underlying username. You can set up an anonymous login capability by adding a username such as `guest` with no password, then map this username to a user group that has limited access to the SQL objects.

Password Expiration

You can enable the password expiration notification feature, which sends a warning message to the driver when a password is scheduled to expire in the next two days or a password has already expired but is still within the grace period.

To enable notification of password expiration:

1. Using NSM/web, set the SQL_ATTR_WARNING environment variable in the SET tab to 1. (The default value of SQL_ATTR_WARNING is 0 (disabled).)

   This setting directs SQLConnect, SQLBrowseConnect and SQLDriverConnect to return SQL_SUCCESS_WITH_INFO with one of these warnings:

   - [WARNING 8857] Password to Expire in *n* days, *n* hours, *n* minutes. (If the password expires in less than two days.)

     or

   - [WARNING 8837] CLI Authorization failure: Error 48 with status 11 was returned by Safecom. (If the password is expired and in the grace period.)

2. If the application uses SQLDriverConnect with the SQL_DRIVER_PROMPT attribute, the ODBC/MX driver prompts the user to change the password.

   The user has these options:

   - Cancel the password change and proceed with the current session.
   - Change the password by providing the old and new passwords. If an invalid password is entered, the connection is terminated.

Operation Access Controls

You must use an ID created under the super group when installing MXCS. Starting the association server requires that you are logged on with a user ID that has Guardian access to run the MXOAS file. Thereafter, a user ID with MXCS OPERATOR permission can enable and disable the MXCS service and manage MXCS from NSM/web or MXCI. To perform any operation in MXCS, you must have the correct access privileges to NonStop SQL catalogs, MXCS catalogs, and individual NonStop SQL objects. A user ID with OPERATOR permission is the only user ID that can manage MXCS.

Controlling who can manage the MXCS service

MXCS users have access permissions based on the security values for their Guardian user ID. PUBLIC has USER permission and the super ID has OPERATOR permission by default.

Permissions can be assigned and changed using NSM/web only by the super ID. All other users can view permissions but not assign or change them.

There are two permissions:

MXCS Permission	Access
USER	View status and configuration of MXCS services, data sources, and server. View permissions.
OPERATOR	Enable, disable, manage, and configure MXCS. View permissions.

Errors, Events, and Warnings

All exceptions that occur during the MXCS service are written to the Event Management Service (EMS) operator collector $0 or to the alternate EMS collector if you configured it when you started the MXCS service. You can also configure the level of detailed information sent to the collector, ranging from critical commands only to verbose execution (which includes all internal status changes).

Event and error messages are described in the *Operator Messages Manual*.

Traces

You can trace activity for MXCS SQL servers.

Data collected on the server side:

- Calls to entry functions and input values received
- Calls to exit functions and output values returned

For detailed information about Server traces, see "Tracing Server Activity" (page 46).

Resource Statistics

The MXCS Resource Statistics facility collects statistics for use in estimating resource usage and predicting resource cost. When statistics collection is turned on, collection starts when the client connects to MXCS Server; thereafter, compilation and execution of SQL statements are recorded. Data is recorded until the end of a user session, or until you stop statistics collecting.

If the MXCS service is stopped, or if you stop a data source that has statistics collecting started, the statistics options are retained. Therefore, when the data source is restarted, statistics collection automatically resumes.

MXCS collects two categories of statistics:

- Session statistics are typically used if the data is used for charge-back. Session statistics report:

  Connection information

  Session summary

- Statement collection is typically used for monitoring query performance or to identify usage patterns. Statement statistics report data about the following SQL statements executed during statistics collection.

  SQLExecDirect

  SQLPrepare

  SQLExecute

  SQLFetch

See the MXCS online help that is available on NSM/web for detailed descriptions of the statistics collected.

For information on configuring statistics collection, see "Configuring Statistics Collection" (page 40).

MXCS Features

MXCS supports the following features:

- Row size limit is increased to align with maximum block size 32768 for MX tables.
- A clustering key length up to 2048 bytes is allowed for range and hash partitioned MX tables.
- SQL/MX connectivity supports the extended NUMERIC precision data type. The precision of this data type is extended up to 128 digits for signed and unsigned values. The extended NUMERIC precision data type is used to hold temporary values while computing the result of arithmetic expressions.
- Parameterized queries with BIGNUM data type.

Limitation of MXCS

The SQL/MX Release 2.x version of connectivity drivers are not compatible with the SQL/MX Release 3.x version of the connectivity server. If you try to connect a SQL/MX Release 2.x version of the connectivity driver to a SQL/MX Release 3.x version of the connectivity server, the behavior is unpredictable. SQL/MX Release 3.0 base drivers are not compatible with SQL/MX Release 3.1 servers.

Unsupported NonStop ODBC Server Features

MXCS does not support the following NonStop ODBC Server features:

- TDS messages (for DBLib clients)
- T-SQL
- User catalog customization (supported in ANSI)
- Pass-through commands
- Named pipes and IPX/SPX network protocols

Migrating From a NonStop ODBC Server Environment

You must migrate your NonStop ODBC Server applications if you plan to take advantage of the compiler and executor features of SQL/MX Release 3.x for your applications. By migrating your applications, you also take advantage of the ODBC 3.5 compliant API driver. Like SQL/MX (BSR X3.135-1992), ODBC/MX is based on the ISO/IEC 9075-3 standard and documented by Microsoft in the *ODBC 3.5 Programmer's Guide*.

For more information on the necessary application and configuration modifications for migration in both the server and the client environments, see "Migrating Applications from NonStop ODBC Server" (page 56).

2 Installing, Starting, and Uninstalling MXCS

The chapter discusses the following topics:

Components of the MX connectivity service

To fully install all components of connectivity service, you must install the following:

- MXCS server components on a NonStop system

  NOTE: For systems running on J06.05 and later J-series RVUs or H06.16 and later Hseries RVUs, the MXCS server minor version is 3.0 and the MXCS server major version is 3. MXCS will support older versions of connectivity clients.

- To run NSM/web:
  - A workstation with Internet Explorer 6.0 or later and Java Runtime Environment 1.5 or later.
  - NSM/web server components and an installed web server on the NonStop system.
- ODBC/MX client interface or other connectivity API, such as JDBC, on any Windows machine.
- MXCS Configuration and Metadata Tables

The `InstallSqlmx` script creates the following MXCS configuration tables in the system catalog under MXCS_SCHEMA:

NAME2ID
ASSOC2DS
ENVIRONMENTVALUES
DATASOURCES
RESOURCEPOLICIES

For more information about these tables, see the*HP NonStop SQL/MX Release 3.2.1 Reference Manual*.

Installation Requirements

Before MXCS is installed and configured, you must verify that the following are valid and present in your environment, in addition to the installation requirements specified for SQL/MX:

Requirement	Description
Operating system	The server components must be installed on a NonStop system running using an operating system version J06.15 and subsequent J-series RVUs or H06.26 or subsequent H-series RVUs.
OSS	Open System Services (OSS) must be installed and configured. This is necessary because the MXCS SQL servers and the SQL/MX compiler and executors run in an OSS environment.

Requirement	Description
	In addition, the root fileset must be already added and mounted, as required by SQL/MX. For more information, see the *Open System Services Installation Guide*.
Safeguard (optional)	Safeguard must be installed if you want to preserve the NonStop ODBC Server user id names or use name aliases. This product maps the Guardian user IDs to Safeguard alias names.
SQL/MP	To use SQL/MP tables, SQL/MP must be installed and initialized. For more information, see the *SQL/MP Installation and Management Guide*.
SQL/MX	To use SQL/MX, SQL/MX 3.x must be installed and initialized.
TCP/IP	TCP/IP must be installed; it is required by MXCS. For more information, see the *TCP/IP Configuration and Management Guide*.
TFDS	TFDS must be installed; it is required by MXCS for reporting exception errors and program errors from which the system cannot recover. For more information, see the *Tandem Failure Data System (TFDS) Manual*.

Step 1: Install the MXCS Server Components

Use DSM/SCM to install the MXCS server components from the Software Update Tape (SUT). This installs the server components and the ODBC/MX Client installation program in the ZMXODBC installation subvolume. You can copy the installed files to other locations.

Step 2: Configure for Statistics Collection and Server Trace (optional)

If you want the capability to collect resource statistics or trace server activity, you must configure and start an alternate EMS collector(s) before the MXCS service is started. If you do not configure and start an alternate collector, statistics or server trace messages go to the default collector, $0. If you specify an alternate collector, but it is not running, an error message is sent to the default collector. See “Configuring Statistics Collection” (page 40) and “Configuring Server Tracing” (page 46).

Step 3: Start the MXCS Service

The default location of the MXCS software is `$SYSTEM.ZMXODBC`. Alternatively, you can install the software in other subvolumes.

To start the MXCS service:

1. Go to the installation subvolume.
2. Start an MXCS service in one of the following ways:
   - By configuring it as a Kernel managed persistent process. For more information, see “Configuring MXCS as a Persistent Process” (page 19).
   - Manually, from a TACL session. For more information, see “Starting an MXCS Service manually” (page 20).

Configuring MXCS as a Persistent Process

You can configure and start MXCS as a persistent process, which means that whenever your NonStop server is coldloaded or manually restarted, MXCS will automatically restart.

The following example commands configure MXCS as a persistent process, and then start it:

```
TACL> scf
1-> assume process $zzkrn
2-> add #mxoas, cpu firstof(01,00), AutoRestart 10, hometerm
```

```
$zhome, name $mxo, StartMode
2-> application
2-> startupmsg “-pn 35000”, program $system.zmxodbc.mxoas
3-> start #mxoas
4-> status #mxoas
```

The above example indicates that the process `$mxo` starts from file

`$system.zmxodbc.mxoas` and uses the port number 35000.

Your installation might require different parameters or to configure a different set of CPUs. For more information, see the *SCF Manual for the Kernel Subsystem*.

Starting an MXCS Service manually

NOTE: Starting with H06.27 or J06.16 RVU, you can start MXOAS in Secure Sockets Layer (SSL) Encryption mode. SSL and its successor Transport Layer Security (TLS) are cryptographic protocols that provide communication security over the network. JDBC T4 driver uses SSL for communicating with the ODBC/MX server. Enabling NonStop SSL may degrade the performance.

To start the MXCS service, execute the MXOAS command at the TACL prompt to start the association server:

NOTE: Before starting MXCS, set the following parameters if the application uses Java 1.6:

```
PARAM CIPHERSUITES 0.4
PARAM MINVERSION 3.0
```

```
MXOAS /NAME proc-name, NOWAIT, TERM terminal-name,
   CPU cpu-number/ [-TCP tcp-process] [-EMS collector-name]
   [-PN port-number] [-PR port-range] [-I] [-E detail-level]
   [-RMC alternate collector-name]
   [-TMC alternate collector-name ]
  [ -SSL [-SSLPWD password] [-SSLTRACE trace-level] [-CACERT filename[, filename, filename]] [-SRVCERT filename] [-SRVKEY
 filename] ]
```

NAME *as-name*

specifies the process name of the MXCS association server.

TERM *terminal-name*

specifies the name of a valid terminal to be used as the home terminal for all MXCS processes. Specify a terminal that is always available, for example, $ZHOME.

If a home terminal is not specified, or it is a dynamic telnet window, the default is the terminal used to start the MXOAS process. If that terminal is a dynamic telnet window and it closes, MXCS stops accepting new connections because the window is valid only for current server connections.

NOTE: If the home terminal closes, the MXCS service cannot create new servers or process new connection requests.

CPU *cpu-number*

specifies the primary CPU in which the association server is to run.

-TCP *tcp-process*

specifies the Guardian name of the TCP/IP process. If *`TCPIP^process^name`* is defined before starting MXOAS, that name is used. If *`TCPIP^process^name`* is not defined, the default name is $ZTC0.

-EMS *collector-name*

specifies the name of the EMS collector to be used for event messages generated by MXCS; the default is $0.

`-PN` *`port number`*

specifies the starting port number (in decimal) for connections to the MXCS association server; the default is 18650. This port number is used as the single point of contact for all ODBC applications connecting to that IP address; internally MXCS manages the range of port numbers defined by -PR to ensure that each server has a separate port within the port range.

`-PR` *`port-range`*

specifies the number of ports that can be used simultaneously for connections to the MXCS association server (including the starting port number). The minimum requirement is 3 port numbers; the default is 250. The port range specified must not conflict with port numbers used by other applications.

`-I`

specifies the interactive startup mode for the association server. When this mode is specified, the association server is created in the STOPPED state and you must use NSM/web or the interactive MXCI interface to enable it. The default is the non-interactive state.

`-E` *`detail-level`*

0 Minimum, critical events only

1 Informational events

2 Internal state transition events

The default value is 0.

⚠ **WARNING!** Setting the `-E` value greater than zero can result in overflowing EMS log.

`-RMC` *`alternate-collector-name`*

specifies the name of the alternate EMS collector to be used for resource statistics generated by MXCS. If option [-RMC alternate-collector-name] is not provided, MXCS Server Trace sends trace messages to the primary EMS collector. You must start the alternate collector before starting MXCS. If resource statistics is enabled but the alternate collector is not started, an error message is logged to the default collector ($0). See “Configuring Statistics Collection” (page 40).

`-TMC` *`alternate-collector-name`*

specifies the name of the alternate EMS collector to be used for server tracing generated by MXCS. If option [-TMC alternate-collector-name] is not provided, MXCS Server Trace sends trace messages to the primary EMS collector. You must start the alternate collector before starting MXCS. If server tracing is enabled but the alternate collector is not started, an error message is logged to the default collector ($0). See “Configuring Server Tracing” (page 46).

`-SSL`

indicates that MXOAS must be started in SSL Encryption mode. If the SSL initialization fails, this option is ignored, and MXOAS is started without SSL encryption (pass through mode). You can see the `SSL Mode Activated` message in the EMS log if MXOAS starts in SSL Encryption mode.

`-SSLPWD` *`password`*

specifies the password for the encrypted server key. This option is mandatory when starting MXOAS in SSL Encryption mode. `password` value is not encrypted.

`-SSLTRACE` *`trace-level`*

indicates that SSL calls must be traced. The `trace-level` specifies the granularity for tracing the SSL calls. Value range is 0–9999.

`-CACERT` *`filename[,filename,filename]`*

specifies the CA certificates for signing. The value is a list of comma separated filenames. The first file on the list must contain a certificate for signing the given server certificate. Subsequent files must contain certificates that sign the previous certificate in the list. `filename` must be the absolute Guardian file path.

`-SRVCERT` *`filename`*

specifies the server certificate.

`-SRVKEY` *`filename`*

specifies the private key.

NOTE: The certificates must be in PKCS#8 format, with DER encoding. Encrypt the server key with a password. The certificates must have file code 800.

The following example command starts MXOAS in SSL Encryption mode:

```
run mxoas /name $SSLR, nowait/ -pn 33005 -ssl -ssltrace 50 -sslpwd 123456
-CACERT $DATA05.CERTS.CACERT2,$DATA05.CERTS.CACERT1 -SRVCERT $DATA05.CERTS.SERVCERT
-SRVKEY $DATA05.CERTS.SERVKEY
```

Additional Information

- After the MXCS service is started, you use NSM/web or MXCI to enable the MXCS service.
- Only the parameters described here are valid. The IN, OUT, stdin, stdout, or stderr parameters are not recognized by MXCS.
- If you specify an invalid value for any parameter, the default value is used (if available). If the parameter has no default value, MXOAS terminates with an error.
- When the MXCS service is started, it then starts the other MXCS server components, as specified in the configuration. When the server components are started and available, the MXCS service is ready to accept incoming connections.
- When the MXCS service is started, if the same option is given multiple times, the value of the last occurrence is taken into consideration.
- Check for port range configuration settings if the application response is slow, or new connection requests fail with connection errors. Check for the EMS message "020007" on the server side, the following is a message sample:

```
2-09-24 01:37:27 \ODBM.$MXZTI      TANDEM.ODBCMX.G06     020007 A NonStop
                                    Process Service error No more ports available to spawn new
                                    MXOSRVR,there were a total of 18
                                    available/connected MXOSRVR, 2 more ports
                                    used by ODBC/MX and 0 more MXOSRVR were
                                    spawned. This is a total of 20 ports.
                                    Current port range is 20, Details for
                                    each DS follows :DS
                                    (TDM_Default_DataSource) had MXOSRVR
                                    Spawned = 0  and Connencted/Available
                                    MXOSRVR = 1  ,DS (mmxodbc01) had MXOSRVR
                                    Spawned = 0  and Connencted/Available
                                    MXOSRVR = 17  , has occurred.
                                    Event Type: 1
```

```
Component Name: ODBC/MX Association Server
Object Reference:
```

The port range limits the number of `mxosrvrs` that can be spawned from all started datasources. This also includes two ports reserved for MXOAS and MXOCFG. HP recommends you to maintain the default configuration of 250.

List of Installed Files

Server Files

File Name	ISV	Description
MXOAS	ZMXODBC	Association server
MXOCFG	ZMXODBC	Configuration server
MXOSRVR	ZMXODBC	MXCS SQL server
TDMODBC	ZMXODBC	File that contains the Windows client 32-bit ODBC components for communicating with the server.
NSODBC64	ZMXODBC	File that contains the Windows client 64-bit ODBC components for communicating with the server.
ODBCW32	ZMXODBC	File that contains the Windows client 32-bit Unicode ODBC components for communicating with the server.
ODBCW64	ZMXODBC	File that contains the Windows client 64-bit Unicode ODBC components for communicating with the server.
LODBCTAR	ZMXODBC	File that contains the Linux client 32-bit ODBC components for communicating with the server.
ODBC64HI	ZMXODBC	File that contains the HP-UX client 64-bit ODBC components for communicating with the server.
MXODSN	ZMXODBC	Template file that contains data source options for MXCS connectivity OSS ODBC/MX Driver for NonStop SQL/MX.
ZODBCDLL	ZMXODBC	File that contains the 32-bit OSS ODBC components to required communicate with the server.
YODBCDLL	ZMXODBC	File that contains the 64-bit OSS ODBC components required to communicate with the server.
ZCMDDLL	ZMXODBC	File that contains the components required to communicate with MXDM.
MXOMSG	SYSTEM	Error message file for the MXCS connectivity service
TDMNSM	ZMXNSSM	Placeholder file. **Note:** The NonStop Software Manager is unsupported. Use NSM/web to manage MXCS.
ZMXOTMPL	ZTEMPL	MXCS event message template. When SYSGEN is run, the event messages are linked together into a general TEMPLATE file in the subvolume specified for SYSGEN.

File Name	ISV	Description
SMXOTMPL	ZTEMPL	MXCS event messages.
T0611PAX	ZOSSUTL	SQL/MX and MXCS module files

Migrating from SQL/MX Release 2.3.4 to SQL/MX Release 3.2

NOTE: The same procedure is applicable when you migrate from SQL/MX 2.3.4 to 3.2.1.

The format of the CPU lists stored in the ODBC metadata, in SQL/MX Release 2.3.x is different from the format of the CPU lists stored in SQL/MX Release 3.2. The SQL/MX Release 2.3.x CPU list format is not recognized in SQL/MX Release 3.2. Starting with SQL/MX Release 3.2, `mxtool` includes the `fixmxcsmd` option. The `fixmxcsmd` option supports the conversion of the SQL/MX Release 2.3.x CPU list format to the SQL/MX Release 3.2 CPU list format in the `ENVIRONMENTVALUES MXCS` metadata table. To convert the SQL/MX Release 2.3.x CPU list format to the SQL/MX Release 3.2 CPU list format, you must run the `mxtool` command with the `fixmxcsmd` option.

To convert the SQL/MX Release 2.3.x CPU list format to the SQL/MX Release 3.2 CPU list format, you must run the `mxtool` command with the `fixmxcsmd` option after cold loading with SQL/MX Release 3.2 objects.

NOTE: The `fixmxcsmd` option also supports the conversion of the SQL/MX Release 3.2 CPU list format to the SQL/MX Release 2.3.x CPU list format in the `ENVIRONMENTVALUES MXCS` metadata table.

To convert the SQL/MX Release 3.2 CPU list format to the SQL/MX Release 2.3.x CPU list format, you must run the `mxtool` command with the `fixmxcsmd` option before cold loading with SQL/MX Release 2.3.x objects.

The following is the syntax:

```
mxtool FixMxcsmd -v <target> [-d]
```

Where `<target>` is the major SQL/MX release that corresponds to the required target format. The values allowed are 2 and 3.

The `-d` flag enables `mxtool` to display the source and target CPU list values without performing any conversion. Using the `-d` flag is optional.

NOTE: You must perform the `mxtool fixmxcsmd` operation only when MXCS is in the Stopped state.

Errors are generated under the following circumstances:

- If you assign a wrong value for `number`
- If a normal user executes the command (Only a SUPER ID user can convert the MXCS metadata.)

Warnings are generated under the following circumstances:

- If a qualifying row from `ENVIRONMENTVALUES` has an inconsistent value, the row is skipped, and a warning is issued.
- If a qualifying row from `ENVIRONMENTVALUES` has an ambiguous value, the row is converted according to the format specification, and a warning is issued.

If a given CPU list value is a number between 1 and 15, you cannot determine the format. For such CPU list values, a wrong target format specification generates appropriate error messages. For example, the CPU list value 10 can imply a SQL/MX Release 3.x format CPU list consisting of the CPU, 10 only, or it can imply a SQL/MX Release 2.x format CPU list with bits set for CPU 12 and 14.

Executing the following command results in the CPU list 12 and 14, which is wrong if the CPU list is in SQL/MX Release 3.2 format:

```
mxtool fixmxcsmd -v 3
```

However, executing the following command results in the CPU list value 32, which is wrong if the CPU list is in SQL/MX Release 2.3.x format:

```
mxtool fixmxcsmd -v 2
```

HP recommends that prior to converting the CPU list format, you test the option using the `-d` flag to view the conversions.

The following are examples:

```
/home/user: mxtool fixmxcsmd -v 2 -d

*** Display MXCS cpu list transformations to R2 format
Data source DS1, translate '0,1,2' -> '57344'
Data source YOUR_DS, translate '14,15' -> '3'
              ...

/home/user: mxtool fixmxcsmd -v 2

*** Perform MXCS cpu list transformations to R2 format
Data source DS1, translate '0,1,2' -> '57344'
Data source YOUR_DS, translate '14,15' -> '3'
              ...
```

The previous examples highlight the difference in output when you use the `-d` flag.

Creating the default catalog and schema for MXCS

After upgrading to SQL/MX Release 3.2.1, you must create the default catalog and schema for MXCS. The default catalog is `NONSTOP_SYSTEM_NSK`. The default schema is `PUBLIC_ACCESS_SCHEMA`.

To create the default catalog and schema, run the following script located at `/usr/tandem/sqlmx/bin`:

```
.\InstallMXCS -loc \$volume
```

Where,

`$volume` is the name of an audited, non-SMF DAM volume.

For example,

```
.\InstallMXCS -loc \$data01
```

NOTE: When you create a fresh installation of SQL/MX Release 3.2.1, the `InstallMXCS` script is executed automatically, and the default catalog and schema are created. You do not have to run the script.

The NonStop SQL/MX Database Manager (MXDM) requires the default catalog and schema to display table statistics.

For more information on why the default catalog and schema are required, see Viewing table statistics and their properties in the *HP NonStop MXDM User Guide for SQL/MX Release 3.2.1*.

Uninstalling MXCS from the Server

To uninstall MXCS components from the server, you must:

1. Stop the MXCS service.
2. Remove the object files from the installation subvolume.

3 Starting, Configuring, and Managing MXCS

For more information on the MXCS tasks, see the *SQL/MX Connectivity Service Administrative Command Reference* and the NSM/web online help. For information on installing, uninstalling, and managing NSM/web, see the *HP NonStop SQL/MX Release 3.2.1 Installation and Upgrade Guide*. Starting with SQL/MX Release 3.2, a new tool, called NonStop SQL/MX Database Manager (MXDM), is available. The new tool enables you to view and manage MXCS objects. For information on viewing and managing the MXCS objects, such as MXCS Services, data sources, and MXCS user permissions, see the *HP NonStop MXDM User Guide for SQL/MX Release 3.2.1*.

Managing MXCS Services

Only user ID's with MXCS OPERATOR permission can perform MXCS management tasks. See "Operation Access Controls" (page 15) for information about MXCS permissions.

Logging on

To make changes to the MXCS configuration, you must be logged on with a user ID that has OPERATOR permission. User IDs with USER permission can only view the configuration. Use NSM/web to view the user permissions assigned.

Displaying the MXCS Service Status

1. Using NSM/web, click MXCS Services.
2. Click the status tab.
3. For more information, click **Help**.

Enabling the MXCS Service

To start the MXCS service, execute the MXOAS command at the TACL prompt (see "Starting an MXCS Service manually" (page 20)). Once the service is started, you can disable and enable it using NSM/web.

Requirement: you must use a user ID with OPERATOR permission to perform management tasks.

1. Using NSM/web, expand MXCS Services.
2. Right-click the service name you want to start.
3. Click Start.

For more information about a screen, click **Help**.

Disabling the MXCS Service

Disabling the service stops all servers and changes the state of the association server to stopped; it does not shutdown the service. You can still perform configuration tasks when the service is disabled.

Requirement: you must use a user ID with OPERATOR permission to perform management tasks.

1. Using NSM/web, expand MXCS Services.
2. Right-click the name of the service you want to stop.
3. Click **Stop**.
4. In the dialog box, enter a reason for disabling the service.
5. Click the appropriate stop option.

For more information about a screen, click **Help**.

Displaying Server Status

1. Using NSM/web, expand MXCS Services.
2. Click the name of the service.
3. Click the MXCS Server Status tab.

For more information about a screen, click **Help**.

Stopping Servers

You must use a user ID that has OPERATOR permission to perform management tasks.

1. Using NSM/web, expand MXCS Services.
2. Click the appropriate MXCS service name. Status tabs for that service display in the right pane.
3. Click the MXCS Server Status tab.
4. Click the process name of the MXCS server you want to stop.

   To select more than one MXCS SQL server, hold down the Shift or Ctrl key and click anywhere in the row for each server name.
5. Click **Stop Immediately**.

For more information about a screen, click **Help**.

Managing Multiple MXCS Services

You can manage multiple MXCS services using one instance of NSM/web.

Managing MXCS Server Data Sources

Use NSM/web to manage the MXCS data sources in the server.

`TDM_Default_DataSource` is the default server-side data source. It starts up by default. If you create a client-side data source, a matching data source must be available on the server. If a matching data source is not available on the server, client applications use the default server-side data source, `TDM_Default_DataSource`, to connect to the database. In this case, default values cannot be changed using NSM/web.

For information on managing data sources using MACL, see the *SQL/MX Connectivity Service Administrative Command Reference manual*.

Managing data sources

Only user ID’s with MXCS OPERATOR permission can perform MXCS management tasks. See “Operation Access Controls” (page 15) for information about MXCS permissions.

Displaying Data Source Status

1. Using NSM/web, click MXCS Services.
2. Click the Data Source Status tab.

For more information about a screen, click **Help**.

Creating and Configuring Server Data Sources

You can modify the configuration of a data source while the data source is running.

You must use a user ID that has OPERATOR permission to perform management tasks.

1. Using NSM/web, right-click Data Source Configurations.
2. Click **Create New Data Source** to start the data source wizard.
3. Fill out each screen displayed by the wizard.

If you have questions about a screen, click **Help**.

Configuring MXCS XA Data Source

For XA transactions, do not use the default data source. You must create one or more data sources with the following configurations and use the JDBC serverDataSource property to refer to them.

1. Use NSM/Web or MXCI to create a data source that has to participate in distributed (XA) transactions. Consider that the XADS data source is created.
2. Log on as the super ID to the NonStop server where you want to configure the server side data source.
3. Start MXCI.
4. At the MXCI prompt, run the following commands:

   a. `set schema NONSTOP_SQLMX_`*`machine_name`*`.MXCS_SCHEMA;`

      where, *`machine_name`* is the NonStop operating system server.

      For example: For a machine `\HPIDMR4`, the command is:

      ```
      set schema NONSTOP_SQLMX_HPIDMR4.MXCS_SCHEMA;
      ```

   b. `select * from name2id;`

      This command is used to find the ID associated with your data source, XADS, created in Step 1.

      For example, the output of the command is:

      ```
      OBJ_ID          OBJ_TYPE        OBJ_NAME                LAST_UPDATED

      1               1               TDM_Default_ODBCService ?

      2               2               TDM_Default_DataSource  ?

      3               2               XADS                    2007-01-24

                                                              11:05:26.846

                                                              563
      ```

The XADS data source is assigned the value '3' in the above table.

c. Set the following Control Query Default (CQD) to access and modify the `ENVIRONMENTVALUES` table:

   1) `Control Query Default ODBC_METADATA_PROCESS 'ON';`

   2) Insert the XA definition of the XADS data source that is created in Step 1.

      Insert the following parameters in the `ENVIRONMENTVALUES` table:

```
ENV_ID: what we got from step a> and b>
VARIABLE_SEQUENCE: 0
VARIABLE_TYPE: 7
VARIABLE_VALUE: -XABRK XABROKER PROCESS NAME -DM DOMAIN
NAME -PATH
OSS PATH OF THE XABROKER FILE [-CONN number of
connections -TRANS
number of transactions -LVL debug level]
```

      `VARIABLE_VALUE` allows the command-line arguments to start the XA Broker product. The default values are:

```
-XABRK XABROKER PROCESS NAME = $XABR
-DM DOMAIN NAME = BRK_TEST_XABR
-PATH OSS PATH OF THE XABROKER FILE =
/G/system/zxabrkr
-CONN number of connections = Default 100, Max 1000
-TRANS number of transactions = Default 10, Max 100
-LVL debug level = Default 0. Valid levels: 0
(Error), 1 (Error and Info), 2 (Error, Info and Debug)
```

      For example:

```
Insert into ENVIRONMENTVALUES (ENV_ID,
VARIABLE_SEQUENCE,VARIABLE_TYPE,VARIABLE_VALUE)
values(3,0,7, '-XABRK
XABR -DM BRK_TEST_XABR -PATH /G/system/ZXABRKR -CONN 100
-TRANS 10 -LVL 0');
```

For more information on XA transactions, see the *XACI Command Reference Manual*.

Reconfiguring Server Data Sources

You must use a user ID that has OPERATOR permission to perform management tasks.

1. Using NSM/web, click Data Source Configurations to expand the list of data sources.
2. Click on the data source you want to reconfigure.
3. Click the tab associated with the change you want to make.

To Change This	Click This Tab
Control Query Default	Control Query Defaults
Controlled Tables	Controlled Tables
DEFINE statement	Defines
SET statement	Sets
Resource management policy	Resource Management Policy
Resource statistics	Resource Statistics
Number of servers, Performance, and Start options	General Configuration
CPU utilization and process priority	Process Management

If you have questions about a screen, click **Help**.

Using Case-Sensitive table names

To handle the system metadata table names that are case-sensitive, use NSM/web to set the SQL_ATTR_METADATA_ID environment variable for the data source.

1. Using NSM/web, expand Data Source Configurations.
2. Click a data source name.
3. Click the Sets tab.
4. Click + to add a new Set.
5. Enter SQL_ATTR_METADATA_ID for **Name** and SQL_TRUE for **Value**.
6. Click **Ok** when finished.

The default setting is SQL_FALSE and is overridden by the setting from the ODBC application when calling SQLSetConnectAttr for the same attribute.

Reconfiguring Character Sets

The default character set is determined at SQL/MX installation. ODBC applications may reconfigure the character set for a data source by prefixing character literals in an SQL statement. The text to be inserted into a Unicode column is prefixed with _UCS2 or N, with no spaces between the prefix and the character literal. For example:

```
N'Unicode_text'
Or
_UCS2'Unicode_text'
```

`N` is used if the target column is defined as NCHAR and the default character set is UCS2.

Starting MXCS Data Sources

Data sources configured with the automatic startup option are started when the MXCS service is started. However, data sources configured with the manual startup option must be started by using NSM/web and MACL. You must use a user ID with OPERATOR permission to start a data source.

1. Using NSM/web, expand MXCS Services and click a service name.
2. Click the Data Source Status tab.
3. Select the data source to be started.
4. Click **Start**.

MXCS starts the data source and its associated MXCS servers.

The MXCS association server must be started before you can start a data source. To determine if the association server is started, in the MXCS Services list, click the service name and check the Status tab to see if the state of the association server is STARTED.

For information on using the MACL, see the *SQL/MX Connectivity Service Administrative Command Reference manual*.

Stopping MXCS Data Sources

You must use a user ID that has OPERATOR permission to perform management tasks.

1. Using NSM/web, click MXCS Services. A list of services displays in the left pane.
2. Click a service name.
3. Click the Data Source Status tab. The list of data sources appears in the right pane.
4. Click a data source name.
5. Click **Stop**.
6. In the dialog box, enter a reason for stopping the data source.
7. Click the appropriate stop option.

For more information about a screen, click **Help**.

If the data source you stop is tracing server activity, tracing is automatically enabled when you restart the data source. However, if you stop the Association Server from TACL, tracing information is lost. In this case, use NSM/web to enable tracing again.

If the data source you stop has statistics collection started, the statistics options are retained. When you restart the data source statistics collection will automatically resume.

For information on using the MACL, see the *SQL/MX Connectivity Service Administrative Command Reference manual*.

Deleting MXCS Data Sources

You must use a user ID that has OPERATOR permission to perform management tasks.

A data source must be stopped in all association servers in the Non Stop server before you can delete it.

1. Using NSM/web, expand Data Source Configurations.
2. Right-click the data source name you want to delete.
3. From the shortcut menu, click **Delete**.

For more information about a screen, click **Help**.

NOTE: You cannot delete the MXCS default data source (TDM_Default_DataSource).

Using MXCS with SQL/MP tables

By default, MXCS uses logical table names (ANSI names); the same logical name can be used by a SQL/MX application running on the NonStop server and a client application using MXCS.

You can use NSM/web to create an SQL/MP alias, or enter the SQL/MX command CREATE SQLMP ALIAS in MXCI, to add the logical name mapping for each SQL/MP table that you want to be available to a client application.

Resource Management

You can assign a resource management policy to a data source by adding the policy using NSM/web. You can also delete the policy that is currently in effect.

Adding or Removing a Resource Management Policy

Requirement: You must have OPERATOR permission to perform management tasks.

To add or remove a resource management policy for a data source

1. Using NSM/web, Expand Data Source Configurations.
2. Click the name of the data source for which you want to add or remove the policy.

   Click the Resource Management tab.
3. Enter the new resource management policy or select an existing policy, then click the appropriate command button.

Changes take effect immediately for an available server, and for a connected server when it goes to available state.

Table 1 Attributes and Values for Resource Management Policies

Attribute	The name of a resource management policy governing attribute that limits the execution of user queries from a client application, based on cost. Enter only one resource management policy for each attribute name.
Action	The action to be taken by MXCS when the limit is exceeded. An event message is sent to the event log in all cases.

Table 1 Attributes and Values for Resource Management Policies *(continued)*

	LOG - The user query continues execution. LOG_WITH_INFO - The user query returns a warning message to the client. The query execution continues. STOP - The user query returns an error message to the client. The query execution stops.
Limit	The maximum value for the attribute. This value is any positive number. The maximum is the value of LARGEINT SQL data type.

Resource Statistics

You can collect resource statistics (server statistics) for a particular data source.

MXCS Server can collect these statistics:

- Session Statistics

  Connection information
  Session summary

- Statement Statistics

  - SQL statement — shows the text of the SQL query for individual SQL statements executed but does not include other statistics

  - Statistical data about individual statements

    SQLExecDirect

    SQLExecute

    SQLPrepare

    SQLFetch

  You may choose any combination of options. For example, choosing SQL statement plus one or more of the individual statements returns the SQL statement that was executed and the statistics for that statement. Choosing SQL Statement alone returns the text of the SQL query for individual SQL statements. See "Example resource statistics output for three selected options" (page 37).

  Use NSM/web or MXCI to select these options for collecting resource statistics. See "Starting Statistics Collection" (page 40).

Statement Statistics - Statements and Values

SQL Statement

Statement statistics are written to the alternate EMS collector at the time a Prepare statement is received.

Table 2 Statement Statistics - SQL Statement

Statistic	Value
Session ID	Unique session ID generated by the MXCS SQL server at the time the connection is established.
Message Attribute	STATEMENT:SQLStatement (the Statistics option that was selected.)
Sequence Number	Message sequence number generated by MXCS. The number begins with 0 (zero) and is incremented by 1 with each new record.

Table 2 Statement Statistics - SQL Statement *(continued)*

Statistic	Value
StatementID	Unique statement ID generated by the MXCS SQL server at the time the Prepare command is received.
SQL Text	Text of the SQL statement. This field can be distributed in multiple messages.

SQLExecute

SQLExecute statistics are written to the alternate EMS collector at the time an Execute statement is received.

Table 3 Statement Statistics - SQLExecute

Statistic	Value
Session ID	Unique session ID generated by the MXCS SQL server at the time the connection is established.
Message Attribute	STATEMENT:SQLExecute (the Statistics option that was selected.)
Sequence Number	Message sequence number generated by MXCS. The number begins with 0 (zero) and is incremented by 1 with each new record.
StatementID	Unique statement ID generated by the MXCS SQL server at the time the Prepare command is received.
ODBCElapsedTime	Total MXCS SQL server real time, in microseconds, from the time the execute started to the time the results are sent back.
ODBCExecutionTime	Total MXCS SQL server CPU process time, in microseconds, from the time the execute started to the time the response is sent back to the client with the results error code.
NumberOfRows (ins/upd/del)	Number of rows that were inserted, updated, or deleted.
ErrorCode	Execute error code, if any. A zero (0) indicates that the execute was successful.
RowsAccessed	Cumulative number of rows accessed.
RowsRetrieved	Cumulative number of rows retrieved.
DiscReads	Cumulative number of disk reads.
MsgsToDisc	Number of messages sent to the disk process.
MsgsBytesToDisc	Size, in bytes, of the total number of messages sent to the disk process.
LockWaits	Number of lock waits (this is the concurrency cost).
LockEscalation	Cumulative number of times record locks are escalated to file locks.
SQLValues	Input parameter value.

SQLExecDirect

SQLExecDirect statistics are written to the alternate EMS collector at the time an Execute statement is received.

Table 4 Statement Statistics - SQLExecDirect

Statistic	Value
Session ID	Unique session ID generated by the MXCS SQL server at the time the connection is established.
Message Attribute	STATEMENT:SQLExecDirect (Statistics option that was selected)
Sequence Number	Message sequence number generated by MXCS. The number begins with 0 (zero) and is incremented by 1 with each new record.
StatementID	Unique statement ID generated by the MXCS SQL server at the time the Prepare command is received.
EstimatedCost	Estimated query cost returned by the SQL compiler.
StatementType	Valid types are: Insert, Delete, Update, Select
ODBCElapsedTime	Total MXCS SQL server real time, in microseconds, from the time the execdirect started to the time the results are sent back.
ODBCExecutionTime	Total MXCS SQL server CPU process time, in microseconds, from the time the execdirect started to the time the results are sent back.
NumberOfRows (ins/upd/del)	Number of rows that were inserted, updated, or deleted.
ErrorCode	Execdirect error code, if any. A 0 (zero) indicates that the execdirect was successful.
RowsAccessed	Cumulative number of rows accessed.
RowsRetrieved	Cumulative number of rows retrieved.
DiscReads	Cumulative number of disk reads.
MsgsToDisc	Number of messages sent to the disk process.
MsgsBytesToDisc	Size, in bytes, of the total number of messages sent to the disk process.
LockWaits	Number of lock waits (this is the concurrency cost).
LockEscalation	Cumulative number of times record locks are escalated to file locks.

SQLPrepare

SQLPrepare statistics are written to the alternate EMS collector at the time a Prepare statement is received.

Table 5 Statement Statistics - SQLPrepare

Statistic	Value
Session ID	Unique session ID generated by the MXCS SQL server at the time the connection is established.
Message Attribute	STATEMENT:SQLPrepare (Statistics option that was selected)
Sequence Number	Message sequence number generated by MXCS. The number begins with 0 (zero) and is incremented by 1 with each new record.
StatementID	Unique statement ID generated by the MXCS SQL server at the time the Prepare command is received.

Table 5 Statement Statistics - SQLPrepare *(continued)*

Statistic	Value
EstimatedCost	Estimated query cost.
StatementType	Valid types are: Insert, Delete, Update, Select.
SQLCompileTime	Total MXCS SQL server real time, in microseconds, from the time the prepare starts to the time it returns to the client with the compile results.
ErrorCode	Prepare error code, if any. A zero (0) indicates that the prepare was successful.

SQLFetch

SQLFetch statistics are written to the alternate EMS collector at the time a statement is closed.

Table 6 Statement Statistics - SQLFetch

Statistic	Value
Session ID	Unique session ID generated by the MXCS SQL server at the time the connection is established.
Message Attribute	STATEMENT:SQLFetch/SQLClose (Statistics option that was selected).
Sequence Number	Message sequence number generated by MXCS. The number begins with 0 (zero) and is incremented by 1 with each new record.
StatementID	Unique statement ID generated by the MXCS SQL server at the time the Prepare command is received.
RowsAccessed	Cumulative number of rows accessed.
RowsRetrieved	Cumulative number of rows retrieved.
DiscReads	Cumulative number of disk reads.
MsgsToDisc	Number of messages sent to the disk process.
MsgsBytesToDisc	Size, in bytes, of the total number of messages sent to the disk process.
LockWaits	Number of lock waits (this is the concurrency cost).
LockEscalation	Cumulative number of times record locks are escalated to file locks.
ODBCElapsedTime	Total MXCS SQL server real time, in microseconds, from the time the fetch was prepared to the time it was closed.
ODBCExecutionTime	Total MXCS SQL server CPU process time, in microseconds, from the time the fetch was prepared to the time it was closed.
TotalExecutes	Total number of executes performed on the statement.

Session Statistics - Summaries and Values

Session statistics are written to the alternate EMS collector at the time a connection is made (see Table 7 (page 36)) and when the session is terminated (see Table 8 (page 36)).

Table 7 Session Statistics - Connection Information

Statistic	Value
Session ID	Unique session ID generated by the MXCS SQL server at the time the connection is established.
Message Attribute	SESSION:ConnectionInformation (Statistics option that was selected) SESSION:ConnectionInformation Authentication failed (for unsuccessful connections due to incorrect password) SESSION:ConnectionInformation Authentication failed; password expired (for unsuccessful connections due to password expiry).
Sequence Number	Message sequence number generated by MXCS. The number begins with 0 (zero) and is incremented by 1 with each new record.
Component	Name of the MXCS component to which the statistics apply.
UserName	User name logged into the session (the logical name used at the time of connection).
UserID	User ID that is logged into the session.
ClientID	Client workstation TCP/IP name or the NetBios/network name.
Application ID	Client application name or the main window caption.
DataSource	Client data source name used at the time of connection.
NodeName	Name of the NonStop server used by the data source.
CpuPin	NonStop Kernel CPU number in which the MXCS server is running, followed by the process ID of the server.

Table 8 Session Statistics - Session Summary

Statistic	Value
Session ID	Unique session ID generated by the MXCS SQL server at the time the connection is established.
Message Attribute	SESSION:Summary (Statistics option that was selected).
Sequence Number	Message sequence number generated by MXCS. The number begins with zero (0) and is incremented by 1 with each new record.
StartTime	Time when the connection request was received.
EndTime	Time when the session was terminated due to an application request or a server idle timeout.
StartPriority	Execution priority of the MXCS process at the time the connection request is received.
TotalOdbcExecutionTime	Total MXCS SQL server CPU process time, in microseconds, from the time the connection started to the time the connection closed. Does not include SQL/MX compiler time.
TotalOdbcElapsedTime	Total MXCS SQL server real time, in microseconds, from the time the connection started to the time the connection closed.
TotalInsertStmtsExecuted	Total number of SQL INSERT statements sent by the ODBC/MX driver to the MXCS SQL server. Does not include internally generated SQL statements.

Table 8 Session Statistics - Session Summary *(continued)*

Statistic	Value
TotalDeleteStmtsExecuted	Total number of SQL DELETE statements sent by the ODBC/MX driver to the MXCS SQL server. Does not include internally generated SQL statements.
TotalUpdateStmtsExecuted	Total number of SQL UPDATE statements sent by the ODBC/MX driver to the MXCS SQL server. Does not include internally generated SQL statements.
TotalSelectStmtsExecuted	Total number of SQL SELECT statements sent by the ODBC/MX driver to the MXCS SQL server. Does not include internally generated SQL statements.
TotalCatalogStmts	Total number of SQL Catalog statements sent by the ODBC/MX driver to the MXCS SQL server.
TotalPrepares	Total number of SQLPrepare calls.
TotalExecutes	Total number of SQLExecute calls.
TotalFetches	Total number of SQLFetch calls.
TotalCloses	Total number of SQLClose calls.
TotalExecDirects	Total number of SQLExecDirect calls.
TotalErrors	Total number of SQL statement execution errors detected by the MXCS SQL server.
TotalWarnings	Total number of SQL statement execution warnings detected by the MXCS SQL server.

Example resource statistics output for three selected options

The following example shows the resource statistics collected for the options SQL Statement, SQLPrepare, and Session Summary. A separate EMS message is returned for each option selected.

```
01-10-03 09:27:50 \ODBC.$Z1782 TANDEM.ODBCMX.G10 021035 ODBC/MX
Statistics:
 Session ID: $SS:11:$Z1782:20011039272
 Message Attribute:
 STATEMENT:SQLStatement
 Sequence Number: 0
 Message Info:
 StatementId:200110392749 SqlText:select * from employee

01-10-03 09:27:50 \ODBC.$Z1782 TANDEM.ODBCMX.G10 021035 ODBC/MX
Statistics:
 Session ID: $SS:11:$Z1782:200110392729
 Message Attribute:
 STATEMENT:SQLPrepare
 Sequence Number: 0
 Message Info:
 StatementId:200110392749
 EstimatedCost:0
 StatementType:select
 SQLCompileTime:1106627 ErrorCode:0
01-10-03 09:27:29 \ODBC Z1782 TANDEM.ODBCMX.G10 021035 ODBC/MX
Statistics:
 Session ID: $SS:11:$Z1782:200110392729
 Message Attribute: SESSION:SessionSummary
 Sequence Number: 0
 Message Info:
 StartTime:2001/10/3 9:27:29.45
 EndTime:2001/10/3 9:47:36.44
 StartPriority:168 TotalOdbcExecutionTime:332432
 TotalOdbcElapsedTime :2347517
 TotalInsertStmtsExecuted:2
 TotalDeleteStmtsExecuted:0
 TotalUpdateStmtsExecuted:0
 TotalSelectStmtsExecuted:2
 TotalCatalogStmts:0 TotalPrepares:2 TotalExecutes:2
```

```
TotalFetches:4 TotalCloses:3 TotalExecDirects:3
TotalErrors:1 TotalWarnings:2
```

- SQL Statement — The first message contains information for the SQL Statement option and shows that an SQL SELECT statement was prepared for execution. Note that no detailed statistics are given for that statement - just the query itself, and the time of execution.
- SQLPrepare — The second message contains information for the SQLPrepare option and shows that an SQL SELECT statement was prepared at 9:27. This time matches the time for the SELECT statement returned by SQL Statement. This entry also contains the SQLPrepare statistics for that SELECT statement. Note that to obtain complete statistics information for SQL statements, you must select the SQL Statement option and the option for each specific statement for which you want statistics.
- Session Summary — The third message contains information for the Session Summary option. This message shows statistics for the entire user session, from client connect to client disconnect.

For detailed definitions of statistics that are collected, see the online help for NSM/web.

For information on configuring statistics collection, see “Configuring Statistics Collection” (page 40).

Param Caching

Param Caching is an enhancement of resource statistics which allows logging SQL query parameter value as part of resource statistics information during the start of the SQL Execute operation.

Configuring SQLValues

To configure SQLValues:

1. Set SQLExecuteStat in Data Source to ON.
2. In Data Source of type set, add the `SELECT_INPUT_PARAM_LOG` environment. The `SELECT_INPUT_PARAM_LOG` takes either TRUE or FALSE as values. By default, this attribute is not enabled.

   ```
   CS>add evar $XYZ."TDM_Default_DataSource".SELECT_INPUT_PARAM_LOG, type set,
   value 'TRUE';
   -- ADD EVAR $XYZ.TDM_Default_DataSource.SELECT_INPUT_PARAM_LOG Successful
   CS>add evar $XYZ."TDM_Default_DataSource".SELECT_INPUT_PARAM_LOG, type set,
   value 'FALSE';
   -- ADD EVAR $XYZ.TDM_Default_DataSource.SELECT_INPUT_PARAM_LOG Successful
   ```
3. To set the `SELECT_INPUT_PARAM_LOG` attribute using NSM/web, set the `SELECT_INPUT_PARAM_LOG` attribute to TRUE in NSM/web SET tab for data source.

Logging Format for SQL Values

The logging format for two arbitrary SQL Query parameter values is:

`SQLValues:{TXABCDEF}`

Where:

T: 'Z', one byte indicates that the value is a sequence of Tag-Length-Value (TLV)

X: A total length of three bytes encoded in Base64

TLV Sequence:

A: One byte indicates the data tag for the first parameter value

B: Two bytes for the data length encoded in Base64

C: Data value for the first parameter value

D: One byte indicates the data tag for the second parameter value

E: Two bytes for data length encoded in Base64

F: Data value for the second parameter value

The list of available data tags are:

A: Printable ISO88591

B: Base64 encoded ISO88591

C: Base64 encoded UCS2

D: Date

I: Interval

N: Numeric

K: Base64 encoded Kanji

S: Base64 encoded KSC560

a: Truncated printable ISO88591

b: Truncated Base64 encoded ISO88591

c: Truncated Base64 encoded UCS2

k: Truncated Base64 encoded Kanji

s: Truncated Base64 encoded KSC5601

Sample EMS Log for Select Query

The following is a sample EMS log for SQL SELECT Query:

```
select * from ALLDataType where c1=?,c2=?,c3=?,c4=?,c5=?,c6=?
,c7=?,c8=?,and c9=?
```

All "?" values are in the order of C1 through C9.

```
'abcdefghi’,_ucs2'aß0123456’,a123456',_ucs2'678aß’,interval '01
12:23:34.987654'day to second,date '1988-07-25’,2333
,123456789,987654.12345
```

Table DDL

You can create a table of 9 columns, with each column representing different data or charset as follows:

```
create table ALLDataType(C1 char(10),C2 char(10) character set
ucs2,C3 varchar(10),C4 varchar(50) character set ucs2,C5
interval day to second,C6 date,C7 int,C8 bigint ,C9
numeric(11,5))
```

```
11-06-26 05:43:51 \ODBC.$Y2N9 TANDEM.ODBCMX.G06 021035 MXCS
Statistics:
 Session ID: \ODBC:9:$Y2N9:201162654347
 Message Attribute:
 STATEMENT:QueryParamValues
 Sequence Number: 0/0
 Message Info:
 StatementId:SQL_CUR_MFC00000001
 QueryId:MXID01223001061212175852220031727
 000000000111SUPER.SUPER00_20_SQL_CUR_MFC00
 000001 StatementType:SQL_SELECT_NON_UNIQUE
 SQLValues:{ZACOAAKabcdefghi
 CAcAGEA3wAwADEAMgAzADQANQA2ACA=AAHa123456
  CAQADYANwA4AGEA3w==IAT01 12:23:34.987654
 DAK1988-07-25NAE2333NAJ123456789NAM987654s,
 . 12345}
```

Limitations of SQLValues

SQLValues have the following limitations:

- Logs parameter value only for select query type.
- Truncates the parameter value on exceeding 800 bytes.

- Logs empty string `SQLValues:{}` when the total length of the query parameter message exceeds 3.4 KB. The message length of three bytes reserves a byte for data tag and other two bytes for parameter length. The total length of the query parameter message is calculated for 'n' number of parameter value with ni parameter length as:

  If (n1 + 3) + (n2 + 3) + (n3 + 3) + (n4 + 3)... (Nn + 3) > 3400 , then `SQLValues:{}` is logged.

  ```
  11-08-19 17:16:31 \LLSW2.$Y0NN TANDEM.ODBCMX.G06 020007 A NonStop
  Process Service error
  Total Length of the Input Param values 3415 exceeded MAX length 3400 has
  occurred.
  Event Type: 1
  Component Name: ODBC/MX Server
  Object Reference: TCP:$ZTC0/33338:NonStopODBC
  ```

- Logs an empty string `SQLValues:{}` for Insert, Update, or Delete query type.

Reading SQLValue

You can read the SQL query parameter value in either of the ways:

- Start EMSDIST TYPE PRINTING with indent 0 and redirect the output to a file. A sample java program is available as a downloadable sample, which you can decode to obtain the actual data as per the data tag. HP recommends to start the MXCS service on a different collector other than $0, and you can use the same collector to start EMSDIST.
- Start EMSDIST TYPE CONSUMER and read it using SPI. For more information, see *SPI Programmer Manual*, and *EMS manual*.

Collecting Server Statistics

Configuring Statistics Collection

You must use a user ID that has OPERATOR permission to perform management tasks.

1. Configure an alternate collector, using the EMSACOLL command. For details on the EMSACOLL command, see the *EMS Manual*.
2. Start the alternate collector before starting the MX connectivity service.
3. Use the MXOAS command to start the MX connectivity service, and pass the name of the alternate collector to the server in the -RMC parameter. See “Starting Statistics Collection” (page 40).

Starting Statistics Collection

You must use a user ID that has OPERATOR permission to perform management tasks.

1. Using NSM/web, expand Data Source Configurations.
2. Click a data source name.
3. Click on the Resource Statistics tab.
4. Check one or more statistics options.
5. Click **Apply**.

   Statistics collection begins for all available servers and connected servers.

For more information about a screen, click **Help**.

Stopping Statistics Collection

You must use a user ID that has OPERATOR permission to perform management tasks.

1. Using NSM/web, click Data Source Configurations.
2. Click on a data source name.
3. Click the Resource Statistics tab.

4. Clear statistics options.
5. Click **Apply**.

For more information about a screen, click **Help**.

Viewing Statistics

The EMS alternate collector collects the MXCS Server statistics and returns them in event message 21035. See message “Resource Statistics Event Message Format” (page 41) for the message format.

Each statistics option returns a separate message for every qualified statement, for connection information, and for session summary. See “Example resource statistics output for three selected options” (page 37) for an example of statistics returned.

See the MXCS online help, which is available on NSM/web, for descriptions of the fields that are collected. To easily access a description of each statistics field while viewing the collected statistics, keep an NSM/web window open as you look at the statistics messages.You can view statistics messages by running a printing distributor using the EMSDIST command, or by using ViewPoint or WebViewPoint. See the *EMS Manual, the ViewPoint Manual*, or the *Web ViewPoint User’s Guide* for details.

Resource Statistics Event Message Format

Message 21035

Parameter	Attribute
MXCS Statistics	Session ID: *session-id* Message Attribute: *message-attribute* Sequence Number:*sequence-number* Message Info:*statistics-information*

message-attribute	{ SESSION:ConnectionInformation }
	{ SESSION: SessionSummary }
	{ STATEMENT: SQLStatement }
	{ STATEMENT: SQLExecDirect }
	{ STATEMENT: SQLExecute }
	{ STATEMENT: SQLPrepare }
	{ STATEMENT: SQLFetch }

session-id	Unique session ID generated by the SQL server at the time the connection is established.
message-attribute	Name of the statistics option you selected.
sequence–number	Messages too large for one message template are divided into smaller messages. Sequence number indicates which part of the message you are viewing. The sequence number is 0 for the first or only part of a message, and 1 for additional parts, if any.
statistics-information	The statistics collected for the session or statement options selected.

21035

See above.

Cause. A resource statistics informational message.

Effect. None.

Recovery. Informational message only; no corrective action is needed.

Session Statistics: Connection Information

Session statistics are written to the alternate EMS collector at the time the connection is made (listed in the order written in the event message):

Statistic	Description
Session ID	Unique session ID generated by the MXCS server at the time the connection is established.
Message Attribute	SESSION:ConnectionInformation (the statistics option that was selected).
Sequence Number	Message sequence number generated by MXCS. The number begins with zero (0) and is incremented by 1 with each new record.
Component	Name of the MXCS component to which the statistics apply.
UserName	User name logged into the session (the logical name used at the time of connection).
UserId	User ID that is logged into the session.
ClientId	Client workstation TCP/IP name or the NetBios/network name.
ApplicationId	Client application name or the main window caption.
DataSource	Client data source name used at the time of connection.
NodeName	Name of the NonStop platform used by the data source.
CpuPin	Operating system CPU number in which the MXCS server is running, followed by the process ID of the server.

Session Statistics: Session Summary Information

Session statistics are written to the alternate EMS collector when the session is terminated (listed in the order written in the event message):

Statistic	Description
Session ID	Unique session ID generated by the MXCS SQL server at the time the connection is established.
Message Attribute	SESSION:Summary (the statistics option that was selected).
Sequence Number	Message sequence number generated by MXCS. The number begins with zero (0) and is incremented by 1 with each new record.
StartTime	Time when the connection request was received.
EndTime	Time when the session was terminated due to an application request or a server idle timeout.
StartPriority	Execution priority of the MXCS process at the time the connection request is received.
TotalOdbcExecution Time	Total MXCS SQL server CPU process time, in microseconds, from the time the connection started to the time the connection closed; does not include NonStop SQL compiler time.
TotalOdbcElapsedTime	Total MXCS SQL server real time, in microseconds, from the time the connection started to the time the connection closed.
TotalInsertStmtsEx ecuted	Total number of SQL INSERT statements sent by the HP ODBC driver to the MXCS SQL server; does not include internally generated SQL statements.

Statistic	Description
TotalDeleteStmtsEx ecuted	Total number of SQL DELETE statements sent by the HP ODBC driver to the MXCS SQL server; does not include internally generated SQL statements.
TotalUpdateStmtsExecuted	Total number of SQL UPDATE statements sent by the HP ODBC driver to the MXCS SQL server; does not include internally generated SQL statements.
TotalSelectStmtsExecuted	Total number of SQL SELECT statements sent by the HP ODBC driver to the MXCS SQL server; does not include internally generated SQL statements.
TotalCatalogStmts	Total number of SQL Catalog statements sent by the HP ODBC driver to the MXCS SQL server.
TotalPrepares	Total number of SQLPrepare calls.
TotalExecutes	Total number of SQLExecute calls.
TotalFetches	Total number of SQLFetch calls.
TotalCloses	Total number of SQLClose calls.
TotalExecDirects	Total number of SQLExecDirect calls.
TotalErrors	Total number of SQL statement execution errors detected by the MXCS SQL server.
TotalWarnings	Total number of SQL statement execution warnings detected by the MXCS SQL server.

Statement Statistics: SQL Statement

Statement statistics are written to the alternate EMS collector at the time a PREPARE statement is received (listed in the order written in the event message):

Statistic	Description
Session ID	Unique session ID generated by the MXCS SQL server at the time the connection is established.
Message Attribute	STATEMENT:SQLStatement (the statistics option that was selected).
Sequence Number	Message sequence number generated by MXCS. The number begins with zero (0) and is incremented by 1 with each new record.
StatementID	Unique statement ID generated by the MXCS SQL server at the time the PREPARE statement is received.
SQL Text	Text of the SQL statement. This field can be distributed in multiple messages.

Statement Statistics: SQLExecute

SQLExecute statistics are written to the alternate EMS collector at the time an EXECUTE statement is received (listed in the order written in the event message):

Statistic	Description
Session ID	Unique session ID generated by the MXCS server at the time the connection is established.
Message Attribute	STATEMENT:SQLExecute (Statistics option that was selected.)
Sequence Number	Message sequence number generated by MXCS. The number begins with zero (0) and is incremented by 1 with each new record.

Statistic	Description
StatementID	Unique statement ID generated by the MXCS SQL server at the time the PREPARE statement is received.
ODBCElapsedTime	Total MXCS SQL server real time, in microseconds, from the time the execute started to the time the results are sent back.
ODBCExecutionTime	Total MXCS SQL server CPU process time, in microseconds, from the time the execute started to the time the response is sent back to the client with the results error code.
NumberOfRows	(ins/upd/del) Number of rows that were inserted, updated, or deleted.
ErrorCode	Execute error code, if any. A zero (0) indicates that the execute was successful.
RowsAccessed	Cumulative number of rows accessed.
RowsRetrieved	Cumulative number of rows retrieved.
DiscReads	Cumulative number of disk reads.
MsgsToDisc	Number of messages sent to the disk process.
MsgsBytesToDisc	Size, in bytes, of the total number of messages sent to the disk process.
LockWaits	Number of lock waits (this is the concurrency cost).
LockEscalation	Cumulative number of times record locks are escalated to file locks.
SQLValues	Input parameter value.

Statement Statistics: SQLExecDirect

SQLExecDirect statistics are written to the alternate EMS collector at the time an EXECUTE statement is received (listed in the order written in the event message):

Statistic	Description
Session ID	Unique session ID generated by the MXCS SQL server at the time the connection is established.
Message Attribute	STATEMENT:SQLExecDirect (Statistics option that was selected.)
Sequence Number	Message sequence number generated by MXCS. The number begins with zero (0) and is incremented by 1 with each new record.
StatementID	Unique statement ID generated by the MXCS SQL server at the time the PREPARE statement is received.
EstimatedCost	Estimated query cost returned by the SQL compiler.
StatementType	Valid types are: Insert, Delete, Update, Select.
ODBCElapsedTime	Total MXCS SQL server real time, in microseconds, from the time the execdirect started to the time the results are sent back.
ODBCExecutionTime	Total MXCS SQL server CPU process time, in microseconds, from the time the execdirect started to the time the results are sent back.
NumberOfRows	(ins/upd/del) Number of rows that were inserted, updated, or deleted.
ErrorCode	Execdirect error code, if any. A zero (0) indicates that the execdirect was successful.
RowsAccessed	Cumulative number of rows accessed.

Statistic	Description
RowsRetrieved	Cumulative number of rows retrieved.
DiscReads	Cumulative number of disk reads.
MsgsToDisc	Number of messages sent to the disk process.
MsgsBytesToDisc	Size, in bytes, of the total number of messages sent to the disk process.
LockWaits	Number of lock waits (this is the concurrency cost).
LockEscalation	Cumulative number of times record locks are escalated to file locks.

Statement Statistics: SQLPrepare

SQLPrepare statistics are written to the alternate EMS collector at the time a PREPARE statement is received (listed in the order written in the event message):

Statistic	Description
Session ID	Unique session ID generated by the MXCS SQL server at the time the connection is established.
Message Attribute	STATEMENT:SQLPrepare (Statistics option that was selected.)
Sequence Number	Message sequence number generated by MXCS. The number begins with zero (0) and is incremented by 1 with each new record.
StatementID	Unique statement ID generated by the MXCS SQL server at the time the PREPARE statement is received.
EstimatedCost	Estimated query cost.
StatementType	Valid types are: Insert, Delete, Update, Select.
SQLCompileTime	Total MXCS SQL server real time, in microseconds, from the time the prepare started to the time it returns to the client with the compile results
ErrorCode	Prepare error code, if any. A zero (0) indicates that the prepare was successful.

Statement Statistics: SQLFetch

SQLFetch statistics are written to the alternate EMS collector at the time a statement is closed (listed in the order written in the event message):

Statistic	Description
Session ID	Unique session ID generated by the MXCS SQL server at the time the connection is established.
Message Attribute	STATEMENT:SQLFetch/SQLClose (Statistics option that was selected.)
Sequence Number	Message sequence number generated by MXCS The number begins with zero (0) and is incremented by 1 with each new record.
StatementID	Unique statement ID generated by the MXCS SQL server at the time the PREPARE statement is received.
RowsAccessed	Cumulative number of rows accessed.
RowsRetrieved	Cumulative number of rows retrieved.
DiscReads	Cumulative number of disk reads.
MsgsToDisc	Number of messages sent to the disk process.

Statistic	Description
MsgsBytesToDisc	Size, in bytes, of the total number of messages sent to the disk process.
LockWaits	Number of lock waits (this is the concurrency cost).
LockEscalation	Cumulative number of times record locks are escalated to file locks.
TotalOdbcElapsedTime	Total MXCS SQL server real time, in microseconds, from the time the fetch was prepared to the time it was closed.
TotalOdbcExecutionTime	Total MXCS SQL server CPU process time, in microseconds, from the time the fetch was prepared to the time it was closed.
TotalExecutes	Total number of executes performed on the statement.

Tracing Server Activity

You can trace MXCS activity for a data source in order to track error conditions and server function.

Server data collected:

Session ID

Function Name

Sequence Number

Function information

Every trace event generates a pair of messages. The first message is written on entry to a function, and contains the input parameters to the function. The second is written on exit, and contains the exit parameters for the function. See "Example Server Trace Log for the Connect Function" (page 47).

For detailed definitions of server trace fields that are collected, see the MXCS online help, which is available on NSM/web.

For information on configuring server trace, see "Configuring Server Tracing" (page 46).

NOTE: If you stop an association server from TACL, the Enable Trace option is not retained for a data source that is tracing server activity. When the association server is restarted, you must enable tracing again.

Configuring Server Tracing

You must use a user ID that has OPERATOR permission to perform management tasks.

1. Configure an alternate collector, using the EMSACOLL command. For details on the EMSACOLL command, see the *EMS Manual*.
2. Start the alternate collector before starting the ODBC/MX service.
3. Use the MXOAS command to start the ODBC/MX service, and pass the name of the alternate collector to the server in the -TMC parameter.

See "Starting Server Tracing" (page 46).

Starting Server Tracing

1. Do one of the following:
   a. Using NSM/web click MXCS Services.
   b. Expand MXCS Services and click a service name.
2. Click the Data Source Status tab.
3. Check the Tracing column for the data sources you want to trace.
4. When finished, click **Apply**.

   Tracing starts for all available servers and connected servers.

For more information about a screen, click **Help**.

Using NSM/web, you can see the enabled or disabled status of traces on the Data Sources Status tab.

Stopping Server Tracing

You must use a user ID that has OPERATOR permission to perform management tasks.

1. Do one of the following:
   a. Using NSM/web, click MXCS Services.
   b. Expand MXCS Services and click a service name.
2. Click the Data Source Status tab.
3. Clear the Tracing column for the data sources you want to stop tracing.
4. When finished, click **Apply**.

For more information about a screen, click **Help**.

Using NSM/web, you can see the enabled or disabled status of traces on the Data Sources Status tab.

Viewing Trace Information

The EMS alternate collector collects the ODBC/MX server trace information and stores it in EMS log, message 21034. For a list of data collected in a server trace, see "Server data collected:" (page 46). See message "Trace Event Message Format" (page 48) or the message format.

Example Server Trace Log for the Connect Function

The following example shows the trace messages returned for the connect function. The first message, regarding entering the connect function, shows the date and time of entry and contains the input parameters. The second message, regarding exiting the connect function, shows the exit date and time and contains the exit parameters.

```
01-09-25 15:31:00 \TEST.$Y8499 TANDEM.ODBCMX.G10 021034 ODBC/MX
         Trace
         Session ID: $YIJ:2:$Y8499:20019251531
         Function: EnterConnect
         Sequence Number: 0
         UserDescType:2 DomainName:
         UserName:super.super Datasource:TEST
         Catalog:CAT Schema:SCH AccessMode:0
         AutoCommit:1 IsolationLevel:2
         RowSetSize:0 Component0:7 Major0:3
         Minor0:51 Build0:65583 Component1:8
         Major1:3 Minor1:0 Build1:0
          DialogueId:1827180711

01-09-25 15:31:00 \TEST.$Y8499 TANDEM.ODBCMX.G10 021034 ODBC/MX
         Trace:
         Session ID: $YIJ:2:$Y8499:20019251531
         Function: ExitConnect
         Sequence Number: 0
         ExceptionNr:0 ExceptionDetail:1879411928
         Component0:260 Major0:2 Minor0:2
         Build0:919 Component1:3 Major1:1 Minor1:0
          Build1:0 Catalog:CAT Schema:SCH
```

Trace Event Message Format

Message 21034

```
MXCS Trace:

Session ID: session-id
Function: Enterfunction
Sequence Number: sequence-number
function-input-parameters
OR
Session ID: session-id
Function: Exitfunction
Sequence Number: sequence-number
function-exit-parameters
```

`session-id`

unique session ID generated by the MXCS SQL server at the time the connection is established.

`function`

The MXCS SQL server function that is being traced.

`sequence number`

Messages too large for one message template are divided into smaller messages. *`Sequence number`* indicates which part of the message you are viewing. The sequence number is 0 for the first or only part of a message, and 1 for the additional parts, if any.

`function-input-parameters`

list of input parameters to the function.

`function-exit-parameters`

list of output parameters from the function.

Cause. A trace informational message.

Effect. None.

Informational message only, no corrective action is needed.

Module File Caching (MFC)

Prior to H06.18/J06.07 RVU, the ODBC/MX driver did not cache the NonStop SQL/MX prepared statements. As a result, applications recompiled the same SQL/MX statements, causing higher host utilization, which could lead to worse perceived performance of the ODBC client application.

The Module File Caching (MFC) feature shares the SQL/MX prepared statement plans among the SQL/MX database connections. It helps in reducing the SQL/MX compilation time during the steady state of the JDBC or ODBC application, thereby reducing resource consumption.

The MFC feature is supported by the JDBC Type 4 driver, JDBC Type 2 driver, and ODBC/MX driver for Windows.

Starting with SQL/MX Release 3.2 or later, MFC supports parameterized queries with BIGNUM data type.

For information on the JDBC Type 4 driver support for MFC, see the *HP NonStop JDBC Type 4 Driver Programmer's Reference for SQL/MX Release 3.2.1*.

For information on the ODBC/MX driver support for MFC, see the *HP NonStop ODBC/MX Driver for Windows for SQL/MX Release 3.2.1*.

Design of MFC

MFC shares static module files created from dynamic queries using SQL/MX tools, across MXOSRVR or SQL/MX connections.

MFC uses disk files on the NonStop Open System Services file system for storing the compiled SQL/MX prepared statement plans. When the application uses prepareexecute or execdirect API, MFC checks for an existing plan that matches the query. If a plan exists, it is loaded from the disk thereby eliminating the requirement of SQL compilation of the query.

If a plan for an incoming PREPARE statement for a query is not found on the disk, a plan is created as follows:

1. The incoming query is compiled as any other SQL/MX prepared statement.
2. After successful completion of the SQL/MX prepared statement, a plan is generated and stored in the directory specified in the `compiled_module_location` attribute of SQL/MX. Synchronization issues are handled using temporary lock files.

Use an MFC management script for maintaining the compiled module plans on the NonStop server. The MFC management script is described under “Managing MFC” (page 51).

In releases prior to SQL/MX Release 3.2.1, the `MXOSRVR` waits until the SQL query compilation generates the MFC file, thus blocking the client. Starting with SQL/MX Release 3.2.1, this limitation is overcome by running the MFC file generation process in the background. As a result of this optimization, MFC module files may fail to generate for few SQL queries if any DDL query such as DROP or ALTER is executed immediately after compilation of the SQL query.

Configuring MFC from the MXCS mode

To configure the module file caching feature using the MXCS subsystem:

1. Enter the MXCS subsystem through MXCI:

   ```
   mode mxcs;
   ```

2. Configure a data source for the user-specified MXOAS service:

   ```
   add ds $MXOAS.”MCBS_datasource”;
   ```

   NOTE: For instructions on adding a data source, see the *SQL/MX Connectivity Service Administrative Command Reference* manual.

3. Stop the data source:

   ```
   stop ds “MCBS_datasource”, reason ‘test’;
   ```

   NOTE: Perform step 3 only if the configured data source is in the started state.

4. Add the `statement_module_caching` attribute to the data source to enable the module file caching feature:

   ```
   add evar $MXOAS.”MCBS_datasource”.statement_module_caching, type
   set, value ‘true’;
   ```

   This attribute has only two values, `true` and `false`. By default, this attribute is not enabled.

5. Add the `compiled_module_location` attribute to the data source:

   ```
   add evar $MXOAS.”MCBS_datasource”.compiled_module_location, type
   set, value ‘default’;
   ```

   ```
   add evar $MXOAS.”MCBS_datasource”.compiled_module_location, type
   set, value ‘/usr/MCBS_modules’;
   ```

If the 'default' value is specified then, compiled_module_location directory that is set would be /usr/tandem/sqlmx/TEMPMODULES

The directory location specified in the compiled_module_location attribute will include the following files:

a. The compiled module plans.

b. Temporary files such as *.lst,*.sql*.mdf ; all these files are retained in the directory to enable easy troubleshooting.

NOTE:

- If you assign the compiled_module_location evar to a value other than the default, such as /usr/temp, ensure that the directory is created before assigning it.
- The created module files are available at the default location or the assigned compiled_module_location. While upgrading from one version of SQL/MX to the other, the created module files from the older versions must be deleted and recreated for the following scenarios:
  - Upgrading from SQL/MX Release 2.3.x to 3.0 and later releases.
  - Upgrading from SQL/MX Release 3.0 to 3.1 and later releases.
  - Upgrading from SQL/MX Release 3.1 to 3.2.

  However, while upgrading from SQL/MX Release 3.2 to 3.2.1, you can reuse the module files.

6. To verify that the values are assigned, enter the following command:

   ```
   CS>info evar <data source>.*, type set;
   ```

   The following output appears indicating that the values are assigned:

   ```
   -- INFO EVAR \DMR15.$VCD9.<data source>.* Report
   Sets:
   SET STATEMENT_MODULE_CACHING TRUE
   SET COMPILED_MODULE_LOCATION default
   ```

7. By default, MFC is disabled for SQLExecDirect. However, when you enable MFC for an application as described from Step 1 to Step 6 of this procedure, MFC for SQLExecDirect with parameterized queries is enabled and module files are created. To enable MFC for SQLExecDirect non parameterized queries, configure the following setting:

   ```
   add evar $R32.RAZ32_MFC.STATEMENT_EXECDIRECT_CACHING, type set, value 'true';
   ```

8. Start the data source:

   ```
   start ds $MXOAS.”MCBS_datasource”;
   ```

Configuring MFC using NSM/Web

To configure MFC using NSM/Web:

1. Log on to NSM/Web.
2. Click **MXCS**.
3. Under **Data Source Configurations**, click the data source name for which you want to enable MFC.
4. Click **Sets**.
5. To set the statement module caching:
   a. Click **+**.
   b. In the **Name** field, enter **statement_module_caching**.
   c. In the **Value** field, enter **true**.

    d. Click **OK**.

6. To add the compiled module location:

    a. Click **+**.

    b. In the **Name** field, enter **`compiled_module_location`**.

    c. In the **Value** field, enter **`default`** or any other location that you want.

    d. Click **OK**.

7. Click **Apply**.
8. Stop and start the data source.

Managing MFC

Every application must have its own module file directory so that all the modules for an application are grouped together.

OSS commands can be used to remove temporary files and modules when required.

The MFC module files can be managed using an OSS script called `mgscript`.

The `mgscript` is located at the Guardian location, where the MXOSRVR objects are present. The following options can be used with the MFC management script:

`-A`

This option deletes all MFC related files from the specified directory. For example:

```
>>mgscript -A <Dir>
```

```
Dir : This must be the directory mentioned
in the compiled_module_location
```

`-T`

This option deletes all MFC related binary files from the `specified` directory related to a specific table. For example:

```
>>mgscript –T <Dir> <table_name>
<catalog_name.schema_name>
```

`-T` must have three options: Directory, table name, and catalog.schema name.

When you alter the table using MXCI, specify the correct table name. Do not use wild card characters.

`-C`

This option deletes all MFC related files from the `specified` directory related to a specific catalog and schema. For example:

```
>>mgscript –C <Dir> <catalog_name>
<schema_name>
```

`-C` must have three options: Directory, catalog name, and schema name.

Benefits of MFC

JDBC and ODBC applications that use complex queries result in lower processor utilization, lower memory consumption, and better response time. For more information on best practices and benefits of MFC, refer the technical white paper on Module File Cache for NonStop SQL/MX.

Limitations of MFC

- It does not handle session-specific SQL/MX Control Query Defaults (CQD) and SQL/MX Control Query Shape (CQS).
- For light weight queries, MFC performs only marginally better than SQL/MX compile.
- MFC is not supported for rowset operations.
- Starting with SQL/MX 3.1, MFC supports only INSERT, SELECT, UPDATE, or DELETE queries.

Troubleshooting MFC

The troubleshooting of MFC includes:

.err Files

The *.err files are intermediate files available at a location specified in COMPILED_MODULE_LOCATION EVAR, which reports MXCMP errors, if any, generated during module file creation.

If MXCMP passes successfully, the *.err files are removed automatically after the module file is created. Otherwise, the *.err files are stored at the specified location for further analysis of the reported failure.

.lock Files

The *.lock files are generated for every query that pass through the MFC module file creation process. These files are also used for synchronizing, so that different connections do not re-create the same module file. The *.lock files are deleted from the specified compiled_module_location directory after the binary module is successfully created.

The *.lock files are not deleted for the queries that cannot create module files. For example, ODBC application queries using scalar functions, such as SUM or AVG or ABS cannot create module files.

.mdf Files

The .mdf temporary files are generated during preprocessing. These .mdf files are retained for easier support and troubleshooting.

The preprocessor creates an SQL module definition file in the <compiled_module_location> location that contains the embedded SQL statements.

NOTE: For more information on module definition files, see the *HP NonStop SQL/MX Programming Manual for C and COBOL*.

.lst Files

The .lst files are the output of the SQL/MX C/C++ preprocessor. In rare cases, these files might contain compiler errors.

.sql Files

The .sql files contain the generated source code of the statement. This file is read by the mxsqlc preprocessor to produce the .mdf file. The .sql files can be used to tune an SQL statement by adding statement-specific Control Query Default and/or shapes.

In releases prior to SQL/MX Release 3.2, MFC did not support parameterized queries with BIGNUM data type.

The following is a sample .sql file used prior to SQL/MX Release 3.2:

```
# include <stdio.h>

EXEC SQL MODULE "MFCABS1031CAT"."SCH"."B4FABF7702B9046B0E8F94FAC72E7027" NAMES A
RE ISO88591;
int main ()
{
 EXEC SQL BEGIN DECLARE SECTION;
SQL_TYPE_NULL param0;
short param0_ind = -1;
EXEC SQL END DECLARE SECTION  ;
EXEC SQL CONTROL QUERY DEFAULT    ODBC_PROCESS                   'TRUE';
EXEC SQL CONTROL QUERY DEFAULT    JDBC_PROCESS                   'TRUE';
EXEC SQL CONTROL QUERY DEFAULT    GENERATE_EXPLAIN               'ON';
EXEC SQL CONTROL QUERY DEFAULT    NAMETYPE                       'ANSI';
EXEC SQL CONTROL QUERY DEFAULT    INFER_CHARSET                  'ON';
EXEC SQL CONTROL QUERY DEFAULT    RECOMPILATION_WARNINGS         'ON';
EXEC SQL DECLARE SCHEMA CAT.SCH ;
EXEC SQL SET SCHEMA CAT.SCH ;
EXEC SQL DECLARE MXSTMT01 CURSOR FOR SELECT * FROM MYTAB6 WHERE INFO = :param0 ;
return 0;
 }
```

Starting with SQL/MX Release 3.2, MFC supports parameterized queries with BIGNUM data type.

The following is a sample `.sql` file:

```
EXEC SQL MODULE "MFCABS1031CAT"."SCH"."B4FABF7702B9046B0E8F94FAC72E7027" NAMES A
RE ISO88591;
int main ()
{
 EXEC SQL BEGIN DECLARE SECTION;
NUMERIC(56,20) param0;
short param0_ind = -1;
EXEC SQL END DECLARE SECTION  ;
EXEC SQL CONTROL QUERY DEFAULT    ODBC_PROCESS                   'TRUE';
EXEC SQL CONTROL QUERY DEFAULT    JDBC_PROCESS                   'TRUE';
EXEC SQL CONTROL QUERY DEFAULT    GENERATE_EXPLAIN               'ON';
EXEC SQL CONTROL QUERY DEFAULT    NAMETYPE                       'ANSI';
EXEC SQL CONTROL QUERY DEFAULT    INFER_CHARSET                  'ON';
EXEC SQL CONTROL QUERY DEFAULT    RECOMPILATION_WARNINGS         'ON';
EXEC SQL DECLARE SCHEMA CAT.SCH ;
EXEC SQL SET SCHEMA CAT.SCH ;
EXEC SQL DECLARE MXSTMT01 CURSOR FOR SELECT * FROM MYTAB6 WHERE INFO = :param0 ;
return 0;
 }
```

Disk Activity

The MFC access plans are stored in the disk OSH location as specified in the attribute `compiled_module_location`. For more information on setting this attribute, see "Configuring MFC from the MXCS mode" (page 49). Assume that the value for the attribute is `/usr/mymodules`. If specifying this location attribute causes an increase in the processor utilization of disks, then to overcome this problem, use the fileset setting for that directory. It is beneficial to have OSS caching enabled on data volumes. To enable fileset and OSS caching, see "Enable Fileset and OSS Caching" (page 53).

NOTE: If the table is altered using data definition language (DDL) or if it is dropped, run the management script (`mgscript`) to delete module files associated with that table or catalog. For information on management script, see "Managing MFC" (page 51).

Enable Fileset and OSS Caching

To add a fileset pointing to a value in the `compiled_module_location` directory, complete the following steps:

1. At a TACL prompt, enter:

   SCF

   and then enter:

```
assume $zpmon
```

2. At an SCF prompt, enter the SCF command:

   ```
   add server <SERVER_NAME>, cpu 1, backupcpu 2
   ```

3. Add a fileset:

   ```
   add fileset mxc1,nameserver #(server name),catalog $(volume other
   than $oss), pool mxc1pool, mntpoint "/usr/mymodules"
   ```

   NOTE: Copy the 'POOL' template file located in the $SYSTEM.ZXOSSMON volume and modify the default volume setting to point to the location of the newly defined fileset mountpoint.

4. Verify the status of the fileset:

   ```
   info fileset mxc1,detail
   ```

5. Start the fileset:

   ```
   start fileset mxc1
   ```

To enable OSS caching, perform the following steps:

1. At a TACL prompt, enter:

   ```
   SCF
   ```

   and then enter:

   ```
   assume $zpmon
   ```

2. At an SCF prompt, enter the following SCF command to stop all filesets on your system:

   ```
   STOP FILESET $ZPMON.*
   ```

   This command begins with the last fileset mounted and stops the filesets in the reverse order in which they were last started.

3. Stop the OSS Monitor process:

   If the OSS Monitor is running as a standard process, enter the STOP command at a TACL prompt:

   ```
   STOP $ZPMON
   ```

   If the OSS Monitor is running as a persistent process, enter the ABORT command at an SCF prompt:

   ```
   ABORT PROCESS $ZZKRN.#ZPMON
   ```

4. At the SCF prompt, enter the following set of commands for each disk volume in the fileset:

   ```
   STOP DISK diskname
   ALTER DISK diskname, OSSCACHING ON
   START DISK diskname
   ```

   `diskname` is the name of a disk volume that contains OSS files.

5. Restart the OSS Monitor as a normal or persistent process with the appropriate command.
6. Restart the OSS file system by entering the SCF command:

   ```
   START FILESET $ZPMON.filesetname
   ```

   where, `filesetname` is the name of each fileset that contains OSS files, beginning with the root and specified in the order in which mount points occur.

Known Issues

Scenario 1

MFC plans become obsolete when the base table is altered or dropped. The following sequence of operations illustrates the issue.

Operation	Expected Result	Actual Result	Remarks
`Create table testing( info int);`	Success	Success	Table testing is created.
`Stmt1 = Prepare(" select * from testing")`	Success	Success	Stmt1 is prepared with MXCMP.
`Stmt1.execute()`	Success	Success	Stmt1 is executed.
`Stmt1.fetch()`	Success	Success	Data in the table testing is retrieved.
`Stmt1 = Prepare(" select * from testing")`	Success	Success	Compiled plan is retrieved from MFC.
`Stmt1.execute()`	Success	Success	MFC statement works as expected.
`Stmt1.executeUpdate(" drop table testing")`	Success	Success	The table testing is dropped.
`Stmt1.executeUpdate(" create table testing (mycol varchar( 10))")`	Success	Success	The table testing is created with varchar column.
`Stmt1 = Prepare(" select * from testing")`	Success	Success	Compiled plan is retrieved from MFC, which is not correct because the table datatype is changed when the MFC plan is created.
`Stmt1.execute()`	Success	Failure	MXOSRVR turned the SQL/MX CQD recompilation_warnings ON. SQL/MX throws SQL exception upon similarity check failure and MXOSRVR drops the invalid module file from the `<compiled_module_location>` location.
`Stmt1 = Prepare(" select * from testing")`	Success	Success	A new plan is created in the MFC location.
`Stmt1.execute()`	Success	Success	MFC statement works as expected.

When performing the above operations, the execute() call fails when an invalid module file is found in the MFC. However, subsequent prepare() calls create a new module file. This open issue is similar to the Driver Side Cache (DSC) present in the JDBC/T4 driver.

A Migrating Applications from NonStop ODBC Server

You must migrate your NonStop ODBC Server applications if you plan to take advantage of the compiler and executor features of SQL/MX for your applications. By migrating your applications, you also take advantage of the ODBC 3.5 compliant API driver. Like SQL/MX (BSR X3.135-1992), ODBC/MX is based on the ISO/IEC 9075-3 standard, documented by Microsoft in the *ODBC 3.5 Programmer's Guide*.

- Most ODBC applications can be seamlessly migrated from the NonStop ODBC Server environment to the MXCS environment by duplicating the object names from NonStop ODBC Server.
- In some cases, minor changes are required to configure the SQL/MX environment to match the NonStop ODBC Server environment.

This section provides two checklists that describe necessary application modifications for migration in both the server and the client environments.

Topic

Server-Side Migration Steps

1. Install SQL/MX on the NonStop system (including MXCS). See *SQL/MX Installation and Management Guide* for detailed instructions.
2. Start the MXCS service by using MXOAS. For detailed information, see "Starting an MXCS Service manually" (page 20).
3. Duplicate the NonStop ODBC Server logical database object names to the SQL/MX environment. For detailed information, see "Server: Duplicating NonStop ODBC Server Logical Database Object Names" (page 56)
4. Migrate the NonStop ODBC Server configuration settings to MXCS. For detailed information, see "Server: Migrating NonStop ODBC Server Configuration Settings" (page 59).
5. Preserve the user alias settings. For detailed information, see "Server: Preserving User Alias Settings" (page 62).

For more information on migrating to SQL/MX Release 3.2 or later, see "Migrating from SQL/MX Release 2.3.4 to SQL/MX Release 3.2" (page 24).

Server: Duplicating NonStop ODBC Server Logical Database Object Names

Database Object Names

Both products use three-part logical names to identify database objects, as follows:

NonStop ODBC Server	`database.owner-name.object-name`
SQL/MX	`catalog.schema.object-name`

The following table lists the relationships between the database object names:

NonStop ODBC Server 3-part database name based on SQL Server implementation	Corresponding SQL/MX 3-part database name based on ANSI SQL 92 standard
Database name	Catalog
Owner	Schema
Object name	Object name

In the NonStop ODBC Server, a database name is in the form:

```
system-name_volume-name_subvolume-name
```

The *`system_name`* is the name of the NonStop system without the backslash (\). The *`volume_name`* is the name of the volume without the dollar sign ($), and *`subvol_name`* is the name of the subvolume. Do not use periods between the names because the name represents one logical name for each catalog and schema.

ODBC metadata information (through ODBC Catalog APIs) is returned based on the values for each of the three parts, which describe all database objects.

For ODBC applications that use NonStop ODBC Server object names, you must duplicate those names by using the logical ANSI names in the SQL/MX environment that match the names defined in the NonStop ODBC Server environment. All metadata information returned by ODBC/MX for those names will match the metadata information defined for NonStop ODBC Server.

NOTE: You do not need to duplicate names if the application does not rely on the NonStop ODBC Server logical names. However, if you want to use long object names, use the CREATE SQLMP ALIAS command in MXCI to add the long names to SQL/MX. The catalog and schema names must exist before you create the alias.

ANSI Names

ANSI names (also called alias names) are logical names that map to physical Guardian names of existing SQL/MP objects; they are defined by using the MXCI interface CREATE SQLMP ALIAS command. ANSI is the default name type.

To migrate to SQL/MX, use MXCI to move the NonStop ODBC Server database object names to the SQL/MX environment.

Name Mapping

NonStop ODBC Server uses customized catalogs to provide a three-part name mapping (logical names) for all SQL objects stored in a user SQL catalog. User catalog customization is done through the NOSCOM (or NOSUTIL) programs. An SQL/MP object name is not visible to the NonStop ODBC Server environment if it has not been customized. NonStop ODBC Server provides a three-part naming convention based on the Microsoft SQL Server implementation. The Owner (Schema) represents a collection of objects created by one user in the given Database name (catalog).

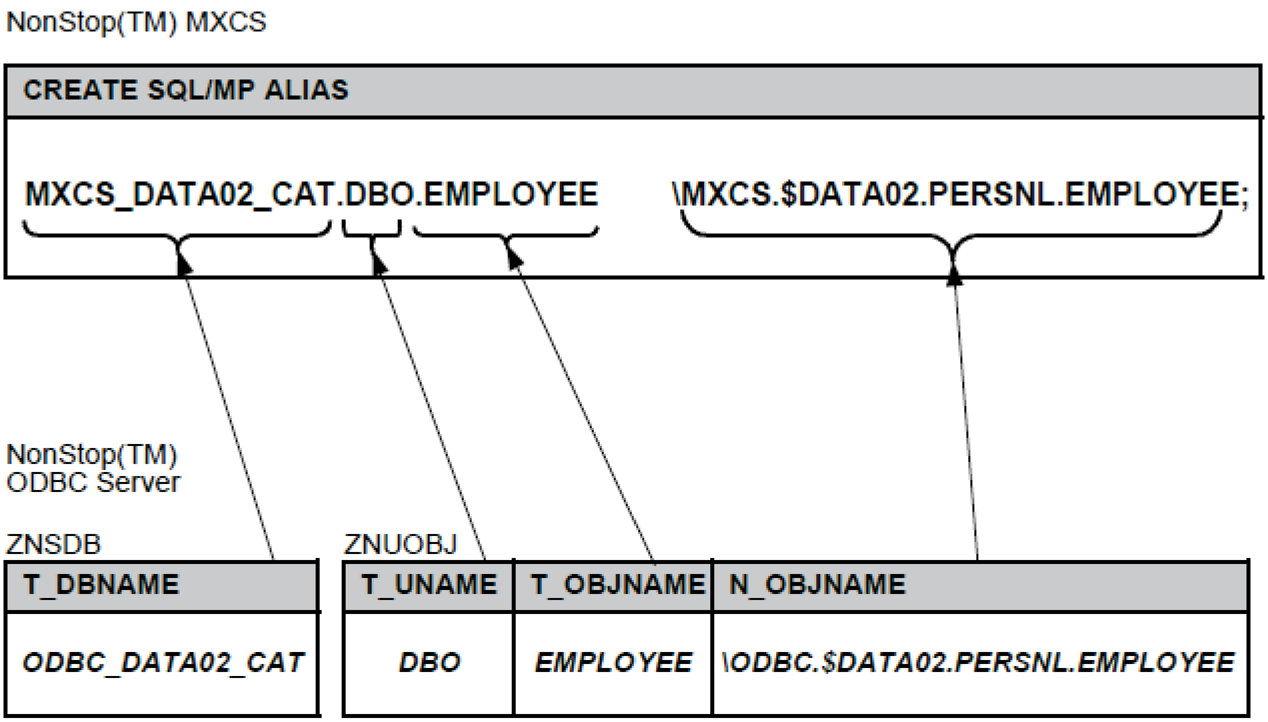
Within the ODBC/MX environment, you can add selected table names to the logical table names by using an SQL/MX command. The logical name is available through the SQL/MX MXCI in addition to client applications.

SQL/MX supports a three-part naming convention based on the ANSI SQL-92 standard. The catalog name represents a collection of schemas, and a schema represents a collection of database object names.

If the same names are required, you must add the corresponding alias names to tables used in NonStop ODBC Server by using the database name as the catalog name, the owner name as the schema name, and the object name as the SQL/MX object name. Object names can be defined in the SQL/MX environment by using the CREATE SQLMP ALIAS command in MXCI.

NOTE: When you create an SQL/MP alias, the catalog and schema must exist. If they do not, create them using MXCI before you create the alias.

Figure 3 Mapping NonStop ODBC Server Name Components to ODBC/MX



CREATE SQLMP ALIAS Command

```
CREATE SQLMP ALIAS catalog_name.schema_name.object_name Guardian_file_name;
```

catalog_name logical name of an existing SQL/MX catalog (database name in NonStop ODBC Server)

schema_name logical name for an existing SQL/MX schema (owner name in NonStop ODBC Server)

object_name logical name for the object name (table or view)

Guardian_file_name physical name (fully qualified) of the object name (table or view)

Using SQL Scripts to Automate Name Mapping

You can simplify the process of adding logical names based on NonStop ODBC Server names by running an SQL statement in MXCI that produces a script with SQL statements. You can edit the script and use MXCI to execute it to create the alias names.

For example, the following command generates an initial script containing SQL commands:

```
log mvscript;
select 'SELECT ''CREATE SQLMP ALIAS ''',
'||'T_DBNAME || ''.'' || T_UNAME || ''.'' || T_OBJNAME || '' '' ||N_OBJNAME',
'|| '';''',
'FROM ZNSDB, ', TRIM(N_CATALOG) || '.ZNUOBJ ',
'WHERE N_CATALOG = ''' || N_CATALOG || ''';'
FROM ZNSDB;
log;
```

After the script `mvscript` is generated, you must edit the script file to remove the headers and the initial SQL command. Make sure to leave the commands that insert the aliases into SQL/MX.

You must make some additional modifications to meet the specific installation rules when using SQL/MX.

The generated SQL commands for this example are:

```
select 'SELECT ''CREATE SQLMP ALIAS '' || T_DBNAME || ''.'' '
|| '|| T_UNAME || ''.'' || T_OBJNAME || '' '' || '
|| 'N_OBJNAME || '';'' '
|| 'from \TEST.$data02.startup.znsdb, '
|| trim(N_CATALOG) || '.ZNUOBJ WHERE '
|| 'N_CATALOG = ''' || N_CATALOG || ''';'
from \TEST.$data02.startup.znsdb;
```

where the ZNSDB table is in the system catalog and contains a list of NonStop ODBC Server databases, and \TEST.$DATA02.STARTUP is the catalog you are migrating to SQL/MX.

The same query is repeated for each of the NonStop ODBC Server customized catalogs to provide a complete mapping of NonStop ODBC Server objects to the SQL/MX environment. The generated MXCI commands have this form:

```
CREATE SQLMP ALIAS MASTER.DBO.TABLES \TEST.$DATA02.STARTUP.TABLES;
```

Accessing Migrated Names

After the names have been migrated to SQL/MX, you can access them through both MXCI and client applications.

Server: Migrating NonStop ODBC Server Configuration Settings

You do not need to migrate the existing NonStop ODBC Server configuration to MXCS. No exact correspondence exists between their settings. MXCS supports the same highly scalable and available servers as NonStop ODBC Server and achieves the same functionality even when specific settings do not match between products.

While NonStop ODBC Server configuration is performed through the NOSCOM program on the NonStop system, MXCS uses MXCI or NSM/web (a management console) to enter the required configuration parameters.

NOTE: If both products are used at the same time (co-exist) in the same system, the port number defined for both products must be different, and the MXCS port range should not overlap the NonStop ODBC Server port number.

Comparison of Configuration Terms

NonStop ODBC Server Configuration	MXCS Configuration
Resource governing	Same concept defined in the data source using the Resource Management tab.
Trace	Not Applicable
Query status	Not Applicable
Control statements	Same concept defined in the data source using the Control tab.
Profiles (maps to resource governing, resource accounting, trace and default SQL values)	MXCS has no profile concept. The data source configuration provides an equivalent concept for all users connecting to the data source (by name).
User name aliases (and map to a profile)	User aliases are not added to the ODBC environment. To preserve the same alias name, add them as Safeguard aliases. In addition, MXCS uses Guardian user names and IDs.
Server class (map them to a profile and user name)	Data source configuration (which includes control statements, define names, and SET environment variables).
SCS configuration	The configuration parameters are provided as part of the command line, as defined when starting the MXCS service. Configuration parameters are not stored and no check on overlapping parameters occurs between multiple instances of the MXCS service in the same system.

NonStop ODBC Server Configuration	MXCS Configuration
Network configuration	The configuration parameters are provided as part of the command line.
Map server class names to SCS names	Not Applicable. All data source names (and configuration values) are shared between all ODBC/MX service instances in the same system.
Map user name aliases to SCS and server class names	Not Applicable

To configure the MXCS environment, use NSM/web or MXCI. The service does not have to be started for you to configure MXCS data sources.

NonStop ODBC Server user aliases can be created in Safeguard at any time.

Resource Governing

Both products provide resource governing capabilities.

MXCS supports resource governing based on cost. When configured, MXCS sends an event message (EMS) to register any attempt to exceed the cost limit. The query can be stopped or executed. Resource governing policies are configured through the Resource Management tab in NSM/web.

NonStop ODBC Server Configuration	MXCS Configuration
Elapsed time	Not Applicable
Estimated cost	Same (only stop and log actions are supported)
Rows accessed	Not Applicable
Rows used	Not Applicable
Execution time	Not Applicable

Setting Control Statements

Use control statements to modify the execution of an SQL query.

You can configure control statements in both products but the statements are associated with different configuration objects. In the NonStop ODBC Server configuration, a control statement is associated with a profile. In MXCS, a control statement is associated with a data source (which implements the same concept as a profile).

An MXCS control statement can be any valid SQL/MX control statement and is executed at the time the client connects to the server. Use NSM/web or MXCI to configure the control statements for a data source.

Control statements defined for NonStop SQL/MP might not work with SQL/MX. For information on control statements, see *HP NonStop SQL/MX Release 3.2.1 Reference Manual*.

Configuring MXCS Data Sources

A server class in an NonStop ODBC Server environment corresponds to a data source in an MXCS environment. The following table compares the terminology between NonStop ODBC Server and MXCS.

NonStop ODBC Server Class Attributes	Corresponding MXCS Data Source Attributes
Guardian user name	User name is based on the login name for the connecting user ID; there is no default.
Priority	Priority is the same as specified for the association server or as defined in the data source configuration.
CPU list	All available CPUs in the system are used in a round-robin, or as defined in the data source configuration.
Available servers	Available servers

NonStop ODBC Server Class Attributes	Corresponding MXCS Data Source Attributes
Max servers	Maximum servers
Initial heap size	Not applicable
Maximum heap size	Not applicable
Idle delete delay timeout	Connection idle timeout
NOS create options	Environment variables
NOS debug options	Diagnostic information and resource management options provide an alternative debug method.
NOS run options	Run options are the same as specified for the association server.
SWAPVOL	Swap attributes are defined in the Kernel- Managed Swap Facility swap configuration.
LOGIN timeout	Not available on the server side; defined by the client data source.
Cancel timeout	Not available on the server side, defined by the client application.

To migrate to MXCS, use NSM/web or MXCI to define the data sources. See "Creating and Configuring Server Data Sources" (page 28).

Setting Network Attributes

The network configuration is defined in the command line when you start the MXCS service.

NonStop ODBC Server stores named network service configurations, which can be connected to a specific SCS. Network service parameters are:

- network service name
- protocol name
- IOP process name
- SCS process name
- Multilan specific parameters

MXCS supports only the TCP/IP protocol for remote clients and file system for local clients on NonStop servers. All parameters required by the network are defined in the command line when you start the association server.

The following table describes the network parameters available in both products:

NonStop ODBC Server	MXCS
Protocol	TCP/IP is the only protocol supported, no definition required
Network service name (NET_NAME)	Port number defined by the -PN parameter (it is not required to add it to the Services file)
IOP Process Name	TCP/IP process name defined by the -TCP parameter
Services file name	Not Applicable
Keep alive	Not Applicable
OOB Inline	OOB Inline
Linger	Not Applicable
Reuse address	Always allowed

Setting MXCS Service Attributes

The MXCS service is started differently from NonStop ODBC Server, which uses a configured value for SCS. For detailed information on starting MXCS, see "Starting an MXCS Service manually" (page 20).

Server: Preserving User Alias Settings

MXCS uses Safeguard aliases or Guardian user names; it does not use NonStop ODBC Server user alias definitions. In addition, MXCS does not associate user names with object names as part of the three-part name unless the you configure them that way. SQL/MX schema names can be any user-defined logical name. NonStop ODBC Server profile information is stored only by data source name (server class), and all users share access to all data sources in any instance of the association server in the same NonStop system.

The NonStop ODBC Server environment maps existing Guardian user names with user name aliases. Use these aliases in NonStop ODBC Server to identify:

- The owner (schema) of object names in a given database (catalog).
- User profile information (default database name, default schema name, default location for the creation of objects, default security, trace configuration, accounting configuration, resource governing configuration, SQL access mode, cursor mode, SQL dialect, maximum statement caching, control statements).
- The mapping between SCS and Server class names.

To preserve the user aliases in the NonStop ODBC Server environment, you must add those user alias names to the Guardian Safeguard system. For more details on how to add a user alias, see the Safeguard documentation.

To preserve a user alias name as the second part of the name (schema name), you must explicitly add the tables by using that logical name. MXCS does not automatically link between the Safeguard system and the MP_PARTITIONS name.

NOTE: No action is required if you use Guardian user IDs to connect to the MXCS environment.

Client-Side Migration Steps

1. Install the ODBC/MX client interface. For additional information, see the *ODBC/MX Driver for Windows Manual*.
2. Modify your ODBC applications then execute the ODBC applications through the new ODBC/MX driver. For detailed information, see "Client: Modifying Applications" (page 62).

Most ODBC applications do not need to be modified because the Microsoft driver manager provides an automatic mapping for ODBC API calls between a SQL/MX Release 2.x application and a SQL/MX Release 3.x driver. You might need to make changes when using features that are specific to NonStop ODBC Server (for example, pass-through commands or using TSQL and some Microsoft SQL Server-like system tables).

Client: Modifying Applications

To migrate ODBC-compliant applications that do not use or require specific NonStop ODBC Server features, simply redefine the client data sources to use the ODBC/MX driver. When redefining client data sources, you may use the same name to replace the previous data source definition, but you must delete the old definition and create the new definition with that same name.

To migrate ODBC applications that use specific NonStop ODBC Server features, you must replace the existing proprietary features with equivalent ODBC/MX features.

NOTE: You do not need to migrate the ODBC API calls from a SQL/MX Release 2.x application to a SQL/MX Release 3.x application because the Driver Manager already provides the version mapping.

You must modify:

- SQL statements (to comply with SQL/MX syntax)
- NonStop ODBC Server-specific syntax commands

ODBC API calls

ODBC 2.*x* compliant applications do not need to be migrated to support the new 3.x functions. The Microsoft ODBC driver manager allows version 2.*x* applications to run using version 3.*x* -compliant drivers and vice versa. It automatically maps version 2.*x* application functions to version 3.*x*-compliant drivers, version 3.*x* application functions to version 2.*x*-compliant drivers, and it performs mapping between 2.*x* and 3.*x* functions and parameter options.

ODBC/MX includes a 3.5-compliant ANSI driver; it does not include a Unicode API. However, the driver manager automatically provides the mapping between the Unicode API and the ANSI API.

Table 9 (page 63) shows how ODBC 2.x calls map to ODBC 3.x, and whether NonStop ODBC and ODBC/MX support each call.

Table 9 ODBC API Mapping Between 2.x and 3.x Versions

ODBC 2.x Calls*	ODBC 3.x Calls*	3.*x* Calls Supported by ODBC/MX?	2.*x* Calls Supported by ODBC Server?
SQLAllocConnect	*SQLAllocHandle*	Y	Y
SQLAllocEnv	*SQLAllocHandle*	Y	Y
SQLAllocStmt	*SQLAllocHandle*	Y	Y
SQLBindCol	SQLBindCol	Y	Y
SQLBindParameter	SQLBindParameter	Y	Y
SQLBrowseConnect	SQLBrowseConnect	Y	Y
No equivalent	SQLBulkOperations	N	N
SQLCancel	SQLCancel	Y	Y
SQLColAttributes	*SQLColAttribute*	Y	Y
No equivalent	SQLCloseCursor	Y	N
SQLColumnPrivileges	SQLColumnPrivileges	Y	N
SQLColumns	SQLColumns	Y	Y
SQLConnect	SQLConnect	Y	Y
No equivalent	SQLCopyDesc	Y	N
SQLDataSources	SQLDataSources	Y	Y
SQLDescribeCol	SQLDescribeCol	Y	Y
SQLDescribeParam	SQLDescribeParam	Y	N
SQLDisconnect	SQLDisconnect	Y	Y
SQLDriverConnect	SQLDriverConnect	Y	Y
No equivalent	SQLDrivers	Y	N
SQLError	*SQLGetDiagField*	Y	Y
No equivalent	SQLGetDiagRec	Y	N
SQLExecDirect	SQLExecDirect	Y	Y
SQLExecute	SQLExecute	Y	Y

Table 9 ODBC API Mapping Between 2.x and 3.x Versions *(continued)*

ODBC 2.x Calls*	ODBC 3.x Calls*	3.x Calls Supported by ODBC/MX?	2.x Calls Supported by ODBC Server?
SQLExtendedFetch	*SQLFetchScroll*	Y	N
SQLFetch	SQLFetch	Y	Y
SQLForeignKeys	SQLForeignKeys	Y	N
SQLFreeConnect	*SQLFreeHandle*	Y	Y
SQLFreeEnv	*SQLFreeHandle*	Y	Y
SQLFreeStmt	*SQLFreeHandle*	Y	Y
SQLGetConnectOption	*SQLGetConnectAttr*	Y	Y
SQLGetCursorName	SQLGetCursorName	Y	Y
SQLGetData	SQLGetData	Y	Y
No equivalent	SQLGetDescField	Y	N
No equivalent	SQLGetDescRec	Y	N
No equivalent	SQLGetEnvAttr	Y	N
SQLGetFunctions	SQLGetFunctions	Y	Y
SQLGetInfo	SQLGetInfo	Y	Y
SQLGetStmtOption	*SQLGetStmtAttr*	Y	Y
SQLGetTypeInfo	SQLGetTypeInfo	Y	Y
SQLMoreResults	SQLMoreResults	Y	Y
SQLNativeSql	SQLNativeSql	Y	Y
SQLNumParams	SQLNumParams	Y	Y
SQLNumResultCols	SQLNumResultCols	Y	Y
SQLParamData	SQLParamData	Y	Y
SQLParamOptions	SQLSetStmtAttr	Y	N
SQLPrepare	SQLPrepare	Y	Y
SQLPrimaryKeys	SQLPrimaryKeys	Y	Y
SQLProcedureColumns	SQLProcedureColumns	Y	Y
SQLProcedures	SQLProcedures	Y	Y
SQLPutData	SQLPutData	Y	Y
SQLRowCount	SQLRowCount	Y	Y
SQLSetConnectOption	*SQLSetConnectAttr*	Y	Y
SQLSetCursorName	SQLSetCursorName	Y	Y
No equivalent	SQLSetDescField	Y	N
No equivalent	SQLSetDescRec	Y	N
No equivalent	SQLSetEnvAttr	Y	N
No equivalent	SQLSetStmtAttr	Y	N
SQLSetParam	*SQLBindParameter*	Y	Y
SQLSetPos	SQLSetPos	Y	Y

Table 9 ODBC API Mapping Between 2.x and 3.x Versions *(continued)*

ODBC 2.x Calls*	ODBC 3.x Calls*	3.*x* Calls Supported by ODBC/MX?	2.*x* Calls Supported by ODBC Server?
SQLSetScrollOption	SQLSetScrollOption	N	N
SQLSetStmtOption	SQLSetStmtAttr	Y	Y
SQLSpecialColumns	SQLSpecialColumns	Y	Y
SQLStatistics	SQLStatistics	Y	Y
SQLTablePrivileges	SQLTablePrivileges	Y	N
SQLTables	SQLTables	Y	Y
SQLTransact	*SQLEndTran*	Y	Y

* Function names in italics are converted between both versions.

Modifying SQL Statements

NonStop ODBC Server uses Core SQL statements for ODBC connections and T-SQL statements for DBLib clients. Core SQL language is a subset of the ANSI-92 standard.

ODBC/MX supports all DML SQL statements that are defined in SQL/MX and are compliant with the ANSI-92 standard. In addition, SQL/MX supports ODBC extensions for function calls and datetime literals.

ODBC/MX supports all DDL statements as defined in SQL/MX.

Grant and Revoke statements are supported as defined in SQL/MX.

Migrating SQL Server Objects

To migrate T-SQL system table objects, you must define them in the ODBC/MX environment. The following T-SQL system table names are used by DBLib and have no ODBC/MX equivalents:

SYSCOLUMNS
SYSDATABASES
SYSINDEXES
SYSMESSAGES
SYSOBJECTS
SYSPROTECTS
SYSTYPES
SYSUSERS

NOTE: HP recommends that an ODBC-compliant application use the ODBC Catalog APIs to retrieve catalog information instead of selecting data from Microsoft SQL Server system tables.

If NonStop ODBC Server is running, you can migrate the system tables objects by adding the logical name mappings for those tables into the MP_PARTITIONS table; use the ANSI names to select the tables. To prevent the name mappings from getting out of sync, use this method only if your NonStop ODBC Server environment is not creating or dropping tables.

Modifying Pass-Through Commands

MXCS does not support the NonStop ODBC Server pass-through syntax. To migrate, you must make syntax changes to achieve the same behavior.

The following table lists the MXCS equivalents for pass-through commands.

NonStop ODBC Server Pass-Through Commands	Column HeadMXCS Equivalents
SQL commands for sending SQL/MP commands	Change the syntax to SQL/MX syntax.
UTIL commands for executing ODBC utility commands	Not supported. However, you can add new table logical names using the CREATE SQLMP ALIAS command and drop

NonStop ODBC Server Pass-Through Commands	Column HeadMXCS Equivalents
	table logical names using the DROP SQLMP ALIAS command.
SET commands for setting environment variables	If the SQL/MX language supports the commands, they can be sent directly. CONTROL statements defined in SQL/MX can also be sent (no special syntax is required).

Features Not Supported

The following NonStop ODBC Server features are not supported:

- DBLib messages using the TDS transport. To work around this, you can use an intermediate gateway to convert TDS messages to ODBC calls.
- IPX/SPX and Named Pipes network protocols. To work around this, you can use an intermediate gateway to convert protocols to TCP/IP.

NonStop ODBC Server Migration Example

Server Characteristics

In this example, the server has the following characteristics:

System Catalog:	\TEST.$DATA02.SYSCAT
User Catalog:	\TEST.$DATA07.SAMDBCA
User Tables:	\TEST.$DATA04.PERSNL.EMPLOYEE
	\TEST.$DATA04.PERSNL.JOB
	\TEST.$DATA04.PERSNL.DEPT
	\TEST.$DATA04.PERSNL.MGRLIST
Table owner	DBAGR.PRSNL
NonStop ODBC Server users:	ODBCUSR1, which is mapped to the user ID SQLUSR.USER1
	and
	ODBCUSR2, which is mapped to the user ID SQLUSR.USER2.

System Catalog Names

Database Name	Owner Name	Object Name	Guardian Name
MASTER	DBO	BASETABS	\TEST.$DATA02.SYSCAT.BASETABS
MASTER	DBO	CATALOGS	\TEST.$DATA02.SQL.CATALOGS
MASTER	DBO	COLUMNS	\TEST.$DATA02.SYSCAT.COLUMNS
MASTER	DBO	COMMENTS	\TEST.$DATA02.SYSCAT.COMMENTS
MASTER	DBO	CONSTRNT	\TEST.$DATA02.SYSCAT.CONSTRNT
MASTER	DBO	CPRLSRCE	\TEST.$DATA02.SYSCAT.CPRLSRCE
MASTER	DBO	CPRULES	\TEST.$DATA02.SYSCAT.CPRULES
MASTER	DBO	FILES	\TEST.$DATA02.SYSCAT.FILES
MASTER	DBO	INDEXES	\TEST.$DATA02.SYSCAT.INDEXES
MASTER	DBO	KEYS	\TEST.$DATA02.SYSCAT.KEYS
MASTER	DBO	PARTNS	\TEST.$DATA02.SYSCAT.PARTNS
MASTER	DBO	PROGRAMS	\TEST.$DATA02.SYSCAT.PROGRAMS
MASTER	DBO	SYSCOLUMNS	\TEST.$DATA02.SYSCAT.ZVUCOL
MASTER	DBO	SYSDATABASES	\TEST.$DATA02.SYSCAT.ZVSDB

Database Name	Owner Name	Object Name	Guardian Name
MASTER	DBO	SYSINDEXES	\TEST.$DATA02.SYSCAT.ZVUIX
MASTER	DBO	SYSMESSAGES	\TEST.$DATA02.SYSCAT.ZVSMSG
MASTER	DBO	SYSOBJECTS	\TEST.$DATA02.SYSCAT.ZVUOBJ
MASTER	DBO	SYSPROTECTS	\TEST.$DATA02.SYSCAT.ZVUPROT
MASTER	DBO	SYSTYPES	\TEST.$DATA02.SYSCAT.ZVUDT
MASTER	DBO	SYSUSERS	\TEST.$DATA02.SYSCAT.ZVUUS
MASTER	DBO	TABLES	\TEST.$DATA02.SYSCAT.TABLES
MASTER	DBO	TRANSIDS	\TEST.$DATA02.SYSCAT.TRANSIDS
MASTER	DBO	USAGES	\TEST.$DATA02.SYSCAT.USAGES
MASTER	DBO	VERSIONS	\TEST.$DATA02.SYSCAT.VERSIONS
MASTER	DBO	VIEWS	\TEST.$DATA02.SYSCAT.VIEWS
MASTER	DBO	ZNSALT	\TEST.$DATA02.SYSCAT.ZNSALT
MASTER	DBO	ZNSCON	\TEST.$DATA02.SYSCAT.ZNSCON
MASTER	DBO	ZNSDB	\TEST.$DATA02.SYSCAT.ZNSDB
MASTER	DBO	ZNSDEF	\TEST.$DATA02.SYSCAT.ZNSDEF
MASTER	DBO	ZNSDUMMY	\TEST.$DATA02.SYSCAT.ZNSDUMMY
MASTER	DBO	ZNSGOV	\TEST.$DATA02.SYSCAT.ZNSGOV
MASTER	DBO	ZNSMSG	\TEST.$DATA02.SYSCAT.ZNSMSG
MASTER	DBO	ZNSNET	\TEST.$DATA02.SYSCAT.ZNSNET
MASTER	DBO	ZNSPROF	\TEST.$DATA02.SYSCAT.ZNSPROF
MASTER	DBO	ZNSPROT	\TEST.$DATA02.SYSCAT.ZNSPROT
MASTER	DBO	ZNSSCFG	\TEST.$DATA02.SYSCAT.ZNSSCFG
MASTER	DBO	ZNSSCS	\TEST.$DATA02.SYSCAT.ZNSSCS
MASTER	DBO	ZNSSER	\TEST.$DATA02.SYSCAT.ZNSSER
MASTER	DBO	ZNSSMAP	\TEST.$DATA02.SYSCAT.ZNSSMAP
MASTER	DBO	ZNSTRA	\TEST.$DATA02.SYSCAT.ZNSTRA
MASTER	DBO	ZNSUMAP	\TEST.$DATA02.SYSCAT.ZNSUMAP
MASTER	DBO	ZNSUS	\TEST.$DATA02.SYSCAT.ZNSUS
MASTER	DBO	ZNSVALUE	\TEST.$DATA02.SYSCAT.ZNSVALUE
MASTER	DBO	ZNUDT	\TEST.$DATA02.SYSCAT.ZNUDT
MASTER	DBO	ZNUIX	\TEST.$DATA02.SYSCAT.ZNUIX
MASTER	DBO	ZNUOBJ	\TEST.$DATA02.SYSCAT.ZNUOBJ
MASTER	DBO	ZNUPCOL	\TEST.$DATA02.SYSCAT.ZNUPCOL
MASTER	DBO	ZNUPROC	\TEST.$DATA02.SYSCAT.ZNUPROC
MASTER	DBO	ZVUOCOL	\TEST.$DATA02.SYSCAT.ZVUOCOL
MASTER	DBO	ZVUPCOL	\TEST.$DATA02.SYSCAT.ZVUPCOL
MASTER	DBO	CUSTOMER	\TEST.$DATA02.SYSCAT.CUSTOMER
MASTER	DBO	MP_PARTITIONS	\TEST.$DATA02.SYSCAT.MP_PARTITIONS
MASTER	DBO	ZOAS2DS	\TEST.$DATA02.SYSCAT.ZOAS2DS
MASTER	DBO	ZODS	\TEST.$DATA02.SYSCAT.ZODS
MASTER	DBO	ZODT	\TEST.$DATA02.SYSCAT.ZODT
MASTER	DBO	ZOENV	\TEST.$DATA02.SYSCAT.ZOENV

Database Name	Owner Name	Object Name	Guardian Name
MASTER	DBO	ZONAM2ID	\TEST.$DATA02.SYSCAT.ZONAM2ID
MASTER	DBO	ZORES	\TEST.$DATA02.SYSCAT.ZORES
MASTER	DBO	ZTEST	\TEST.$DATA02.SYSCAT.ZTEST

User Catalog Names

Database Name	Owner Name	Object Name	Guardian Name
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	BASETABS	\TEST.$DATA07. SAMDBCAT. BASETABS
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	COLUMNS	\TEST.$DATA07. SAMDBCAT. COLUMNS
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	COMMENTS	\TEST.$DATA07. SAMDBCAT. COMMENTS
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	CONSTRNT	\TEST.$DATA07. SAMDBCAT. CONSTRNT
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	CPRLSRCE	\TEST.$DATA07. SAMDBCAT. CPRLSRCE
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	CPRULES	\TEST.$DATA07. SAMDBCAT. CPRULES
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	FILES	\TEST.$DATA07. SAMDBCAT.FILES
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	INDEXES	\TEST.$DATA07. SAMDBCAT. INDEXES
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	KEYS	\TEST.$DATA07. SAMDBCAT. KEYS
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	PARTNS	\TEST.$DATA07. SAMDBCAT. PARTNS
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	PROGRAMS	\TEST.$DATA07. SAMDBCAT. PROGRAMS
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	SYSCOLUMNS	\TEST.$DATA07. SAMDBCAT. ZVUCOL
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	SYSINDEXES	\TEST.$DATA07. SAMDBCAT. ZVUIX
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	SYSOBJECTS	\TEST.$DATA07. SAMDBCAT. ZVUOBJ
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	SYSPROTECTS	\TEST.$DATA07. SAMDBCAT. ZVUPROT
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	SYSTYPES	\TEST.$DATA07. SAMDBCAT. ZVUDT
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	SYSUSERS	\TEST.$DATA07. SAMDBCAT. ZVUUS
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	TABLES	\TEST.$DATA07. SAMDBCAT. TABLES
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	TRANSIDS	\TEST.$DATA07. SAMDBCAT. TRANSIDS
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	USAGES	\TEST.$DATA07. SAMDBCAT. USAGES
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	VERSIONS	\TEST.$DATA07. SAMDBCAT. VERSIONS
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	VIEWS	\TEST.$DATA07. SAMDBCAT. VIEWS

Database Name	Owner Name	Object Name	Guardian Name
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	ZNUDT	\TEST.$DATA07. SAMDBCAT. ZNUDT
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	ZNUIX	\TEST.$DATA07. SAMDBCAT. ZNUIX
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	ZNUOBJ	\TEST.$DATA07. SAMDBCAT. ZNUOBJ
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	ZNUPCOL	\TEST.$DATA07. SAMDBCAT. ZNUPCOL
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	ZNUPROC	\TEST.$DATA07. SAMDBCAT. ZNUPROC
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	ZVUOCOL	\TEST.$DATA07. SAMDBCAT. ZVUOCOL
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	ZVUPCOL	\TEST.$DATA07. SAMDBCAT. ZVUPCOL
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	CUSTLIST	\TEST.$DATA07. SALES.CUSTLIST
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	CUSTOMER	\TEST.$DATA07. SALES. CUSTOMER
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	DEPT	\TEST.$DATA07. PERSNL.DEPT
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	EMPLIST	\TEST.$DATA07. PERSNL.EMPLIST
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	EMPLOYEE	\TEST.$DATA07. PERSNL. EMPLOYEE
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	JOB	\TEST.$DATA07. PERSNL.JOB
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	KEVIN	\TEST.$DATA07. SAMDBCAT. KEVIN
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	MGRLIST	\TEST.$DATA07. PERSNL.MGRLIST
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	ODETAIL	\TEST.$DATA07. SALES.ODETAIL
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	ORDERS	\TEST.$DATA07. SALES.ORDERS
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	ORDREP	\TEST.$DATA07. SALES.ORDREP
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	PARTLOC	\TEST.$DATA07. INVENT.PARTLOC
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	PARTS	\TEST.$DATA07. SALES.PARTS
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	PARTSUPP	\TEST.$DATA07. INVENT. PARTSUPP
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	SUPPLIER	\TEST.$DATA07. SALES.SUPPLIER
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	VIEW207	\TEST.$DATA07. INVENT.VIEW207
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	VIEW207N	\TEST.$DATA07. INVENT. VIEW207N

Database Name	Owner Name	Object Name	Guardian Name
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	VIEWCS	\TEST.$DATA07. INVENT.VIEWCS
TEST_DATA07_SAMDBCAT	DBAGRP_ PRSNL	VIEWCUST	\TEST.$DATA07. INVENT. VIEWCUST

Configuration Example for the NonStop ODBC Server

config	$nsoad
NSODBC_SYSTEM_CATALOG	\TEST.$DATA02.SYSCAT
CHECK_INTERVAL_SECS	600
NOS_OBJECT	\TEST.$DATA09.NOS7.NOS
NOS_LIBRARY_FILE	
NOSUTIL_OBJECT	\TEST.$DATA09.NOS7.NOSUTIL
NOSUTIL_LIBRARY_FILE	
NOSUTIL_RUN_OPTIONS	
NOSUTIL_PRIORITY	140
NOSUTIL_CPU	0
NOSUTIL_CREATE_OPTIONS	0
NOSUTIL_DEBUG_OPTIONS	0
IN_BUFFER_SIZE_B	2448
OUT_BUFFER_SIZE_B	2448
SCS_RUN_OPTIONS	
EMIT_EVENTS	N
MEMORY_CHECK	0
SERVERCLASSES_CONFIGURED	1

SCS_NAME	\TEST.$NSOAD
JOB_ID	1
SCS_OBJECT	\TEST.$DATA09.NOS7.SCSOBJ
SCS_LIBRARY	
SCS_PRIORITY	140
PRIMARY_CPU	0
BACKUP_CPU	
SWAPVOL	
EXT_SWAPFILE	
DATAPAGES	2040
SCS_CREATE_OPTIONS	
EMIT_EVENTS	N
MEMORY_CHECK	0
INFILE	
HOMETERM	
OUTFILE	
ERRFILE	
IN_BUFFER_SIZE_B	2448

SCS_NAME	**\TEST.$NSOAD**
OUT_BUFFER_SIZE_B	2448
DEFAULT_VOLUME	
SCS_DEBUG_OPTIONS	
SCS_RUN_OPTIONS	
NOS_OBJECT	\TEST.$DATA09.NOS7.NOS
NOSUTIL_LIBRARY_FILE	
NOSUTIL_PRIORITY	140
NOSUTIL_CPU	
NOSUTIL_CREATE_OPTIONS	
NOSUTIL_DEBUG_OPTIONS	
NOSUTIL_RUN_OPTIONS	

NET_NAME	**NSOADMIN**
NET_PROTOCOL	TCP/IP
SERVICES_FILENAME	
IOP_NAME	\TEST.$ZTC0
SO_KEEPALIVE	1
SO_OOBINLINE	1
SO_LINGER	0
SO_REUSEADDR	1

NSOADMIN is defined as port number 2400 in the Services file.

SERVERCLASS	**NSOADMIN**
INITIALIZATION PRIORITY:	140
CPU_LIST:	(0,1)
AVAILABLE_SERVERS	1
MAX_SERVERS	5
INITIALIZATION G_USERNAME	SUPER.SUPER
INITIALIZATION PROFILE	NSOADMINPROF
INIT_HEAP_SIZE_KB	750
MAX_HEAP_SIZE_KB	750
IDLE_DELETE_DELAY_SEC	0
NOS_CREATE_OPTIONS	0
NOS_DEBUG_OPTIONS	0
NOS_RUN_OPTIONS	
SWAPVOL:	
LOGIN_TIMEOUT_SEC:	
CANCEL_TIMEOUT_SEC:	

PROFILE_NAME	**NSOADMINPROF**
DEFAULT_DATABASE	MASTER
DEFAULT_SCHEMA	
DEFAULT_LOCATION	

PROFILE_NAME	NSOADMINPROF
DEFAULT_SECURITY	NNNU
TRA_MODE_ON	
TRA_NAME	
ACC_MODE_ON	N
ACC_LOGTABLE_NAME	
ACC_LEVEL	SESSION
GOV_MODE_ON	N
GOV_NAME	
QST_MODE_ON	N
SQL_ACCESS_MODE	RW
SQL_CURSOR_MODE	RW
SQL_DIALECT	TDM_CORE
SQL_MAX_STATEMENT_CACHE	0
SQL_TXN_ISOLATION	1
SQL_UNSUPPORTED	E
OBJ_NAME_CACHE	Y
STMT_CACHE_LEVEL	1
CON_MODE_ON	N
CON_NAME	
CLOSE_TABLES_PER_SESSION	N

Adding Aliases for System Catalog Object Names

To add NonStop ODBC Server system catalog object names to the SQL/MX environment, use MXCI to execute the following commands:

```
CREATE CATALOG MASTER;
CREATE SCHEMA MASTER.DBO;
CREATE SQLMP ALIAS MASTER.DBO.BASETABS \TEST1.$DATA02.SYSCAT.BASETABS;
CREATE SQLMP ALIAS MASTER.DBO.CATALOGS \TEST1.$DATA02.SQL.CATALOGS;
CREATE SQLMP ALIAS MASTER.DBO.COLUMNS \TEST1.$DATA02.SYSCAT.COLUMNS;
CREATE SQLMP ALIAS MASTER.DBO.COMMENTS \TEST1.$DATA02.SYSCAT.COMMENTS;
CREATE SQLMP ALIAS MASTER.DBO.CONSTRNT \TEST1.$DATA02.SYSCAT.CONSTRNT;
CREATE SQLMP ALIAS MASTER.DBO.CPRLSRCE \TEST1.$DATA02.SYSCAT.CPRLSRCE;
CREATE SQLMP ALIAS MASTER.DBO.CPRULES \TEST1.$DATA02.SYSCAT.CPRULES;
CREATE SQLMP ALIAS MASTER.DBO.FILES \TEST1.$DATA02.SYSCAT.FILES;
CREATE SQLMP ALIAS MASTER.DBO.INDEXES \TEST1.$DATA02.SYSCAT.INDEXES;
CREATE SQLMP ALIAS MASTER.DBO.KEYS \TEST1.$DATA02.SYSCAT.KEYS;
CREATE SQLMP ALIAS MASTER.DBO.PARTNS \TEST1.$DATA02.SYSCAT.PARTNS;
CREATE SQLMP ALIAS MASTER.DBO.PROGRAMS \TEST1.$DATA02.SYSCAT.PROGRAMS;
CREATE SQLMP ALIAS MASTER.DBO.SYSCOLUMNS \TEST1.$DATA02.SYSCAT.ZVUCOL;
CREATE SQLMP ALIAS MASTER.DBO.SYSDATABASES \TEST1.$DATA02.SYSCAT.ZVSDB;
CREATE SQLMP ALIAS MASTER.DBO.SYSINDEXES \TEST1.$DATA02.SYSCAT.ZVUIX;
CREATE SQLMP ALIAS MASTER.DBO.SYSMESSAGES \TEST1.$DATA02.SYSCAT.ZVSMSG;
CREATE SQLMP ALIAS MASTER.DBO.SYSOBJECTS \TEST1.$DATA02.SYSCAT.ZVUOBJ;
CREATE SQLMP ALIAS MASTER.DBO.SYSPROTECTS \TEST1.$DATA02.SYSCAT.ZVUPROT;
CREATE SQLMP ALIAS MASTER.DBO.SYSTYPES \TEST1.$DATA02.SYSCAT.ZVUDT;
CREATE SQLMP ALIAS MASTER.DBO.SYSUSERS \TEST1.$DATA02.SYSCAT.ZVUUS;
CREATE SQLMP ALIAS MASTER.DBO.TABLES \TEST1.$DATA02.SYSCAT.TABLES;
CREATE SQLMP ALIAS MASTER.DBO.TRANSIDS \TEST1.$DATA02.SYSCAT.TRANSIDS;
CREATE SQLMP ALIAS MASTER.DBO.USAGES \TEST1.$DATA02.SYSCAT.USAGES;
CREATE SQLMP ALIAS MASTER.DBO.VERSIONS \TEST1.$DATA02.SYSCAT.VERSIONS;
CREATE SQLMP ALIAS MASTER.DBO.VIEWS \TEST1.$DATA02.SYSCAT.VIEWS;
CREATE SQLMP ALIAS MASTER.DBO.ZNSALT \TEST1.$DATA02.SYSCAT.ZNSALT;
CREATE SQLMP ALIAS MASTER.DBO.ZNSCON \TEST1.$DATA02.SYSCAT.ZNSCON;
CREATE SQLMP ALIAS MASTER.DBO.ZNSDB \TEST1.$DATA02.SYSCAT.ZNSDB;
CREATE SQLMP ALIAS MASTER.DBO.ZNSDEF \TEST1.$DATA02.SYSCAT.ZNSDEF;
CREATE SQLMP ALIAS MASTER.DBO.ZNSDUMMY \TEST1.$DATA02.SYSCAT.ZNSDUMMY;
CREATE SQLMP ALIAS MASTER.DBO.ZNSGOV \TEST1.$DATA02.SYSCAT.ZNSGOV;
CREATE SQLMP ALIAS MASTER.DBO.ZNSMSG \TEST1.$DATA02.SYSCAT.ZNSMSG;
CREATE SQLMP ALIAS MASTER.DBO.ZNSNET \TEST1.$DATA02.SYSCAT.ZNSNET;
CREATE SQLMP ALIAS MASTER.DBO.ZNSPROF \TEST1.$DATA02.SYSCAT.ZNSPROF;
CREATE SQLMP ALIAS MASTER.DBO.ZNSPROT \TEST1.$DATA02.SYSCAT.ZNSPROT;
CREATE SQLMP ALIAS MASTER.DBO.ZNSSCFG \TEST1.$DATA02.SYSCAT.ZNSSCFG;
CREATE SQLMP ALIAS MASTER.DBO.ZNSSCS \TEST1.$DATA02.SYSCAT.ZNSSCS;
CREATE SQLMP ALIAS MASTER.DBO.ZNSSER \TEST1.$DATA02.SYSCAT.ZNSSER;
CREATE SQLMP ALIAS MASTER.DBO.ZNSSMAP \TEST1.$DATA02.SYSCAT.ZNSSMAP;
CREATE SQLMP ALIAS MASTER.DBO.ZNSTRA \TEST1.$DATA02.SYSCAT.ZNSTRA;
CREATE SQLMP ALIAS MASTER.DBO.ZNSUMAP \TEST1.$DATA02.SYSCAT.ZNSUMAP;
CREATE SQLMP ALIAS MASTER.DBO.ZNSUS \TEST1.$DATA02.SYSCAT.ZNSUS;
CREATE SQLMP ALIAS MASTER.DBO.ZNSVALUE \TEST1.$DATA02.SYSCAT.ZNSVALUE;
CREATE SQLMP ALIAS MASTER.DBO.ZNUDT \TEST1.$DATA02.SYSCAT.ZNUDT;
CREATE SQLMP ALIAS MASTER.DBO.ZNUIX \TEST1.$DATA02.SYSCAT.ZNUIX;
CREATE SQLMP ALIAS MASTER.DBO.ZNUOBJ \TEST1.$DATA02.SYSCAT.ZNUOBJ;
CREATE SQLMP ALIAS MASTER.DBO.ZNUPCOL \TEST1.$DATA02.SYSCAT.ZNUPCOL;
CREATE SQLMP ALIAS MASTER.DBO.ZNUPROC \TEST1.$DATA02.SYSCAT.ZNUPROC;
CREATE SQLMP ALIAS MASTER.DBO.ZVUOCOL \TEST1.$DATA02.SYSCAT.ZVUOCOL;
CREATE SQLMP ALIAS MASTER.DBO.ZVUPCOL \TEST1.$DATA02.SYSCAT.ZVUPCOL;
CREATE SQLMP ALIAS MASTER.DBO.CUSTOMER \TEST1.$DATA02.SYSCAT.CUSTOMER;
CREATE SQLMP ALIAS MASTER.DBO.MP_PARTITIONS
\TEST1.$DATA02.SYSCAT.MP_PARTITIONS;
CREATE SQLMP ALIAS MASTER.DBO.TEMP3 \TEST1.$DATA02.SYSCAT.TEMP3;
CREATE SQLMP ALIAS MASTER.DBO.ZOAS2DS \TEST1.$DATA02.SYSCAT.ZOAS2DS;
CREATE SQLMP ALIAS MASTER.DBO.ZODS \TEST1.$DATA02.SYSCAT.ZODS;
CREATE SQLMP ALIAS MASTER.DBO.ZODT \TEST1.$DATA02.SYSCAT.ZODT;
CREATE SQLMP ALIAS MASTER.DBO.ZOENV \TEST1.$DATA02.SYSCAT.ZOENV;
CREATE SQLMP ALIAS MASTER.DBO.ZONAM2ID \TEST1.$DATA02.SYSCAT.ZONAM2ID;
CREATE SQLMP ALIAS MASTER.DBO.ZORES \TEST1.$DATA02.SYSCAT.ZORES;
CREATE SQLMP ALIAS MASTER.DBO.ZTEST \TEST1.$DATA02.SYSCAT.ZTEST;
```

Adding Aliases for User Catalog Names

To add user catalog names, create a script for MXCI that contains the following commands:

```
CREATE CATALOG TEST_DATA07_SAMDBCAT;
CREATE SCHEMA TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.BASETABS
\TEST.$DATA07.SAMDBCAT.BASETABS;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.COLUMNS
\TEST.$DATA07.SAMDBCAT.COLUMNS ;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.COMMENTS
\TEST.$DATA07.SAMDBCAT.COMMENTS;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.CONSTRNT
\TEST.$DATA07.SAMDBCAT.CONSTRNT;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.CPRLSRCE
\TEST.$DATA07.SAMDBCAT.CPRLSRCE;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.CPRULES
\TEST.$DATA07.SAMDBCAT.CPRULES ;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.FILES
\TEST.$DATA07.SAMDBCAT.FILES;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.INDEXES
\TEST.$DATA07.SAMDBCAT.INDEXES ;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.KEYS
\TEST.$DATA07.SAMDBCAT.KEYS;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.PARTNS
\TEST.$DATA07.SAMDBCAT.PARTNS;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.PROGRAMS
\TEST.$DATA07.SAMDBCAT.PROGRAMS;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.SYSCOLUMNS
```

```
\TEST.$DATA07.SAMDBCAT.ZVUCOL;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.SYSINDEXES
\TEST.$DATA07.SAMDBCAT.ZVUIX;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.SYSOBJECTS
\TEST.$DATA07.SAMDBCAT.ZVUOBJ;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.SYSPROTECTS
\TEST.$DATA07.SAMDBCAT.ZVUPROT;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.SYSTYPES
\TEST.$DATA07.SAMDBCAT.ZVUDT;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.SYSUSERS
\TEST.$DATA07.SAMDBCAT.ZVUUS;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.TABLES
\TEST.$DATA07.SAMDBCAT.TABLES;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.TRANSIDS
\TEST.$DATA07.SAMDBCAT.TRANSIDS;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.USAGES
\TEST.$DATA07.SAMDBCAT.USAGES;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.VERSIONS
\TEST.$DATA07.SAMDBCAT.VERSIONS;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.VIEWS
\TEST.$DATA07.SAMDBCAT.VIEWS;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.ZNUDT
\TEST.$DATA07.SAMDBCAT.ZNUDT;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.ZNUIX
\TEST.$DATA07.SAMDBCAT.ZNUIX;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.ZNUOBJ
\TEST.$DATA07.SAMDBCAT.ZNUOBJ;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.ZNUPCOL
\TEST.$DATA07.SAMDBCAT.ZNUPCOL;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.ZNUPROC
\TEST.$DATA07.SAMDBCAT.ZNUPROC;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.ZVUOCOL
\TEST.$DATA07.SAMDBCAT.ZVUOCOL;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.ZVUPCOL
\TEST.$DATA07.SAMDBCAT.ZVUPCOL;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.CUSTLIST
\TEST.$DATA07.SALES.CUSTLIST;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.CUSTOMER
\TEST.$DATA07.SALES.CUSTOMER;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.DEPT
\TEST.$DATA07.PERSNL.DEPT;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.EMPLIST
\TEST.$DATA07.PERSNL.EMPLIST;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.EMPLOYEE
\TEST.$DATA07.PERSNL.EMPLOYEE;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.JOB
\TEST.$DATA07.PERSNL.JOB;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.KEVIN
\TEST.$DATA07.SAMDBCAT.KEVIN;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.MGRLIST
\TEST.$DATA07.PERSNL.MGRLIST;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.ODETAIL
\TEST.$DATA07.SALES.ODETAIL;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.ORDERS
\TEST.$DATA07.SALES.ORDERS;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.ORDREP
\TEST.$DATA07.SALES.ORDREP;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.PARTLOC
\TEST.$DATA07.INVENT.PARTLOC;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.PARTS
\TEST.$DATA07.SALES.PARTS;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.PARTSUPP
\TEST.$DATA07.INVENT.PARTSUPP;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.SUPPLIER
\TEST.$DATA07.SALES.SUPPLIER;
```

```
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.VIEW207
\TEST.$DATA07.INVENT.VIEW207;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.VIEW207N
\TEST.$DATA07.INVENT.VIEW207N;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.VIEWCS
\TEST.$DATA07.INVENT.VIEWCS;
CREATE SQLMP ALIAS TEST_DATA07_SAMDBCAT.DBAGRP_PRSNL.VIEWCUST
\TEST.$DATA07.INVENT.VIEWCUST;
```

Adding User Alias Names To Safeguard

To preserve the NonStop ODBC Server user alias names, map the following user alias and Guardian user names to Safeguard aliases:

User Alias	Guardian User Name	Guardian User ID
DBO	SUPER.SUPER	255,255
DBAGRP_PERSNL	DBAGRP.PERSNL	100,100
ODBCUSR1	SQLUSR.USER1	50,1
ODBCUSR2	SQLUSR.USER2	50,2

The following commands add the same aliases in Safeguard (through Safecom) by using a simple security form:

```
Add alias DBO, SUPER.SUPER, password xxx
Add alias DBAGRP_PERSNL, DBAGRP.PERSNL, password xxx
Add alias ODBCUSR1, SQLUSR.USER1, password xxx
Add alias ODBCUSR2, SQLUSER.USER2, password xxx
```

where *xxx* is the password assigned by the system administrator to match the one used by the Guardian user ID.

For detailed information on how to add and configure system security, see the *Safeguard Reference Manual*.

Running and Configuring MXCS

To create an MXCS service configuration equivalent to the NonStop ODBC Server configuration, you must first start the association server using the following command (the example uses a different process name in addition to a different port number to let both products run in the same system):

```
MXOAS /name $mxoad, nowait, cpu 0, pri 140/ -PN 24000 -TCP $ZTC0
```

Using NSM/web or MXCI, add a data source (called NSOADMIN to match the same name, but not required) and specify the following values:

Data Source Name	NSOADMIN
Min Servers	1
Max Servers	5
Avail Servers	1
Connection timeout	System defined
Idle timeout	System defined
Resource accounting	None
Start mode	Manual

Client Data Source Configuration

On the client workstation, add a new data source and select the ODBC/MX driver name. When the wizard is displayed, enter the following information:

Data Source Name	**NSOADMIN**
ODBC IP address	TEST (DNS name or IP address)
Port number	24000 (same as in -PN parameter above)
Login timeout	System default
Connection timeout	System default
Query timeout	No timeout
Fetch buffer size	System default (is 8Kb)
Catalog	MASTER
Schema	DBO

In the NonStop ODBC Server environment, use NSOADMIN to match the server data source name and create a connection with that data source.

Assume that there is no need to modify the application and that existing applications can be executed by connecting to the newly added NSOADMIN data source.

B Using Microsoft Access with ODBC/MX

Microsoft Access applications interact in a special way with NonStop servers.

To enable Microsoft Access, use the NSM/web SET property sheet tab or MXCI to set the environment variables SQL_ATTR_ACCESS_VERSION, and SQL_ATTR_MAP_DATATYPE for the data source to be used by Microsoft Access. If these environment variables are not set, you may receive errors when linking tables or retrieving data from tables.

To enable and disable Read Only Access mode, set the environment variable SQL_ATTR_ACCESS_MODE.

SQL_ATTR_ACCESS_VERSION for Microsoft Access

- To set the attribute:
  - In the NSM/web SET tab for the data source, set this attribute to *1997* or *2000*. For Microsoft Access 2002 and later, set it to *2000*. See "Reconfiguring Server Data Sources" (page 29) for details about using NSM/web to change settings.
  - Using MXCI in MXCS mode, the syntax is:

    ```
    CS> add evar <dsname>.SQL_ATTR_MSACCESS_VERSION, type set, value
    '2000';
    ```
- The setting of this attribute defines this behavior:
  - Sets SQL_ATTR_ACCESS_MODE attribute to SQL_MODE_READ_WRITE. This setting takes precedence of any setting made by the client ODBC application.
  - SQLStatistics returns only unique indexes.

If this attribute is not set, a Microsoft Access application may fail to update, insert, or delete rows in linked tables and it may not allow updates to tables using the graphical record sets.

SQL_ATTR_MAP_DATATYPE for Microsoft Access

- To set the attribute:
  - In the NSM/web SET tab for the data source, set this attribute to BINGINT_TO_NUMERIC. This allows the ODBC driver to map the BIGINT column type to NUMERIC (19,0) for Microsoft applications that cannot process BIGINT.

    SQL_ATTR_MSACCESS_VERSION must be set to 2000 for this mapping to take effect.

    See "Reconfiguring Server Data Sources" (page 29) for details about using NSM/web to change settings.
  - Using MXCI in MXCS mode, the syntax is:

    ```
    CS> add evar <dsname>.SQL_ATTR_MAP_DATATYPE,type set, value
    'BIGINT_TO_NUMERIC';
    ```
- Setting the SQL_ATTR_MAP_DATATYPE attribute defines this behavior:
  - BIGINT data types are returned as NUMERIC (19,0).
  - UNSIGNED INT columns are returned as NUMERIC (19,0) for ODBC 2.0 applications.
  - Tables imported using the NUMERIC (19,0) data type cannot be exported again.

SQL_ATTR_ACCESS_MODE for Microsoft Access

You can set access mode on the server by defining this environment variable. If you are using the SQL_ATTR_MSACCESS_VERSION variable, you do not have to set this variable.

- To set the attribute:
  - In the NSM/web SET tab, the settings are SQL_MODE_READ_WRITE or SQL_MODE_READ_ONLY.

    See "Reconfiguring Server Data Sources" (page 29) for details about using NSM/web to change settings.
  - To set this attribute using MXCI in MXCS mode, the syntax is:

    ```
    CS> add evar <dsname>.SQL_ATTR_ACCESS_MODE,type set, value
    'SQL_MODE_READ_WRITE';
    ```

    or

    ```
    CS> add evar <dsname> .SQL_ATTR_ACCESS_MODE,type set, value
    'SQL_MODE_READ_ONLY';
    ```

The default setting is SQL_MODE_READ_WRITE and is overridden by the setting from ODBC when calling the SQLSetConnectAttr function.

If you have set this variable to SQL_MODE_READ_ONLY and ODBC does not override the setting, update, insert, and delete statements will fail with an error.

Glossary

association server. The HP NonStop MXCS component that starts and manages MXCS SQL servers and associates a client connection request with a specific MXCS SQL server in a data source.

client application. An application that uses a connectivity API to access a HP NonStop SQL database.

configuration server. The MXCS component that manages the MXCS configuration data. The association server communicates with the configuration server to retrieve the MXCS configuration values.

data source. TA logical name that defines the information required to access data. On the client side, it defines driver name, network address, and specific attributes such as catalog and schema names. On the server side, it defines startup values, number of servers in a pool, Guardian defines, SQL/MX control statements, and resource management policies.

driver. The client component that implements function calls allowing a client application to access a NonStop SQL database through an MXCS service.

Estimated Cost governing attribute. An attribute for an MXCS resource management policy. In the current release, the Estimated Cost governing attribute checks the elapsed time in seconds for a query to run against a limit value.

governing action. The action taken by MXCS when the limit is exceeded for a resource management policy attribute.

governing attribute. A resource management policy entity that is used for comparing an aspect of a SQL/MX query to a limit value. In the current release, MXCS supports the Estimated Cost governing attribute.

HP NonStop SQL/MX. The relational database system (RDMS) that is available on HP NonStop Kernel operating systems.

MXCS SQL server. The MXCS component that provides access to a NonStop SQL database through the NonStop SQL/MX engine.

resource governing policy. See resource management policy.

resource management policy limit. The maximum value for a governing attribute. If the limit is exceeded, MXCS takes the governing action specified for the governing attribute.

resource management policy. Also called resource governing policy. An attribute-value pair that associates a governing attribute with a test limit value and a governing action that MXCS takes if the limit value is exceeded.

service. The MXCS components running in an HP NonStop system, including the association server, configuration server, and MXCS SQL servers.

SQL. See HP NonStop SQL/MX database.

SQL/MX connectivity service. The HP NonStop server component that enables connectivity from client applications to the NonStop SQl database. The SQL/MX connectivity service consists of an association server, a configuration server, and an MXCS SQL server.

Index