

Machine Automation Controller

NJ/NX-series

Database Connection CPU Units

User's Manual

NX701-□□20

NX102-□□20

NJ501-□□20

NJ101-□□20

CPU Unit



NOTE

- All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, mechanical, electronic, photocopying, recording, or otherwise, without the prior written permission of OMRON.
- No patent liability is assumed with respect to the use of the information contained herein. Moreover, because OMRON is constantly striving to improve its high-quality products, the information contained in this manual is subject to change without notice.
- Every precaution has been taken in the preparation of this manual. Nevertheless, OMRON assumes no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information contained in this publication.

Trademarks

- Sysmac and SYSMAC are trademarks or registered trademarks of OMRON Corporation in Japan and other countries for OMRON factory automation products.
- Microsoft, Windows, Windows Vista, Excel, and SQL Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and other countries.
- EtherCAT® is registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.
- ODVA, CIP, CompoNet, DeviceNet, and EtherNet/IP are trademarks of ODVA.
- The SD and SDHC logos are trademarks of SD-3C, LLC. 
- Oracle, Java, and MySQL are registered trademarks of Oracle Corporation and/or its affiliates in the USA and other countries.
- IBM and DB2 are registered trademarks of International Business Machines Corporation in the USA and other countries.
- Firebird is a registered trademark of Firebird Foundation Incorporated.
- PostgreSQL is a registered trademark of PostgreSQL Global Development Group.

Other company names and product names in this document are the trademarks or registered trademarks of their respective companies.

Copyrights

- Microsoft product screen shots reprinted with permission from Microsoft Corporation.
- This product incorporates certain third party software. The license and copyright information associated with this software is available at http://www.fa.omron.co.jp/nj_info_e/.

Introduction

Thank you for purchasing an NJ/NX-series CPU Unit.

This manual contains information that is necessary to use the Database Connection Service with the NJ/NX-series CPU Unit. (Database may be referred to as DB hereinafter.) Please read this manual and make sure you understand the functionality and performance of the NJ/NX-series CPU Unit before you attempt to use it in a control system.

Keep this manual in a safe place where it will be available for reference during operation.

Intended Audience

This manual is intended for the following personnel, who must also have knowledge of electrical systems (an electrical engineer or the equivalent).

- Personnel in charge of introducing FA systems.
- Personnel in charge of designing FA systems.
- Personnel in charge of installing and maintaining FA systems.
- Personnel in charge of managing FA systems and facilities.

For programming, this manual is intended for personnel who understand the programming language specifications in international standard IEC 61131-3 or Japanese standard JIS B 3503.

Applicable Products

This manual covers the following products.

- NX-series Database Connection CPU Units
 - a) NX701-1720
 - b) NX701-1620
 - c) NX102-1220
 - d) NX102-1120
 - e) NX102-1020
 - f) NX102-9020
- NJ-series Database Connection CPU Units
 - a) NJ501-1520
 - b) NJ501-1420
 - c) NJ501-1320
 - d) NJ501-4320
 - e) NJ101-1020
 - f) NJ101-9020
- Sysmac Studio
 - a) SYSMAC-SE2□□□
 - NX701-□□20: Version 1.21 or higher
 - NX102-□□20: Version 1.24 or higher
 - NJ501-□□20 or NJ101-□□20: Version 1.14 or higher

Relevant Manuals

The following table provides the relevant manuals for the NJ-series CPU Units. Read all of the manuals that are relevant to your system configuration and application before you use the NJ-series CPU Unit.

Most operations are performed from Sysmac Studio Automation Software. Refer to the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for information on Sysmac Studio.

Purpose of use	Manual									
	Basic information					NJ/NX-series CPU Unit Motion Control User's Manual	NJ/NX-series Motion Control Instructions Reference Manual	NJ/NX-series CPU Unit Built-in EtherCAT Port User's Manual	NJ/NX-series CPU Unit Built-in EtherNet/IP Port User's Manual	NX-series CPU Unit FINS Functions User's Manual
	NJ-series CPU Unit Hardware User's Manual	NX-series NX102 CPU Unit Hardware User's Manual	NJ-series CPU Unit Hardware User's Manual	NJ/NX-series CPU Unit Software User's Manual	NJ/NX-series Instructions Reference Manual					
Introduction to NX701 CPU Units	○									
Introduction to NX102 CPU Units		○								
Introduction to NJ-series Controllers			○							
Setting devices and hardware										
Using motion control						○				
Using EtherCAT	○	○	○					○		
Using EtherNet/IP								○		
Using the database connection service										○
Software settings										
Using motion control						○				
Using EtherCAT								○		
Using EtherNet/IP				○				○		
Using FINS									○	
Using the database connection service										○
Writing the user program										
Using motion control						○	○			
Using EtherCAT								○		
Using EtherNet/IP								○		
Using FINS				○	○				○	
Using the database connection service										○
Programming error processing										○

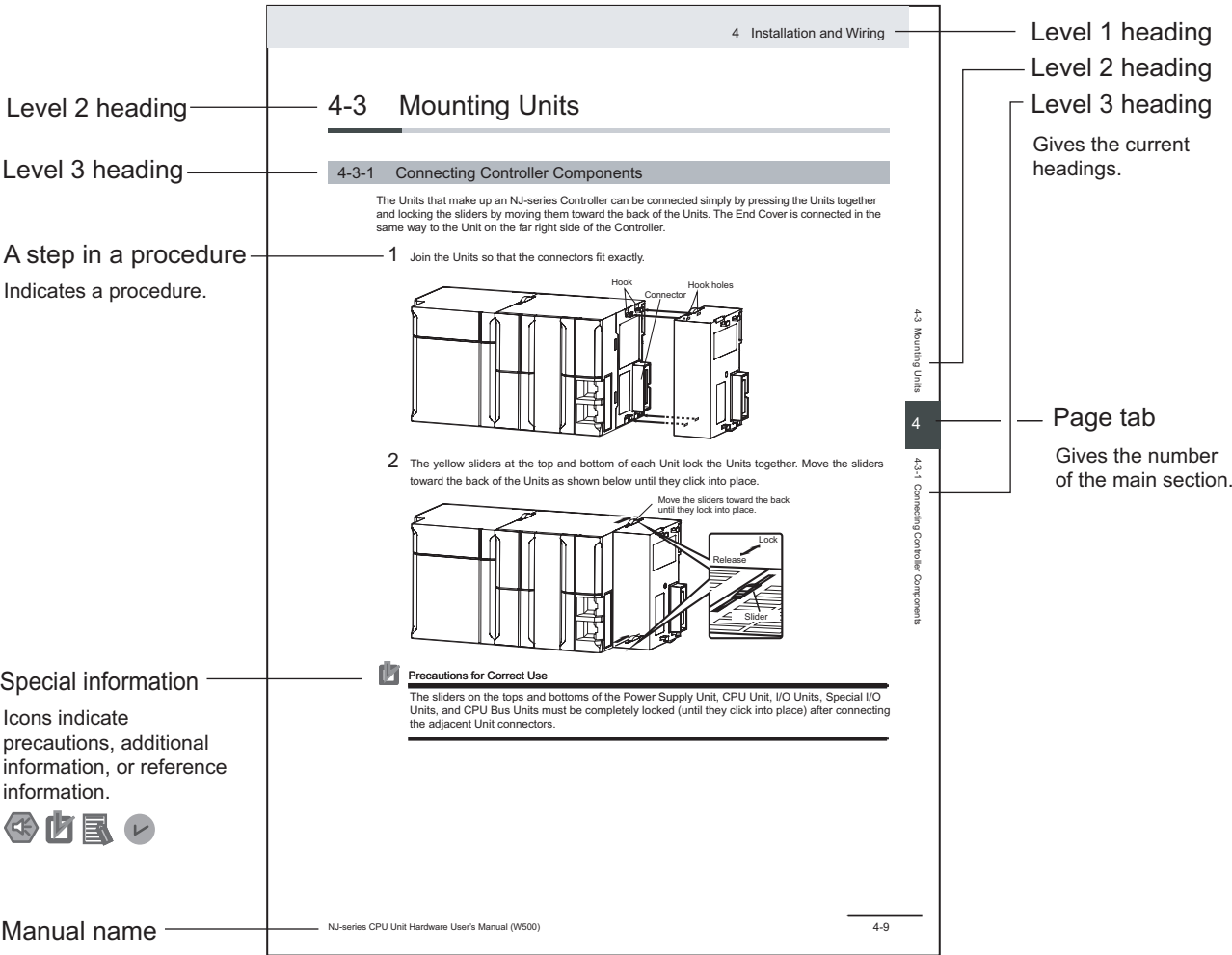
Purpose of use	Manual												
	Basic information					NJ/NX-series Motion Control User's Manual	NJ/NX-series Motion Control Instructions Reference Manual	NJ/NX-series CPU Unit Built-in EtherCAT Port User's Manual	NJ/NX-series CPU Unit Built-in EtherNet/IP Port User's Manual	NX-series CPU Unit FINS Functions User's Manual	NJ/NX-series Database Connection CPU Units User's Manual	NJ/NX-series CPU Unit Troubleshooting Manual	
	NX-series CPU Unit Hardware User's Manual	NX-series NX102 CPU Unit Hardware User's Manual	NJ-series CPU Unit Hardware User's Manual	NJ/NX-series CPU Unit Software User's Manual	NJ/NX-series Instructions Reference Manual								
Testing operation and debug- ging				○			○						
Using motion control													
Using EtherCAT									○				
Using EtherNet/IP										○			
Using FINS											○		
Using the database connec- tion service											○		
Learning about error manage- ment and corrections*1										△	△	○	
Maintenance	○	○	○										
Using motion control						○							
Using EtherCAT									○				
Using EtherNet/IP									○				

*1. Refer to the *NJ/NX-series Troubleshooting Manual*(Cat. No. W503) for the error management concepts and an overview of the error items. However, refer to the manuals that are indicated with triangles (△) for details on errors corresponding to the products with the manuals that are indicated with triangles (△).

Manual Structure

Page Structure

The following page structure is used in this manual.



This illustration is provided only as a sample. It may not literally appear in this manual.

Special Information

Special information in this manual is classified as follows:



Precautions for Safe Use

Precautions on what to do and what not to do to ensure safe usage of the product.



Precautions for Correct Use

Precautions on what to do and what not to do to ensure proper operation and performance.



Additional Information

Additional information to read as required.

This information is provided to increase understanding or make operation easier.

**Version Information**

Information on differences in specifications and functionality for Controller with different unit versions and for different versions of the Sysmac Studio is given.

Precaution on Terminology

In this manual, "download" refers to transferring data from Sysmac Studio to the physical Controller and "upload" refers to transferring data from the physical Controller to Sysmac Studio. For Sysmac Studio, "synchronization" is used to both "upload" and "download" data. Here, "synchronize" means to automatically compare the data for Sysmac Studio on the computer with the data in the physical Controller and transfer the data in the direction that is specified by the user.

Sections in this Manual

			1	I
1	Introduction to the DB Connection Service	I	2	
2	DB Connection settings		3	
3	Programming the DB Connection Function		4	
4	Basic Operations and Status Check		5	
5	Other Functions		6	
6	How to Use Operation Logs		7	
7	DB Connection Instruction		8	
8	Troubleshooting		A	
A	Appendix			

CONTENTS

Introduction	1
Intended Audience	1
Applicable Products	1
Relevant Manuals	2
Manual Structure	4
Page Structure	4
Special Information	4
Precaution on Terminology	5
Sections in this Manual	7
Terms and Conditions Agreement	14
Warranty, Limitations of Liability	14
Application Considerations	15
Disclaimers	15
Safety Precautions	17
Precautions for Safe Use	18
Precautions for Correct Use	19
Regulations and Standards	20
Versions	21
Version Types	21
Checking Versions	21
Unit Versions of CPU Units and Sysmac Studio Versions	23
Related Manuals	24
Terminology	26
Revision History	28

Section 1 Introduction to the DB Connection Service

1-1 Overview and Features	1-2
1-1-1 Overview	1-2
1-1-2 Features	1-3
1-2 DB Connection Service Specifications and System	1-5
1-2-1 DB Connection Service Specifications	1-5
1-2-2 DB Connection System	1-11
1-3 Operation Flow of the DB Connection Service	1-14

Section 2 DB Connection Settings

2-1 Starting Sysmac Studio and Creating a New Project	2-2

2-1-1	Starting Sysmac Studio	2-2
2-1-2	Creating a New Project	2-2
2-1-3	Setting the Built-in EtherNet/IP Port	2-3
2-1-4	Controller Setup	2-3
2-2	DB Connection Settings	2-5
2-2-1	DB Connection Service Settings	2-5
2-2-2	DB Connection Settings	2-7

Section 3 Programming the DB Connection Function

3-1	DB Access Procedure	3-2
3-2	Creating a Structure Data Type	3-3
3-2-1	Overview	3-3
3-2-2	Specifications of Structure Data Type for DB Access	3-3
3-2-3	How to Create a Structure Data Type for DB Access	3-13
3-3	Creating a DB Map Variable	3-17
3-3-1	DB Map Variables and DB Mapping	3-17
3-3-2	Registration and Attributes of DB Map Variables	3-18
3-3-3	Restrictions on DB Map Variables	3-19
3-4	Specifying the Table and Applying the Mapping	3-20
3-4-1	DB Mapping by Executing a Create DB Map Instruction	3-20
3-4-2	Clearing the Mapping of DB Map Variables	3-20
3-4-3	Restrictions on DB Mapping	3-20
3-5	Programming and Transfer	3-24
3-5-1	Programming the DB Connection Service	3-24
3-5-2	Displaying DB Connection Instructions on Sysmac Studio	3-25
3-5-3	DB Connection Instruction Set	3-25
3-5-4	System-defined Variables	3-26
3-5-5	Simulation Debugging of DB Connection Instructions	3-27
3-5-6	Transferring the DB Connection Settings and User Program	3-28
3-6	Debugging in Design, Startup, and Operation Phases	3-29
3-6-1	Design Phase	3-29
3-6-2	Startup Phase	3-29
3-6-3	Operation Phase	3-29

Section 4 Basic Operations and Status Check

4-1	Run Mode of DB Connection Service and Start/Stop Procedures	4-2
4-1-1	Run Mode of the DB Connection Service	4-2
4-1-2	How to Start/Stop the DB Connection Service	4-2
4-1-3	DB Connection Service is Stopped or Cannot be Started	4-4
4-1-4	Changing the Run Mode of the DB Connection Service	4-5
4-2	Establishing/Closing a DB Connection	4-6
4-3	Checking the Status of DB Connection Service and each DB Connection	4-7
4-3-1	Operation Status of the DB Connection Service	4-7
4-3-2	Checking the Status of the DB Connection Service	4-8
4-3-3	Connection Status of each DB Connection	4-10
4-3-4	Checking the Status of each DB Connection	4-11

Section 5 Other Functions

5-1	Examples of Using Functions	5-3
5-2	Spool Function	5-5
5-2-1	Overview	5-5

5-2-2	Spooling System	5-5
5-2-3	Applicable Instructions and Spooling Execution Conditions	5-5
5-2-4	Memory Area Used by the Spool Function	5-6
5-2-5	Spool Function Settings	5-8
5-2-6	How to Resend the SQL Statements Stored in the Spool Memory	5-8
5-2-7	Clearing the SQL Statements from the Spool Memory	5-9
5-2-8	Relationship with the DB Connection Instructions	5-11
5-2-9	How to Estimate the Number of SQL Statements that can be Spooled	5-13
5-3	Stored Procedure Call Function	5-16
5-3-1	Overview	5-16
5-3-2	Specifications of the Stored Procedure Call Function for Databases	5-17
5-3-3	How to Execute the Stored Procedure Call Function	5-19
5-3-4	Specifying the Table and Applying the Mapping	5-20
5-3-5	Errors during Stored Procedure Call	5-22
5-4	Batch Insert Function	5-24
5-4-1	Overview	5-24
5-4-2	How to Execute the Batch Insert Function	5-25
5-5	DB Connection Service Shutdown Function	5-26
5-5-1	Overview	5-26
5-5-2	Shutdown System	5-26
5-5-3	How to Execute the Shutdown Function	5-27
5-5-4	How to Check the Shutdown of the DB Connection Service	5-27
5-6	How to Prevent Losing SQL Statements at Power Interruption	5-28
5-6-1	Overview	5-28
5-6-2	Procedures	5-28
5-7	Timeout Monitoring Functions	5-33
5-7-1	Timeout Monitoring Functions	5-33
5-7-2	Login Timeout	5-33
5-7-3	Query Execution Timeout	5-34
5-7-4	Communications Timeout	5-34
5-7-5	Instruction Execution Timeout	5-35
5-7-6	Keep Alive Monitoring Time	5-35
5-8	Other Functions	5-36
5-8-1	Backup/Restore Function in the DB Connection Service	5-36
5-8-2	Operation Authority Verification in the DB Connection Service	5-37
5-8-3	Encrypted Communication	5-37

Section 6 How to Use Operation Logs

6-1	Operation Logs	6-2
6-2	Execution Log	6-3
6-2-1	Overview	6-3
6-2-2	Application Procedure	6-3
6-2-3	Setting the Execution Log	6-3
6-2-4	Checking the Execution Log	6-4
6-2-5	Execution Log File Specifications	6-4
6-3	Debug Log	6-15
6-3-1	Overview	6-15
6-3-2	Application Procedure	6-15
6-3-3	Set the Debug Log	6-15
6-3-4	Start Recording to the Debug Log	6-16
6-3-5	Stopping Recording to Debug Log	6-17
6-3-6	Checking the Debug Log	6-18
6-3-7	Debug Log File Specifications	6-18
6-4	SQL Execution Failure Log	6-29
6-4-1	Overview	6-29
6-4-2	Application Procedure	6-29
6-4-3	Setting the SQL Execution Failure Log	6-29

6-4-4	Checking the SQL Execution Failure Log	6-30
6-4-5	SQL Execution Failure Log File Specifications	6-30
6-5	SD Memory Card Operations	6-36
6-5-1	Saving Operation Log Files on SD Memory Card	6-36
6-5-2	Directory Used for DB Connection Service	6-36
6-5-3	Operation Log Operations in Replacing the SD Memory Card	6-37
6-5-4	Guidelines for SD Memory Card Replacement Time	6-37
6-5-5	Replacement Timing of SD Memory Card	6-38
6-6	Checking the Operation Logs	6-39
6-6-1	How to Check the Operation Logs	6-39
6-6-2	Checking the Log on the Operation Log Window in Sysmac Studio	6-39
6-6-3	Checking the Log with the SD Memory Card	6-41
6-6-4	Checking the Log by Transfer using FTP Client Software	6-41

Section 7 DB Connection Instructions

DB Connection Instructions and Variables	7-2
DB Connection Instruction Set	7-2
Variables Used in the DB Connection Instructions	7-2
DB_Connect (Establish DB Connection)	7-6
Variables	7-6
Related System-defined Variables	7-7
Related Error Codes	7-7
Function	7-7
Precautions for Correct Use	7-8
Sample Programming	7-9
DB_Close (Close DB Connection)	7-10
Variables	7-10
Related System-defined Variables	7-11
Related Error Codes	7-11
Function	7-11
Precautions for Correct Use	7-11
Sample Programming	7-12
DB_CreateMapping (Create DB Map)	7-13
Variables	7-13
Related System-defined Variables	7-14
Related Error Codes	7-14
Function	7-15
Precautions for Correct Use	7-15
Sample Programming	7-16
DB_Insert (Insert DB Record)	7-17
Variables	7-17
Related System-defined Variables	7-18
Related Error Codes	7-18
Function	7-19
Precautions for Correct Use	7-19
Sample Programming	7-20
DB_Update (Update DB Record)	7-21
Variables	7-21
Related System-defined Variables	7-22
Related Error Codes	7-22
Function	7-23
Precautions for Correct Use	7-24
Sample Programming	7-25
DB_Select (Retrieve DB Record)	7-39
Variables	7-39
Related System-defined Variables	7-40
Related Error Codes	7-40
Function	7-41

Precautions for Correct Use	7-43
Sample Programming	7-44
DB_Delete (Delete DB Record)	7-45
Variables	7-45
Related System-defined Variables	7-46
Related Error Codes	7-46
Function	7-47
Precautions for Correct Use	7-47
Sample Programming	7-48
DB_ControlService (Control DB Connection Service)	7-60
Variables	7-60
Related System-defined Variables	7-61
Related Error Codes	7-61
Function	7-62
Precautions for Correct Use	7-62
Sample Programming	7-63
DB_GetServiceStatus (Get DB Connection Service Status)	7-66
Variables	7-66
Related Error Codes	7-67
Function	7-67
Precautions for Correct Use	7-67
Sample Programming	7-67
DB_GetConnectionStatus (Get DB Connection Status)	7-71
Variables	7-71
Related System-defined Variables	7-72
Related Error Codes	7-72
Function	7-72
Precautions for Correct Use	7-72
Sample Programming	7-73
DB_ControlSpool (Resend/Clear Spool Data)	7-77
Variables	7-77
Related System-defined Variables	7-78
Related Error Codes	7-78
Function	7-78
Precautions for Correct Use	7-78
Sample Programming	7-79
DB_PutLog (Record Operation Log)	7-84
Variables	7-84
Related Error Codes	7-85
Function	7-85
Precautions for Correct Use	7-86
Sample Programming	7-86
DB_Shutdown (Shutdown DB Connection Service)	7-90
Variables	7-90
Related System-defined Variables	7-90
Related Error Codes	7-91
Function	7-91
Precautions for Correct Use	7-91
Sample Programming	7-91
DB_BatchInsert (DB Records Batch Insert)	7-94
Variables	7-94
Related System-defined Variables	7-95
Related Error Codes	7-95
Function	7-96
Precautions for Correct Use	7-96
Sample Programming	7-97
DB_AttachProcedure (Generate DB Stored Procedure Handle)	7-107
Variables	7-107
Related System-defined Variables	7-108
Related Error Codes	7-108

Function	7-109
Precautions for Correct Use	7-110
Sample Programming	7-111
DB_ExecuteProcedure (Execute DB Stored Procedure)	7-112
Variables	7-112
Related System-defined Variables	7-113
Related Error Codes	7-113
Function	7-114
Precautions for Correct Use	7-115
Sample Programming	7-115
DB_DetachProcedure (Release DB Stored Procedure Handle)	7-124
Variables	7-124
Related System-defined Variables	7-124
Related Error Codes	7-125
Function	7-125
Precautions for Correct Use	7-125
Sample Programming	7-126

Section 8 Troubleshooting

8-1 Overview of Errors	8-2
8-1-1 How to Check for Errors	8-2
8-1-2 Errors Related to the DB Connection Service	8-5
8-2 Troubleshooting	8-8
8-2-1 Error Table	8-8
8-2-2 Error Descriptions	8-16

Appendix

A-1 Task Design Procedure	A-2
A-1-1 Startup Time of DB Connection Service	A-2
A-1-2 Reference Values for Execution Time of DB Connection Instructions	A-4
A-1-3 How to Measure Execution Time of DB Connection Instructions	A-11
A-1-4 Guideline for System Service Execution Time Ratio	A-11
A-1-5 Checking the System Service Execution Time Ratio	A-13
A-2 Execution Time of DB Connection Instructions	A-15
A-2-1 Restrictions to Execution Time of DB Connection Instructions	A-15
A-2-2 Impact of Operation Log Recording on Execution Time of DB Connection Instructions	A-22
A-2-3 How to Measure DB Response Time	A-23
A-2-4 Ensuring Equipment Performance (Takt Time) by Monitoring Instruction Execution Timeout	A-23
A-3 Specifications	A-25
A-3-1 General Specifications	A-25
A-3-2 Performance Specifications	A-25
A-3-3 Function Specifications	A-25
A-4 Version Information	A-26
A-4-1 Unit Versions and Corresponding DB Connection Service Versions	A-26
A-4-2 DB Connection Functions that were Added or Changed for Each Unit Version	A-26
A-4-3 Unit Version, DB Connection Service Version, and Unit Version Set in the Sysmac Studio Project	A-27
A-4-4 DB Connection Service Version and Connection Database Types After Changing Devices	A-30
A-4-5 DB Connection Service Versions and Connection Database Types/Versions	A-31

Index

Terms and Conditions Agreement

Warranty, Limitations of Liability

Warranties

● Exclusive Warranty

Omron's exclusive warranty is that the Products will be free from defects in materials and workmanship for a period of twelve months from the date of sale by Omron (or such other period expressed in writing by Omron). Omron disclaims all other warranties, express or implied.

● Limitations

OMRON MAKES NO WARRANTY OR REPRESENTATION, EXPRESS OR IMPLIED, ABOUT NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE OF THE PRODUCTS. BUYER ACKNOWLEDGES THAT IT ALONE HAS DETERMINED THAT THE PRODUCTS WILL SUITABLY MEET THE REQUIREMENTS OF THEIR INTENDED USE.

Omron further disclaims all warranties and responsibility of any type for claims or expenses based on infringement by the Products or otherwise of any intellectual property right.

● Buyer Remedy

Omron's sole obligation hereunder shall be, at Omron's election, to (i) replace (in the form originally shipped with Buyer responsible for labor charges for removal or replacement thereof) the non-complying Product, (ii) repair the non-complying Product, or (iii) repay or credit Buyer an amount equal to the purchase price of the non-complying Product; provided that in no event shall Omron be responsible for warranty, repair, indemnity or any other claims or expenses regarding the Products unless Omron's analysis confirms that the Products were properly handled, stored, installed and maintained and not subject to contamination, abuse, misuse or inappropriate modification. Return of any Products by Buyer must be approved in writing by Omron before shipment. Omron Companies shall not be liable for the suitability or unsuitability or the results from the use of Products in combination with any electrical or electronic components, circuits, system assemblies or any other materials or substances or environments. Any advice, recommendations or information given orally or in writing, are not to be construed as an amendment or addition to the above warranty.

See <http://www.omron.com/global/> or contact your Omron representative for published information.

Limitation on Liability; Etc

OMRON COMPANIES SHALL NOT BE LIABLE FOR SPECIAL, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, LOSS OF PROFITS OR PRODUCTION OR COMMERCIAL LOSS IN ANY

WAY CONNECTED WITH THE PRODUCTS, WHETHER SUCH CLAIM IS BASED IN CONTRACT, WARRANTY, NEGLIGENCE OR STRICT LIABILITY.

Further, in no event shall liability of Omron Companies exceed the individual price of the Product on which liability is asserted.

Application Considerations

Suitability of Use

Omron Companies shall not be responsible for conformity with any standards, codes or regulations which apply to the combination of the Product in the Buyer's application or use of the Product. At Buyer's request, Omron will provide applicable third party certification documents identifying ratings and limitations of use which apply to the Product. This information by itself is not sufficient for a complete determination of the suitability of the Product in combination with the end product, machine, system, or other application or use. Buyer shall be solely responsible for determining appropriateness of the particular Product with respect to Buyer's application, product or system. Buyer shall take application responsibility in all cases.

NEVER USE THE PRODUCT FOR AN APPLICATION INVOLVING SERIOUS RISK TO LIFE OR PROPERTY OR IN LARGE QUANTITIES WITHOUT ENSURING THAT THE SYSTEM AS A WHOLE HAS BEEN DESIGNED TO ADDRESS THE RISKS, AND THAT THE OMRON PRODUCT(S) IS PROPERLY RATED AND INSTALLED FOR THE INTENDED USE WITHIN THE OVERALL EQUIPMENT OR SYSTEM.

Programmable Products

Omron Companies shall not be responsible for the user's programming of a programmable Product, or any consequence thereof.

Disclaimers

Performance Data

Data presented in Omron Company websites, catalogs and other materials is provided as a guide for the user in determining suitability and does not constitute a warranty. It may represent the result of Omron's test conditions, and the user must correlate it to actual application requirements. Actual performance is subject to the Omron's Warranty and Limitations of Liability.

Change in Specifications

Product specifications and accessories may be changed at any time based on improvements and other reasons. It is our practice to change part numbers when published ratings or features are changed, or when significant construction changes are made. However, some specifications of the Product may

be changed without any notice. When in doubt, special part numbers may be assigned to fix or establish key specifications for your application. Please consult with your Omron's representative at any time to confirm actual specifications of purchased Product.

Errors and Omissions

Information presented by Omron Companies has been checked and is believed to be accurate; however, no responsibility is assumed for clerical, typographical or proofreading errors or omissions.

Safety Precautions

Refer to the following manuals for safety precautions.

- *NX-series CPU Unit Hardware User's Manual (Cat. No. W535)*
- *NX-series NX102 CPU Unit Hardware User's Manual (Cat. No. W593)*
- *NJ-series CPU Unit Hardware User's Manual (Cat. No. W500)*

For safety precautions on NJ501-4320, please contact our sales representative and check with the product specification document or other documentation.

Precautions for Safe Use

Refer to the following manuals for precautions for safe use.

- *NX-series CPU Unit Hardware User's Manual (Cat. No. W535)*
- *NX-series NX102 CPU Unit Hardware User's Manual (Cat. No. W593)*
- *NJ-series CPU Unit Hardware User's Manual (Cat. No. W500)*

For precautions for safe use on NJ501-4320, please contact our sales representative and check with the product specification document or other documentation.

Precautions for Correct Use

This section describes the precautions for correct use in the DB Connection Service.

Refer to the following manuals for other precautions for correct use.

- *NX-series CPU Unit Hardware User's Manual (Cat. No. W535)*
- *NX-series NX102 CPU Unit Hardware User's Manual (Cat. No. W593)*
- *NJ-series CPU Unit Hardware User's Manual (Cat. No. W500)*

For precautions for correct use on NJ501-4320, please contact our sales representative and check with the product specification document or other documentation.

- For the NJ-series CPU Unit, when the Spool function is enabled, the DB Connection Service uses the following EM Banks according to the CPU Unit model. If the EM banks are used for processes other than the DB Connection Service, the Spool data in the EM Banks will be overwritten. Do not use the EM Banks that are used by the DB Connection Service for processes other than the DB Connection Service.

NJ501-□□20: EM Bank No. 9 to 18 (E9_00000 to E18_32767)

NJ101-□□20: EM Bank No. 1 to 3 (E1_00000 to E3_32767)

- Before you execute the stored procedure call function, make sure to verify the name of the stored procedure to execute, the processing details, and the argument values.

Regulations and Standards

Refer to the following manuals for regulations and standards.

- *NX-series CPU Unit Hardware User's Manual (W535)*
- *NX-series NX102 CPU Unit Hardware User's Manual (Cat. No. W593)*
- *NJ-series CPU Unit Hardware User's Manual (Cat. No. W500)*

Versions

Hardware revisions and unit versions are used to manage the hardware and software in NJ/NX-series Units and EtherCAT slaves. The hardware revision or unit version is updated each time there is a change in hardware or software specifications. Even when two Units or EtherCAT slaves have the same model number, they will have functional or performance differences if they have different hardware revisions or unit versions.

Version Types

There are two types of versions. One is unit version and the other is DB Connection Service version. These versions are managed independently. Therefore, only one of them may be upgraded.

● Unit Version

Hardware revisions and unit versions are used to manage the hardware and software in NJ/NX-series Units and EtherCAT slaves. The hardware revision or unit version is updated each time there is a change in hardware or software specifications. Even when two Units or EtherCAT slaves have the same model number, they will have functional or performance differences if they have different hardware revisions or unit versions.

● DB Connection Service version

This is the version of DB Connection Service implemented in the Database Connection CPU Units. The version is upgraded at every specification change in the DB Connection Service.

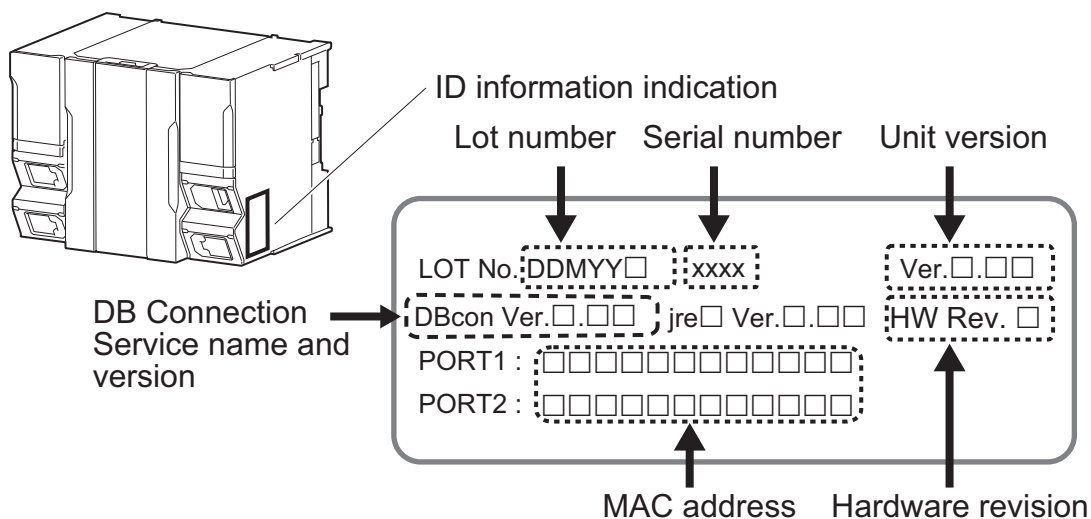
Checking Versions

You can check versions on the ID information indications or with Sysmac Studio.

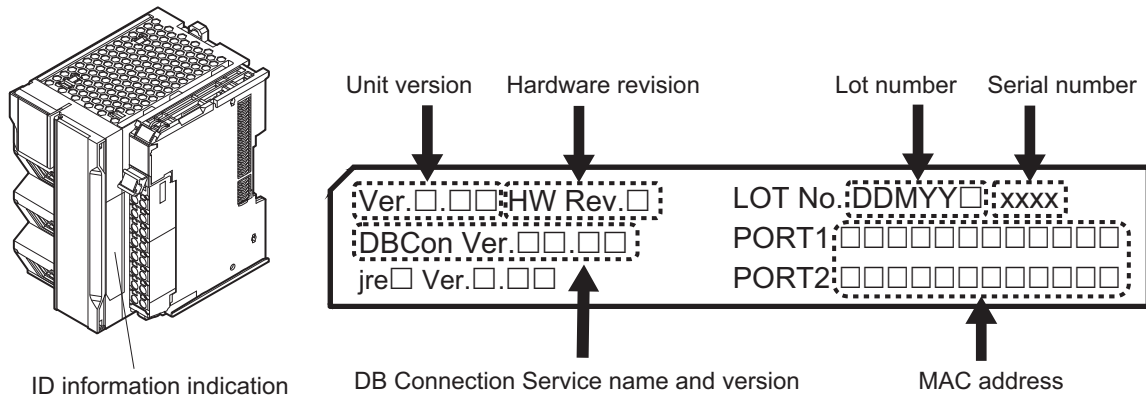
Checking Unit Versions on ID Information Label

The unit version is given on the ID information indication on the side of the product.

The ID information on NX-series NX701-□□20 CPU Units is shown below.

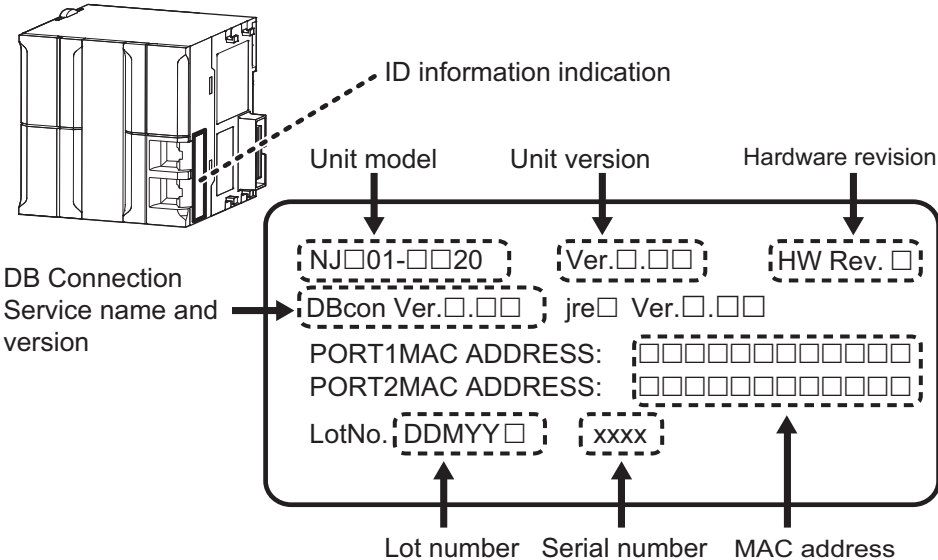


Note The hardware revision is not displayed for the Unit that the hardware revision is in blank.
The ID information on NX-series NX102-□□20 CPU Unit is shown below.



Note The hardware revision is not displayed for the Unit that the hardware revision is in blank.

The ID information on an NJ-series NJ□01-□□20 CPU Unit is shown below.



Note The hardware revision is not displayed for the Unit that the hardware revision is in blank.

Checking Unit Versions with Sysmac Studio

You can use the Sysmac Studio to check unit versions. The procedure is different for Units and for EtherCAT slaves.

● Checking the Unit Version of an NX-series CPU Unit

You can use the **Production Information** while the Sysmac Studio is online to check the unit version of a Unit. You can do this for the following Units.

Unit model	Available unit to check the unit version
NX701-□□□□	CPU Unit
NX102-□□□□	CPU Unit, NX Unit on CPU Rack

- 1 Right-click **CPU Rack** under **Configurations and Setup - CPU/Expansion Racks** in the Multi-view Explorer and select **Production Information**.
The **Production Information** Dialog Box is displayed.

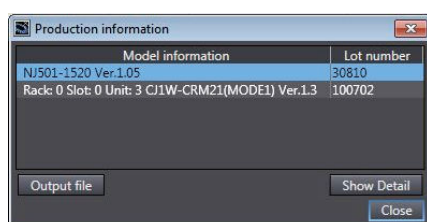
● Checking the Unit Version of an NJ-series CPU Unit

You can use the **Production Information** while the Sysmac Studio is online to check the unit version of a Unit. You can do this for the CPU Unit, CJ-series Special I/O Units, and CJ-series CPU Bus Units. You cannot check the unit versions of CJ-series Basic I/O Units with the Sysmac Studio. Use the following procedure to check the unit version.

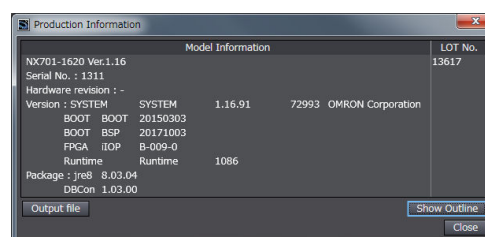
- 1** Double-click **CPU/Expansion Racks** under **Configurations and Setup** in the Multiview Explorer. Or, right-click **CPU/Expansion Racks** under **Configurations and Setup** and select **Edit** from the menu.
The Unit Editor is displayed.
- 2** Right-click any open space in the Unit Editor and select **Production Information**.
The **Production Information** Dialog Box is displayed.

● Changing Information Displayed in Production Information Dialog Box

- 1** Click the **Show Detail** or **Show Outline** Button at the lower right of the **Production Information** Dialog Box.
The view will change between the **Production Information** details and outline.



Outline View



Detail View

The information that is displayed is different for the Outline View and Detail View. The Detail View displays both the unit versions and DB Connection Service version. The Outline View displays only the unit versions.

Note The hardware revision is separated by "/" and displayed on the right of the hardware version. The hardware revision is not displayed for the Unit that the hardware revision is in blank.

Unit Versions of CPU Units and Sysmac Studio Versions

The functions that are supported depend on the unit version of the NJ/NX-series CPU Unit. The version of Sysmac Studio that supports the functions that were added for an upgrade is also required to use those functions.

Refer to *A-4 Version Information* on page A-26 for the relationship between the unit versions of the NJ/NX-series Database Connection CPU Units and the Sysmac Studio versions, and for the functions that are supported by each unit version.

Related Manuals

The following manuals are related to this manual. Use these manuals for reference.

Manual name	Cat. No.	Model numbers	Application	Description
NX-series CPU Unit Hardware User's Manual	W535	NX701-□□□□	Learning the basic specifications of the NX701 CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction to the entire NX701 system is provided along with the following information on the CPU Unit. • Features and system configuration • Introduction • Part names and functions • General specifications • Installation and wiring • Maintenance and inspection
NX-series NX102 CPU Unit Hardware User's Manual	W593	NX102-□□□□	Learning the basic specifications of the NX102 CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction to the entire NX102 system is provided along with the following information on the CPU Unit. • Features and system configuration • Introduction • Part names and functions • General specifications • Installation and wiring • Maintenance and inspection
NJ-series CPU Unit Hardware User's Manual	W500	NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning the basic specifications of the NJ-series CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction to the entire NJ-series system is provided along with the following information on the CPU Unit. • Features and system configuration • Introduction • Part names and functions • General specifications • Installation and wiring • Maintenance and inspection
NJ/NX-series CPU Unit Software User's Manual	W501	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning how to program and set up an NJ/NX-series CPU Unit. Mainly software information is provided.	The following information is provided on a Controller built with an NJ/NX-series CPU Unit. • CPU Unit operation • CPU Unit features • Initial settings • Programming based on IEC 61131-3 language specifications
NJ/NX-series Instructions Reference Manual	W502	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning detailed specifications on the basic instructions of an NJ/NX-series CPU Unit.	The instructions in the instruction set (IEC 61131-3 specifications) are described.
NJ/NX-series CPU Unit Motion Control User's Manual	W507	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning about motion control settings and programming concepts.	The settings and operation of the CPU Unit and programming concepts for motion control are described.

Manual name	Cat. No.	Model numbers	Application	Description
NJ/NX-series Motion Control Instructions Reference Manual	W508	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning about the specifications of the motion control instructions.	The motion control instructions are described.
NJ/NX-series CPU Unit Built-in EtherCAT® Port User's Manual	W505	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Using the built-in EtherCAT port on an NJ/NX-series CPU Unit.	Information on the built-in EtherCAT port is provided. This manual provides an introduction and provides information on the configuration, features, and setup.
NJ/NX-series CPU Unit Built-in EtherNet/IP™ Port User's Manual	W506	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Using the built-in EtherNet/IP port on an NJ/NX-series CPU Unit.	Information on the built-in EtherNet/IP port is provided. Information is provided on the basic setup, tag data links, and other features.
NJ/NX-series Database Connection CPU Units User's Manual	W527	NX701-□□20 NX102-□□20 NJ501-□□20 NJ101-□□20	Using the database connection service with NJ/NX-series Controllers.	Describes the database connection service.
NX-series CPU Unit FINS Function User's Manual	W596	NX701-□□20 NX102-□□□□	Using the FINS function of an NX-series CPU Unit.	Describes the FINS function of an NX-series CPU Unit.
NJ/NX-series Troubleshooting Manual	W503	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning about the errors that may be detected in an NJ/NX-series Controller.	Concepts on managing errors that may be detected in an NJ/NX-series Controller and information on individual errors are described.
Sysmac Studio Version 1 Operation Manual	W504	SYSMAC -SE2□□□	Learning about the operating procedures and functions of the Sysmac Studio.	Describes the operating procedures of the Sysmac Studio.

Terminology

Term	Description
CA	The institution that issues certificates. Certification Authority.
DB	Refers to a database in a server.
DB Connection	Refers to a virtual communication path established between CPU Unit and DB.
DB Connection function	Used to connect a CPU Unit to a DB. This function operates on a CPU Unit.
DB Connection Service	This service provides the DB Connection function to connect a CPU Unit to a DB. In the ID information indication on the side of the CPU Unit and in Sysmac Studio, this service is indicated as "DBCon".
DB Connection Service shutdown function	Used to shut down the DB Connection Service after automatically saving the Operation Log files into the SD Memory Card.
Run mode of the DB Connection Service	Used to switch whether to actually access the DB or to normally end the instructions without accessing the DB when DB Connection Instructions are executed.
DB Connection Instruction	Refers to special instructions for the DB Connection Service.
Structure data type for DB access	Refers to structure data type where all or some of the columns of a specified table are registered as structure members.
DB Map Variable	Refers to a variable that uses a structure data type for DB access as its data type.
DB mapping	Refers to the operation of associating each member of DB Map Variables with the columns of a table, or the arguments, return values, and result sets of a stored procedure.
DB Records Batch Insert instruction	Refers to the DB_BatchInsert instruction.
EM Area	Refers to Expansion DM Area used for CJ-series Units. The data in this area are retained even if the power supply to the CPU Unit is cycled (i.e. ON → OFF → ON) or the operating mode of the CPU Unit is changed (i.e. PROGRAM mode ↔ RUN mode).
SQL	Stands for Structured Query Language, which is one of the languages for DB processing such as data read/write.
SQL Execution Failure Log	One of the Operation Logs. This log is used to record execution failure of SQL statements in the DB.
Record processing	Refers to the process that manipulates DB records, such as record insertion, update, retrieval, and batch insertion.
SQL type	One of the input variables for the DB_CreateMapping instruction. It refers to a type of record processing for the variable to map, such as record insertion, update, retrieval, and batch insertion.
SQL statement	Refers to the statements that show a specific instruction used for DB operations such as data read/write.
Encrypted communication	A method of encrypted data communication between the controller and the database, which is designed to prevent sniffing and tampering by third parties.
Operation Log	Used to trace the operations of the DB Connection function on the CPU Unit. There are three types of Operation Logs; Execution Log, Debug Log, and SQL Execution Failure Log.
Column	One of the information layers of each DB. Refers to the columns of each table.
Server Certificate	It is an X.509 electronic certificate verifying a database. The database generates and manages the certificate along with its secret key. To use the encrypted communication function, it needs to be registered to the CPU Unit with the Sysmac Studio.
Execution Log	One of the Operation Logs. This log is used to record the executions of the DB Connection Service.

Term	Description
Stored procedure	Refers to a series of procedures for a database, which are stored in the DB management system. Complex SQL statement calls are logically grouped into a single processing unit, which can be easily called by the assigned name.
Stored function	In some of the database types, stored procedures that return a value are called stored functions and are distinguished from stored procedures.
Stored procedure call	Refers to the action of calling a stored procedure or a stored function, or the function itself.
Spool memory	Refers to the memory area for storing the SQL statements in the Spool function.
Spool function	Used to store some SQL statements for inserting records into the DB or updating the records in the DB that could not be executed due to a network failure.
Spool data	Refers to the SQL statements stored in the Spool memory.
Table	One of the information layers of each DB, which contains data.
Debug Log	One of the Operation Logs. This log is used for recording which SQL statements are executed, and parameters and execution result of each SQL statements.
Batch insert	Refers to the function or command that inserts multiple records at once.

Revision History

A manual revision code appears as a suffix to the catalog number on the front and back covers of the manual.

Cat. No.	W527-E1-12

↑ Revision code

Revision code	Date	Revised content
01	April 2013	Original production
02	August 2013	- Added description of the time specified for timeout of DB Connection Instructions. - page 5-12 page 7-19 page 7-24 page 7-43, page 7-47 - Corrected mistakes.
03	February 2014	Added description of the functions supported by the DB Connection Service version 1.01 or higher.
04	July 2014	- Added NJ501-4320 - Corrected mistakes.
05	November 2015	- Added NJ101-□□20 - Corrected mistakes.
06	December 2015	- Added description of the functions supported by the DB Connection Service version 1.02 or higher. - Corrected mistakes.
07	June 2016	Updated the EtherNet/IP logo.
08	January 2018	- Added NX701-□□20. - Added description of the functions supported by the DB Connection Service version 1.03 or higher.
09	June 2018	- Added NX102-□□20. - Added description of the functions supported by the DB Connection Service version 1.04 or higher.
10	July 2018	Corrected mistakes.
11	July 2019	- Added description of the functions supported by the DB Connection Service version 2.00 or higher for NX701-□□20 and NX102-□□20. - Corrected mistakes.
12	November 2019	Corrected mistakes.

Introduction to the DB Connection Service

This section provides an introduction to the DB Connection Service.

1-1	Overview and Features	1-2
1-1-1	Overview	1-2
1-1-2	Features	1-3
1-2	DB Connection Service Specifications and System	1-5
1-2-1	DB Connection Service Specifications	1-5
1-2-2	DB Connection System	1-11
1-3	Operation Flow of the DB Connection Service	1-14

1-1 Overview and Features

This section describes the overview and features of the DB Connection Service.

1-1-1 Overview

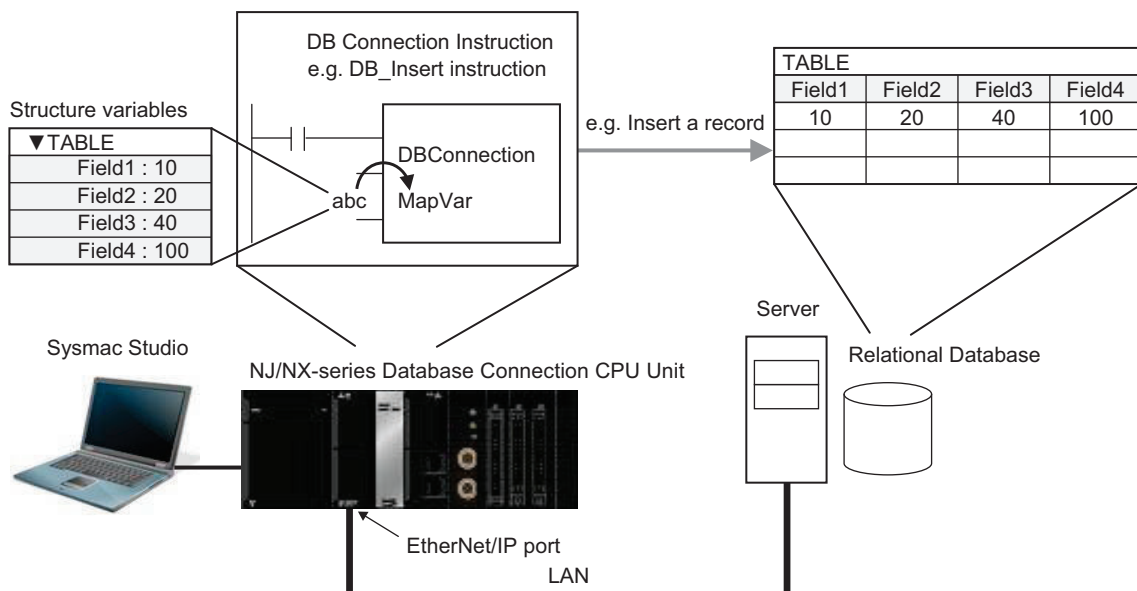
The SYSMAC NJ/NX-series Controllers are next-generation machine automation controllers that provide the functionality and high-speed performance that are required for machine control. They provide the safety, reliability, and maintainability that are required of industrial controllers.

The NJ/NX-series Controllers provide the functionality of previous OMRON PLCs, and they also provide the functionality that is required for motion control. Synchronized control of I/O devices on high-speed EtherCAT can be applied to safety devices, vision systems, motion equipment, discrete I/O, and more.

OMRON offers the new Sysmac Series of control devices designed with unified communications specifications and user interface specifications. The NJ/NX-series Machine Automation Controllers are part of the Sysmac Series. You can use them together with EtherCAT slaves, other Sysmac products, and the Sysmac Studio Automation Software to achieve optimum functionality and ease of operation.

With a system that is created from Sysmac products, you can connect components and operate the system through unified concepts and usability.

The DB Connection Service is a function to insert, update, retrieve, and delete records to/from a relational database (may be referred to as DB hereinafter) on a server connected to the built-in EtherNet/IP port of an NJ/NX-series CPU Unit by executing special instructions (called "DB Connection Instruction") on the NJ/NX-series CPU Unit.



- Oracle Database of Oracle Corporation, SQL Server of Microsoft Corporation, DB2 for Linux, UNIX and Windows of IBM Corporation, MySQL of Oracle Corporation, Firebird of Firebird Foundation Incorporated, and PostgreSQL of PostgreSQL Global Development Group are supported.*1

- It is possible to access more than one database^{*2} in one or more servers. You can realize flexible operations such as switching the database to access according to the specified data and SQL operations (such as INSERT/SELECT) and connecting to another database in a different server when a database cannot be connected, for example, due to a server problem.
- *1. The connectable databases are different for CPU Unit models. Refer to *1-2-1 DB Connection Service Specifications* on page 1-5 for the connectable databases.
- *2. Refer to *1-2-1 DB Connection Service Specifications* on page 1-5 for the number of databases that can be connected simultaneously.

1-1-2 Features

No Special Unit, Tool, nor Middleware Required

- No special Unit is required for the DB Connection function. You can use the NJ/NX-series CPU Units.
- No special tool is required for the DB Connection function. You can use Sysmac Studio.
- The server does not need any special middleware for connection to the NJ/NX-series CPU Units.

Easy Access to the DB

- The SQL operations such as INSERT and SELECT can be easily executed.
 - The advanced database functions, such as the stored procedure and batch insert functions for databases, can be executed easily as well, and data processing can be performed even faster.^{*1}
 - No special knowledge of SQL statements is required.
 - Variables for DB access can be defined just by creating a structure for the table that you want to access.
 - You can easily control the execution timing and prepare the write values because the SQL operations can be executed by special instructions.
 - More secure communication can be established by using the encrypted communication function.^{*1}
- *1. This function is available for the DB Connection Service version 2.00 or higher.

Recording of Operation Logs

- You can save the execution result logs of special instructions and processing (i.e. internal SQL statements) as a log file into the SD Memory Card mounted in the CPU Unit. Also, you can check the logs using Sysmac Studio or FTP client software.^{*1}
- *1. For saving the log files, an SD Memory Card is provided with each Database Connection CPU Unit. The SD Memory Card can be also used for any purposes other than DB Connection functions such as reading from and writing to the files in the SD Memory Card using instructions.

Fail-safe Design against Errors and Power Interruption

- You can spool the data (i.e. internal SQL statements) if the data cannot be sent due to an information exchange error with the DB, and execute the processing when the communications are recovered from the failure.

- You can automatically save the Operation Logs by shutting down the DB Connection Service when turning OFF the power supply to the CPU Unit.

Making a Library of DB Access Function

- You can provide and reuse the special instructions as a library file by describing each special instruction as a user-defined function block.

1-2 DB Connection Service Specifications and System

This section describes the specifications and system of the DB Connection Service.

1-2-1 DB Connection Service Specifications

This section describes the specifications of the DB Connection Service. Refer to *A-3 Specifications* on page A-25 for the general specifications, performance specifications, and function specifications of the Database Connection CPU Units.

Refer to *A-4 Version Information* on page A-26 for the information on version upgrades of the DB Connection Service.

NX-series CPU Unit

The following table shows the specifications of the DB Connection Service for NX-series CPU Units.

Specification item		CPU Unit model					
		NX701-1 720* ¹	NX701-1 620* ¹	NX102-1 220	NX102-1 120	NX102-1 020	NX102-9 020
Supported DB versions* ²	SQL Server by Microsoft	2012, 2014, 2016, 2017					
	Oracle Database by Oracle* ³	11g, 12c, 18c					
	DB2 for Linux, UNIX and Windows by IBM	9.7, 10.1, 10.5, 11.1					
	MySQL Community Edition by Oracle* ⁴	5.6, 5.7, 8.0					
	Firebird by Firebird Foundation	2.5					
	PostgreSQL by PostgreSQL Global Development Group* ⁵	9.4, 9.5, 9.6, 10					
Number of DB Connections (Number of databases that can be connected at the same time)		3* ⁶		2* ⁷			
Instruction	Supported operations	The following operations can be performed by executing DB Connection Instructions in the NJ/NX-series CPU Units. Insert Record (INSERT), Update Record (UPDATE), Retrieve Record (SELECT), Delete Record (DELETE), Execute Stored Procedure* ⁸ , and Execute Batch Insert* ⁸					
	Max. number of instructions for simultaneous execution	32					
	Max. number of columns in an INSERT operation	SQL Server: 1024 Oracle: 1000 DB2: 1000 MySQL: 1000 Firebird: 1000 PostgreSQL: 1000					

Specification item		CPU Unit model					
		NX701-1 720*1	NX701-1 620*1	NX102-1 220	NX102-1 120	NX102-1 020	NX102-9 020
	Max. number of columns in an UPDATE operation	SQL Server: 1024 Oracle: 1000 DB2: 1000 MySQL: 1000 Firebird: 1000 PostgreSQL: 1000					
	Max. number of columns in a SELECT operation	SQL Server: 1024 Oracle: 1000 DB2: 1000 MySQL: 1000 Firebird: 1000 PostgreSQL: 1000					
	Max. number of records in the output of a SELECT operation	65535 elements, 4 MB					
	Stored procedure call	Argument: Up to 256 variables Return value: One variable Result set: Supported Supported databases: SQL Server, Oracle, MySQL, PostgreSQL Spool function: Not supported					
	Batch insert execution	Supported data size: Less than 1,000 columns and upper limit of structure variable size or less Supported databases: SQL Server, Oracle, MySQL, PostgreSQL Spool function: Not supported					
	Max. number of DB Map Variables for which a mapping can be connected	SQL Server: 60 Oracle: 30 DB2: 30 MySQL: 30 Firebird: 15 PostgreSQL: 30 *9		SQL Server: 30 Oracle: 20 DB2: 20 MySQL: 20 Firebird: 15 PostgreSQL: 20 *7, *9			
Run Mode of the DB Connection Service		Operation Mode or Test Mode • Operation Mode: When each instruction is executed, the service actually accesses the DB. • Test Mode: When each instruction is executed, the service ends the instruction normally without accessing the DB actually.					
Spool Function		Used to store SQL statements when an error occurred and resend the statements when the communications are recovered from the error.					
	Spool capacity*10	2 MB		192 KB			
Operation Log function		The following three types of logs can be recorded. • Execution Log: Log for tracing the executions of the DB Connection Service. • Debug Log: Detailed log for SQL statement executions of the DB Connection Service. • SQL Execution Failure Log: Log for execution failures of SQL statements in the DB.					
DB Connection Service shutdown function		Used to shut down the DB Connection Service after automatically saving the Operation Log files into the SD Memory Card.					

Specification item		CPU Unit model					
		NX701-1 720*1	NX701-1 620*1	NX102-1 220	NX102-1 120	NX102-1 020	NX102-9 020
Communications port	Two ports supported	- Both of the two built-in EtherNet/IP ports are available. - Which of the ports will be used for each connection depends on the IP address settings. - Each of the two ports can be used for two separate connections simultaneously.		- Both of the two built-in EtherNet/IP ports are available. - Which of the ports will be used for each connection depends on the IP address settings. - Each of the two ports can be used for two separate connections simultaneously.*7			
Encrypted communication	Supported databases	SQL Server, Oracle, MySQL, PostgreSQL					
	TLS Ver.	TLS 1.2					

- *1. The CIP (Common Industrial Protocol) communications using the built-in EtherNet/IP port support the same functions as with the following CPU models. Therefore, when executing the EtherNet/IP tag data link function, please specify the following CPU models on Network Configurator. The following models are also displayed in Sysmac Gateway or CX-Compolet.

CPU Unit models used	Corresponding CPU Unit models
NX701-1720	NX701-1700
NX701-1620	NX701-1600

- *2. It is assumed that Windows Server OS and Windows Client OS are used to operate the DB. Confirm the operation requirements for each DB for details. Connections to the DB on the cloud are not supported. For details on the database versions that were supported in the past DB Connection Service versions, refer to *A-4-5 DB Connection Service Versions and Connection Database Types/Versions* on page A-31.
- *3. You cannot use Oracle 10g with the DB Connection Service version 2.00 or higher.
- *4. The supported storage engines of the DB are InnoDB and MyISAM.
- *5. When you connect the CPU Unit to PostgreSQL, make the following setting to set the locale of the PostgreSQL to C. Otherwise, the error messages are not correctly displayed.
- Change the value of `lc_messages` in the `postgresql.conf` file stored in the data folder under the installation folder of PostgreSQL and restart the PostgreSQL.
- `lc_messages = 'C'`
- *6. When two or more DB Connections are established, the operation cannot be guaranteed if you set different database types for the connections.
- *7. This function is available for the DB Connection Service version 2.00 or higher. For details on the versions of the DB Connection Service, refer to *Relationship between DB Connection Service Version and Unit Version Set in the Sysmac Studio Project* on page A-28.
- *8. This function is available for the DB Connection Service version 2.00 or higher. For details on the versions and instructions of the DB Connection Service, refer to *DB Connection Instruction Set* on page 7-2.
- *9. The maximum number of DB Map Variables that can be mapped are the total number of DB Map Variables that are used in INSERT/UPDATE, stored procedures, and batch insert. Note that if the number of DB Map Variables has not reached the upper limit, the total number of members of the structure definition used as a data type of DB Map Variables is 10,000 members max.
- *10. Refer to *5-2-9 How to Estimate the Number of SQL Statements that can be Spooled* on page 5-13 for the information.

NJ-series CPU Unit

The following table shows the specifications of the DB Connection Service for NJ-series CPU Unit.

Specification item		CPU Unit model					
		NJ501-152 0*1	NJ501-142 0*1	NJ501-132 0*1	NJ501-432 0*1	NJ101-102 0*1	NJ101-902 0*1
Supported DB ver- sions*2	SQL Server by Microsoft	2012, 2014					
	Oracle Data- base by Oracle	11g, 12c					
	DB2 for Linux, UNIX and Win- dows by IBM	9.7, 10.1, 10.5			Not sup- ported	9.7, 10.1, 10.5	
	MySQL Com- munity Edition by Oracle*3	5.6, 5.7					
	Firebird by Firebird Foun- dation	2.5			Not sup- ported	2.5	
	PostgreSQL by PostgreSQL Global Devel- opment Group*4	9.4			Not sup- ported	9.4	
Number of DB Connections (Num- ber of databases that can be con- nected at the same time)		3*5				1	

Specification item		CPU Unit model					
		NJ501-152 0*1	NJ501-142 0*1	NJ501-132 0*1	NJ501-432 0*1	NJ101-102 0*1	NJ101-902 0*1
Instruction	Supported operations	The following operations can be performed by executing DB Connection Instructions in the NJ/NX-series CPU Units. Insert Record (INSERT), Update Record (UPDATE), Retrieve Record (SELECT), and Delete Record (DELETE)					
	Max. number of instructions for simultaneous execution	32					
	Max. number of columns in an INSERT operation	SQL Server: 1024 Oracle: 1000 DB2: 1000 MySQL: 1000 Firebird: 1000 PostgreSQL: 1000					
	Max. number of columns in an UPDATE operation	SQL Server: 1024 Oracle: 1000 DB2: 1000 MySQL: 1000 Firebird: 1000 PostgreSQL: 1000					
	Max. number of columns in a SELECT operation	SQL Server: 1024 Oracle: 1000 DB2: 1000 MySQL: 1000 Firebird: 1000 PostgreSQL: 1000					
	Max. number of records in the output of a SELECT operation	65535 elements, 4 MB				65535 elements, 2 MBytes	
	Max. number of DB Map Variables for which a mapping can be connected *6	SQL Server: 60 Oracle: 30 DB2: 30 MySQL: 30 Firebird: 15 PostgreSQL: 30			SQL Server: 15 Oracle: 15 MySQL: 15	SQL Server: 15 Oracle: 15 DB2: 15 MySQL: 15 Firebird: 15 PostgreSQL: 15	
Run Mode of the DB Connection Service		Operation Mode or Test Mode • Operation Mode: When each instruction is executed, the service actually accesses the DB. • Test Mode: When each instruction is executed, the service ends the instruction normally without accessing the DB actually.					
Spool Function		Used to store SQL statements when an error occurred and resend the statements when the communications are recovered from the error.					
	Spool capacity*7	1 MB				192 KB	

Specification item	CPU Unit model					
	NJ501-152 0*1	NJ501-142 0*1	NJ501-132 0*1	NJ501-432 0*1	NJ101-102 0*1	NJ101-902 0*1
Operation Log function	The following three types of logs can be recorded. • Execution Log: Log for tracing the executions of the DB Connection Service. • Debug Log: Detailed log for SQL statement executions of the DB Connection Service. • SQL Execution Failure Log: Log for execution failures of SQL statements in the DB.					
DB Connection Service shutdown function	Used to shut down the DB Connection Service after automatically saving the Operation Log files into the SD Memory Card.					

- *1. The CIP (Common Industrial Protocol) communications using the built-in EtherNet/IP port support the same functions as with the following CPU models. Therefore, when executing the EtherNet/IP tag data link function, please specify the following CPU models on Network Configurator. The following models are also displayed in Sysmac Gateway or CX-Compolet.

CPU Unit models used	Corresponding CPU Unit models
NJ501-1520	NJ501-1500
NJ501-1420	NJ501-1400
NJ501-1320	NJ501-1300
NJ501-4320	NJ501-4300
NJ101-□□20	NJ101

- *2. It is assumed that Windows Server OS and Windows Client OS are used to operate the DB. Confirm the operation requirements for each DB for details. Connections to the DB on the cloud are not supported. For details on the database versions that were supported in the past DB Connection Service versions, refer to *A-4-5 DB Connection Service Versions and Connection Database Types/Versions* on page A-31.
- *3. The supported storage engines of the DB are InnoDB and MyISAM.
- *4. When you connect the CPU Unit to PostgreSQL, make the following setting to set the locale of the PostgreSQL to C. Otherwise, the error messages are not correctly displayed.
- Change the value of `lc_messages` in the `postgresql.conf` file stored in the data folder under the installation folder of PostgreSQL and restart the PostgreSQL.
- `lc_messages = 'C'`
- *5. When two or more DB Connections are established, the operation cannot be guaranteed if you set different database types for the connections.
- *6. The maximum number of DB Map Variables that can be mapped are the total number of DB Map Variables that are used in INSERT/UPDATE. Note that if the number of DB Map Variables has not reached the upper limit, the total number of members of the structure definition used as a data type of DB Map Variables is 10,000 members max.
- *7. Refer to *5-2-9 How to Estimate the Number of SQL Statements that can be Spooled* on page 5-13 for the information.

DB Versions That Extended Support for Databases Has Ended

The extended support for databases has ended for the following DB versions.

Please consider replacing the current database with a new version.

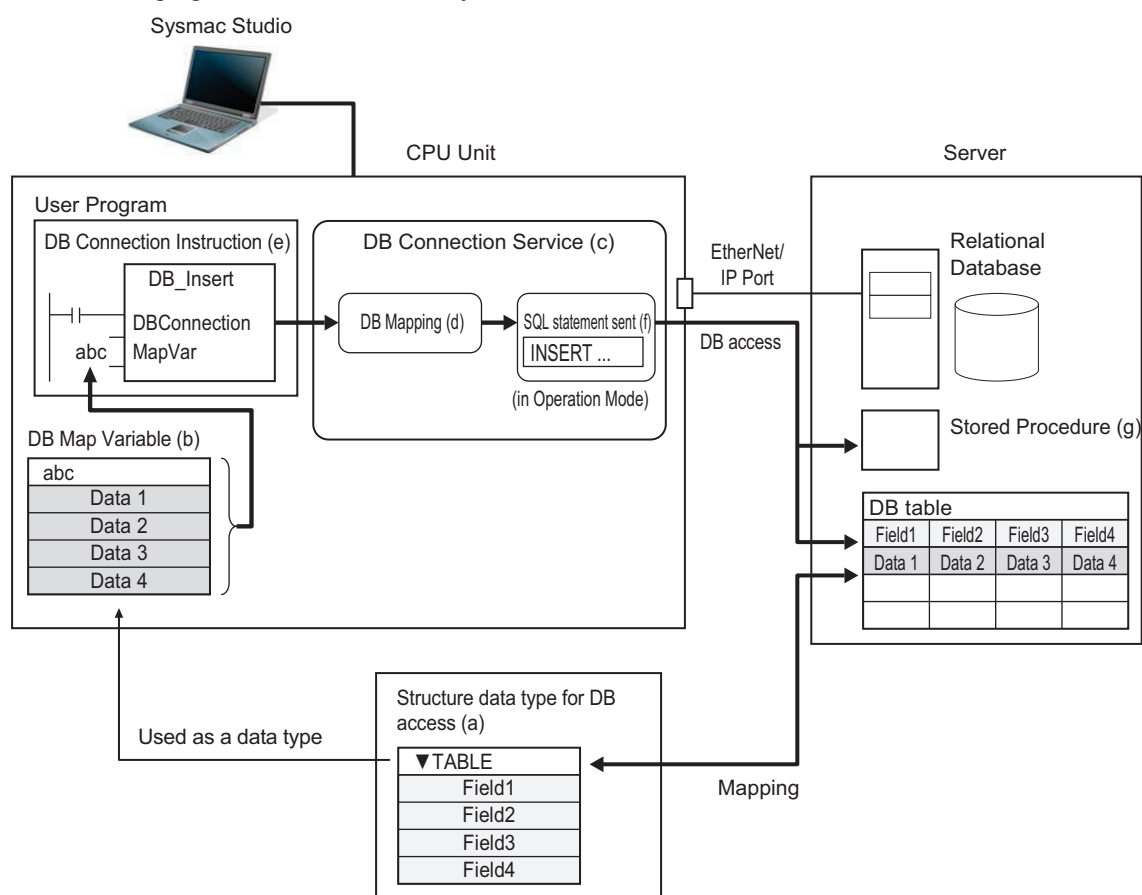
Specification item		CPU Unit model			
		NX701-1□20	NX102-□□20	NJ501-□□20	NJ101-□□20
Database versions	SQL Server by Microsoft	2008/2008R2			
	Oracle Database by Oracle	10g			
	DB2 for Linux, UNIX and Windows by IBM	9.5			
	MySQL Community Edition by Oracle	5.1/5.5			
	Firebird by Firebird Foundation	2.1			
	PostgreSQL by PostgreSQL Global Development Group	9.2/9.3			

1-2-2 DB Connection System

This section describes the basic and other systems of the DB Connection function.
Refer to 1-3 *Operation Flow of the DB Connection Service* on page 1-14 for the operation flow.

Basic System

The following figure shows the basic system of the DB Connection function.

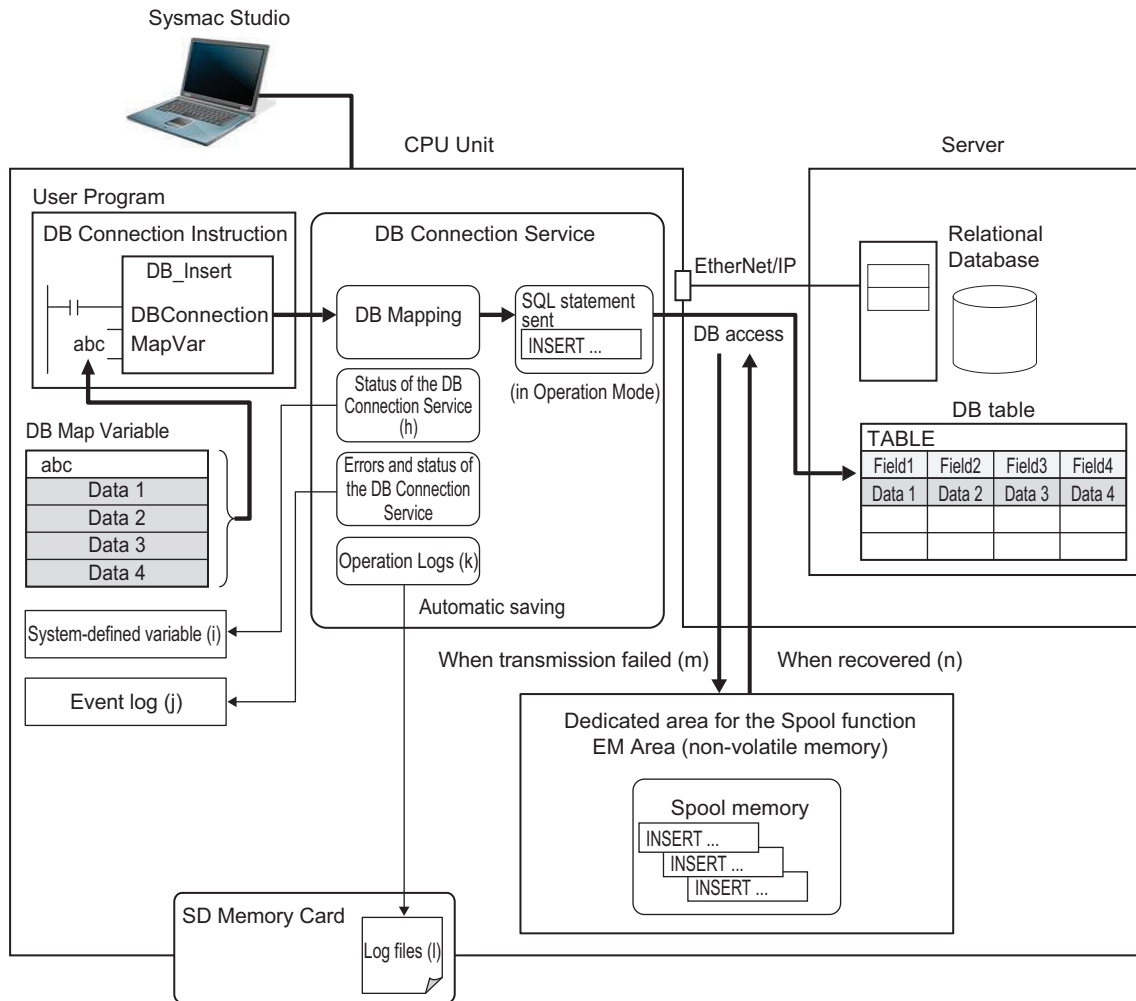


Basic System (The numbers show the processing order.)	Reference
1. Create a structure for NJ-series Controller that matches the column names in the DB table. ((a) in the above figure)	3-2 <i>Creating a Structure Data Type</i> on page 3-3
2. Create a variable called "DB Map Variable" using the structure created in Step 1. ((b) in the above figure)	3-3 <i>Creating a DB Map Variable</i> on page 3-17

Basic System (The numbers show the processing order.)	Reference
3. Start the DB Connection Service. ((c) in the above figure) Specify the Run mode of the DB Connection Service according to the following conditions. • When the DB is connected: Select the Operation Mode • When the DB does not exist or not connected: Select the Test Mode.	<i>4-1 Run Mode of DB Connection Service and Start/Stop Procedures on page 4-2</i>
4. Use a DB_Connect instruction to establish a DB Connection. This checks the IP address or name of the server and log on credentials.	<i>4-2 Establishing/Closing a DB Connection on page 4-6</i>
5. Use a DB_CreateMapping instruction to connect to a table using the DB Map Variable and apply the mapping. (called "DB mapping"). ((d) in the above figure)	<i>3-4 Specifying the Table and Applying the Mapping on page 3-20</i>
6. Specify the DB Map Variable and execute the following DB Connection instructions. ((e) in the above figure) • Insert Record • Records Batch Insert • Update Record • Retrieve Record • Stored Procedure When the DB Connection Service is set to the Operation Mode, the SQL statements are sent. ((f) in the above figure)	<i>3-5 Programming and Transfer on page 3-24</i>
7. When the stored procedure function is used, the stored procedure is executed in the database. ((g) in the above figure)	<i>5-3 Stored Procedure Call Function on page 5-16</i>

Other Systems

The following figure shows the other systems of the DB Connection function.

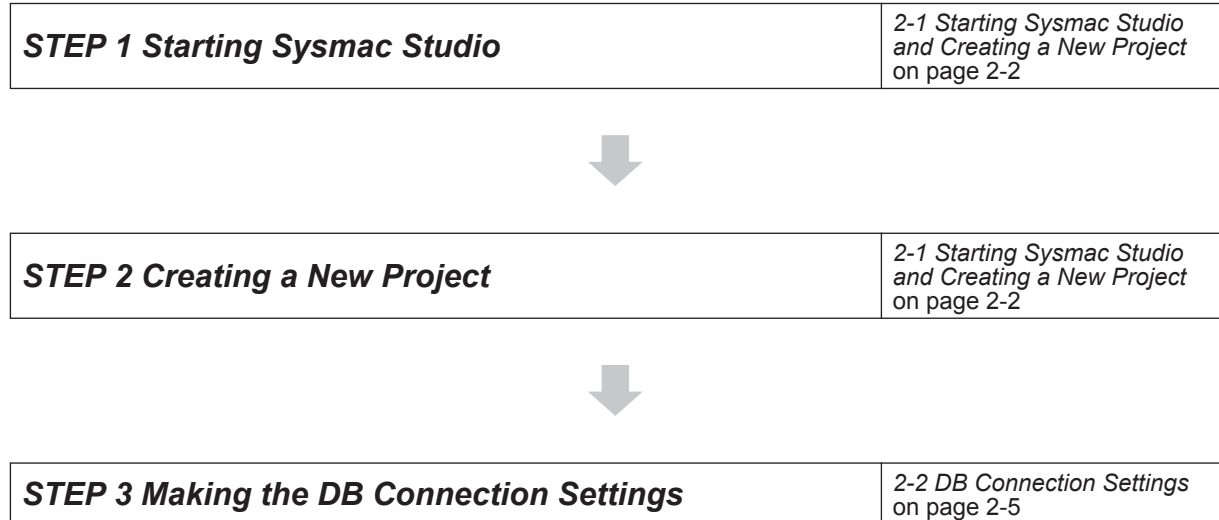


Other Systems	Reference
You can check the status of the DB Connection Service and each DB Connection ((h) in the above figure) with the DB_GetServiceStatus (Get DB Connection Service Status) instruction, DB_GetConnectionStatus (Get DB Connection Status) instruction, or a system-defined variable ((i) in the above figure).	Section 4 Basic Operations and Status Check on page 4-1
Errors and status of the DB Connection Service are stored as an event log. ((j) in the above figure)	Section 8 Troubleshooting on page 8-1
The logs of tracing the operations of the DB Connection Service on the CPU Unit (called "Operation Logs") ((k) in the above figure) are saved as a log file ((l) in the above figure) into the SD Memory Card mounted in the CPU Unit.	Section 6 How to Use Operation Logs on page 6-1
- When transmission of an SQL statement failed, the SQL statement is automatically saved into the dedicated area for the Spool function for an NX-series Controller and the EM Area of the memory for CJ-series Units for an NJ-series Controller. ((m) in the above figure) - When the communications are recovered, the stored SQL statement is resent automatically or by executing an instruction. ((n) in the above figure)	5-2 Spool Function on page 5-5

1-3 Operation Flow of the DB Connection Service

This section gives the basic operation flow.

The DB Connection Service is basically used according to the following flow.



Make a setting for the entire DB Connection Service and each DB Connection. Also, perform a communications test between Sysmac Studio and the DB as necessary.

1. Setting of the entire DB Connection Service:
 Double-click **DB Connection Service Settings** under **Configurations and Setup - Host Connection Settings - DB Connection** in the Multiview Explorer and set the following in the **Service Settings**.
 Service Start, Execution Log, Debug Log, and SQL Execution Failure Log settings
2. Setting of each DB Connection:
 Right-click **DB Connection Settings** under **Configurations and Setup - Host Connection Settings - DB Connection** in the Multiview Explorer and add up to each DB Connection.*¹ Then, set the following for each DB Connection.
 - Database type
 - IP address (IP address of the server)
 - Database name (Database name in the server)
 - User name, password, etc.
 - Spool Settings

*1. Refer to 1-2-1 DB Connection Service Specifications on page 1-5 for the number of DB Connections for each model.
3. Communications test from Sysmac Studio to the DB (only when necessary):
 Double-click a DB Connection under **Configurations and Setup - Host Connection Settings - DB Connection - DB Connection Settings** and click the **Communications Test** Button under the **DB Communications Test** in the Connection Settings.



STEP 4 Creating a Structure for DB Access

3-2 Creating a Structure Data Type on page 3-3

Create a structure data type for DB access. The structure members must satisfy the following conditions.

- Member names are the same as corresponding column name of the table to access.
- Members' data types match the data type of corresponding column of the table to access.

**STEP 5 Creating a Variable Using above Structure**

3-3 Creating a DB Map Variable on page 3-17

Create a variable called "DB Map Variable" using the structure data type created in STEP 4.

**STEP 6 Programming using DB Connection Instructions**3-4 Specifying the Table and Applying the Mapping on page 3-20
3-5 Programming and Transfer on page 3-24

1. Initial Processing
 - a) Write a DB_ControlService (Control DB Connection Service) instruction.
(This instruction is not required if you set the DB Connection Service to auto start in the DB Connection Settings.)
 - b) Write a DB_Connect (Establish DB Connection) instruction or a DB_AttachProcedure (Generate DB Stored Procedure Handle) instruction.
 - c) Write a DB_CreateMapping (Create DB Map) instruction.
The DB Map Variable is mapped with the columns of the table to access and registered as a variable subject to the record processing.
 2. Processing during Operation^{*1}
 - a) Write record processing and stored procedure instructions, etc.
 3. End Processing
 - a) Write a DB_Close (Close DB Connection) instruction.
 4. Power OFF Processing^{*2}
 - a) Write a DB_Shutdown (Shutdown DB Connection Service) instruction.
- ^{*1}. When you continuously execute instructions such as record processing and stored procedure instructions, repeat only the step (2) Processing during Operation.
- ^{*2}. Be sure to execute a DB_Shutdown (Shutdown DB Connection Service) instruction before you turn OFF the power supply to the system.
If the power supply is turned OFF without executing a DB_Shutdown (Shutdown DB Connection Service) instruction, the Operation Log file may be corrupted or its contents may be lost.

**STEP 7 Transferring a Project to the CPU Unit**

3-5 Programming and Transfer on page 3-24



STEP 8 Starting the DB Connection Service

Section 4 Basic Operations and Status Check on page 4-1

Use any of the following methods to start the DB Connection Service.

- Automatically start the service when the operating mode of the CPU Unit is changed from PRO-GRAM mode to RUN mode.
- Right-click **DB Connection Service Settings** under **Configurations and Setup - Host Connection Settings - DB Connection** in the Multiview Explorer and select **Online Settings** from the menu. Then, click the **Start (Test Mode)** or **Start (Operation Mode)** Button.
- Execute a DB_ControlService (Control DB Connection Service) instruction.

Specify the following Run mode when starting the DB Connection Service.

- When the specified DB does not exist in the server or when the DB exists but not connected: Specify the "Test Mode".
- When the specified DB is connected: Specify the "Operation Mode".

**STEP 9 Executing DB Connection Instructions**

*3-5-3 DB Connection Instruction Set on page 3-25
Section 7 DB Connection Instructions on page 7-1*

Confirm that the operation status of the DB Connection Service is "Running" with the `_DBC_Status.Run` system-defined variable (Running flag of the DB Connection Service) and then execute the DB Connection Instructions.

**STEP 10 Debugging the DB Connection Instructions**

3-6 Debugging in Design, Startup, and Operation Phases on page 3-29

**STEP 11 Checking the Status with Sysmac Studio**

Section 7 DB Connection Instructions on page 7-1

You can check the status of the entire DB Connection Service and the connection status of each DB Connection.

- Status of the entire DB Connection Service:
Right-click **DB Connection Service Settings** under **Configurations and Setup - Host Connection Settings - DB Connection** in the Multiview Explorer and select **Monitor DB Connection Service** from the menu. Then, check the status of the entire DB Connection Service on the monitor.
- Connection status of each DB Connection:
Right-click **DB Connection Settings** under **Configurations and Setup - Host Connection Settings - DB Connection** in the Multiview Explorer and select **Connection Monitor Table** from the menu. You can check the connection status of each DB Connection.



STEP 12 Checking the Operation Logs

Section 6 How to Use Operation Logs on page 6-1

You can check the following Operation Logs for tracing the operations of the DB Connection Service on the CPU Unit.

- Execution Log

This log is used to trace the executions of the DB Connection Service. Logging is kept while the DB Connection Service is running.

1. Right-click **DB Connection** under **Configurations and Setup - Host Connection Settings** and select **Show Operation Logs** from the menu and click the **Execution Log** Tab.

- Debug Log

This log is used for tracing which SQL statements were executed and parameters and execution result of each SQL statement.

1. Right-click **DB Connection** under **Configurations and Setup - Host Connection Settings** and select **Show Operation Logs** from the menu and click the **Debug Log** Tab.

- SQL Execution Failure Log

This log is recorded when an SQL execution failed in the DB.

1. Right-click **DB Connection** under **Configurations and Setup - Host Connection Settings** and select **Show Operation Logs** from the menu and click the **SQL Execution Failure Log** Tab.



STEP 13 Checking the Event Log

Section 8 Troubleshooting on page 8-1

DB Connection Settings

This section describes how to make the initial DB Connection settings for using the DB Connection Service.

2-1	Starting Sysmac Studio and Creating a New Project	2-2
2-1-1	Starting Sysmac Studio	2-2
2-1-2	Creating a New Project	2-2
2-1-3	Setting the Built-in EtherNet/IP Port	2-3
2-1-4	Controller Setup	2-3
2-2	DB Connection Settings	2-5
2-2-1	DB Connection Service Settings	2-5
2-2-2	DB Connection Settings	2-7

2-1 Starting Sysmac Studio and Creating a New Project

This section describes how to start Sysmac Studio and create a new project when using the DB Connection function.

Refer to the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for detailed operations.

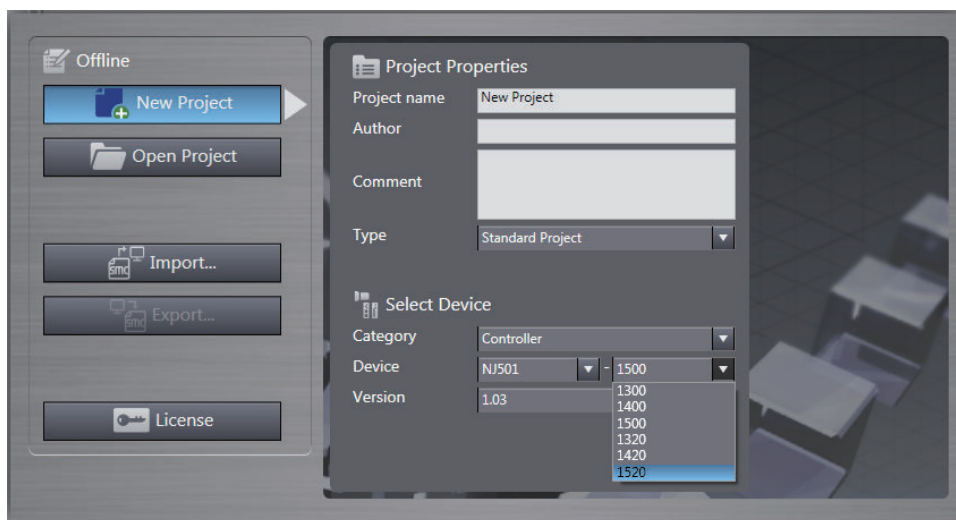
Refer to *A-4 Version Information* on page A-26 for correspondence between CPU Unit and DB Connection Service versions and between CPU Unit and Sysmac Studio versions.

2-1-1 Starting Sysmac Studio

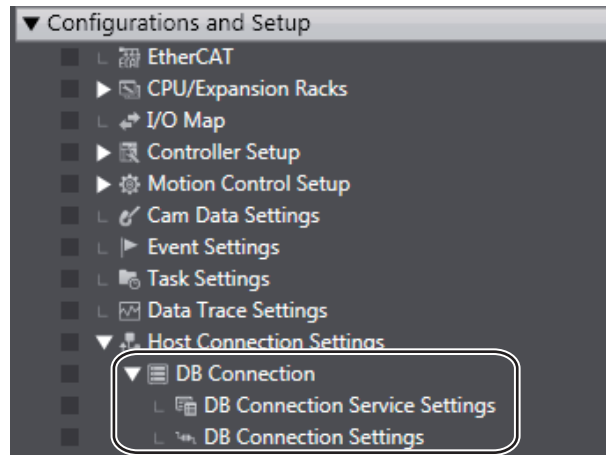
- 1 Install the following Sysmac Studio.
 - NX701-□□20: Version 1.21 or higher
 - NX102-□□20: Version 1.24 or higher
 - NJ501-□□20 or NJ101-□□20: Version 1.14 or higher
- 2 Start Sysmac Studio.

2-1-2 Creating a New Project

- 1 Select one of the following devices in the **Device Field** of the **Select Device Area**.
NX701: 1720 or 1620
NX102: 1220, 1120, 1020, or 9020
NJ501: 1520, 1420, 1320, or 4320
NJ101: 1020 or 9020



- 2 Click the **Create** Button.
DB Connection is displayed under **Host Connection Settings** in the Multiview Explorer.



2-1-3 Setting the Built-in EtherNet/IP Port

- 1 Right-click **Built-in EtherNet/IP Port Settings** under **Configurations and Setup - Controller Setup** in the Multiview Explorer and select **Edit** from the menu.
- 2 Make the TCP/IP, LINK, FTP, NTP, SNMP, SNMP Trap, and FINS settings in the Built-in EtherNet/IP Port Settings Tab Page.

Refer to the *NJ/NX-series CPU Unit Built-in EtherNet/IP Port User's Manual (Cat. No. W506)* for details on the settings.

When you use the DB Connection Service, the following port numbers are used in the built-in EtherNet/IP port. Do not set them for the other purposes.

Refer to the *NJ/NX-series CPU Unit Built-in EtherNet/IP Port User's Manual (Cat. No. W506)* for the port numbers commonly used in the NX701-□□□□, NX102-□□□□, NJ501-□□□□, and NJ101-□□□□ CPU Units.

Application	UDP	TCP
System-used	---	9800 to 9819

2-1-4 Controller Setup

Use Sysmac Studio to make the operation settings of the Controller.

Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for detailed settings that are not described below.

Operation Settings

- 1 Right-click **Operation Settings** under **Configurations and Setup - Controller Setup** in the Multiview Explorer and select **Edit** from the menu.

● Basic Settings

The Basic Settings are functions supported by the CPU Unit, such as the definitions of operations when the power is turned ON or when the operating mode changes.

Category	Item	Description	Value	Default	Update timing	Changes in RUN mode
Operation Settings	Start delay time at startup	Sets the time to perform system services with priority during startup after the power supply is turned ON.*1	0 to 10 s	0 s	When down-loading to CPU Unit	Not allowed

*1. The startup time of the DB Connection Service can be reduced with this setting. Set the value to "10" if you give priority to system services. Otherwise, set the value to "0".

If you set the value to "10", after the power supply is turned ON, the CPU Unit gives priority to the system services for approximately 10 seconds during startup before the Unit changes the "startup state" to the "normal operation state". The time until the DB Connection Service becomes available (i.e., the `_DBC_Status.Run` system-defined variable changes to True) can be reduced by performing a part of processing of the system services with priority during "startup".

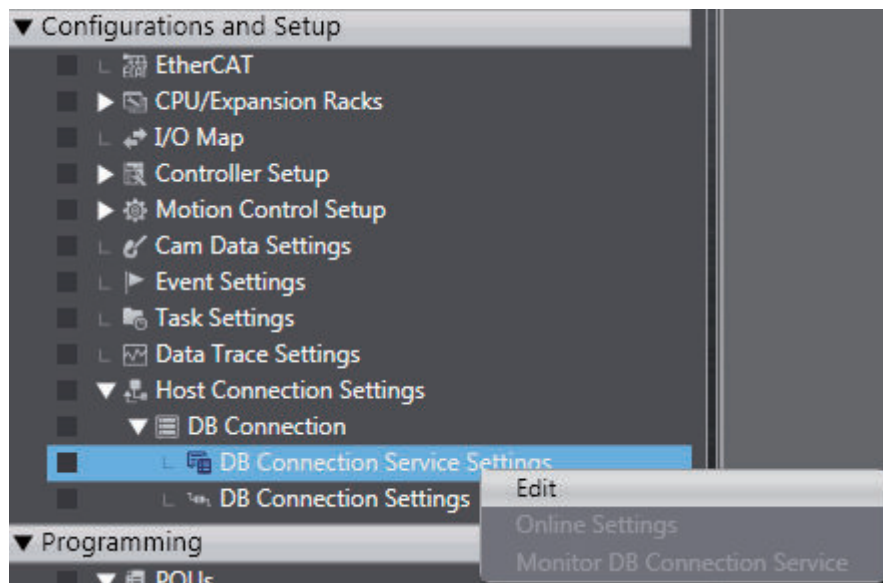
If you specify the value between "1 and 10", the time until the CPU Unit changes the state to the "normal operation state" is increased because the Unit gives priority to the system services for the specified time.

2-2 DB Connection Settings

You need to make the initial DB Connection settings before executing the DB Connection Service. Please make the settings of the entire DB Connection Service and each DB Connection. This section describes the DB Connection Service settings and DB Connection settings.

2-2-1 DB Connection Service Settings

Right-click **DB Connection Service Settings** under **Configurations and Setup - Host Connection Settings - DB Connection** in the Multiview Explorer and select **Edit** from the menu.

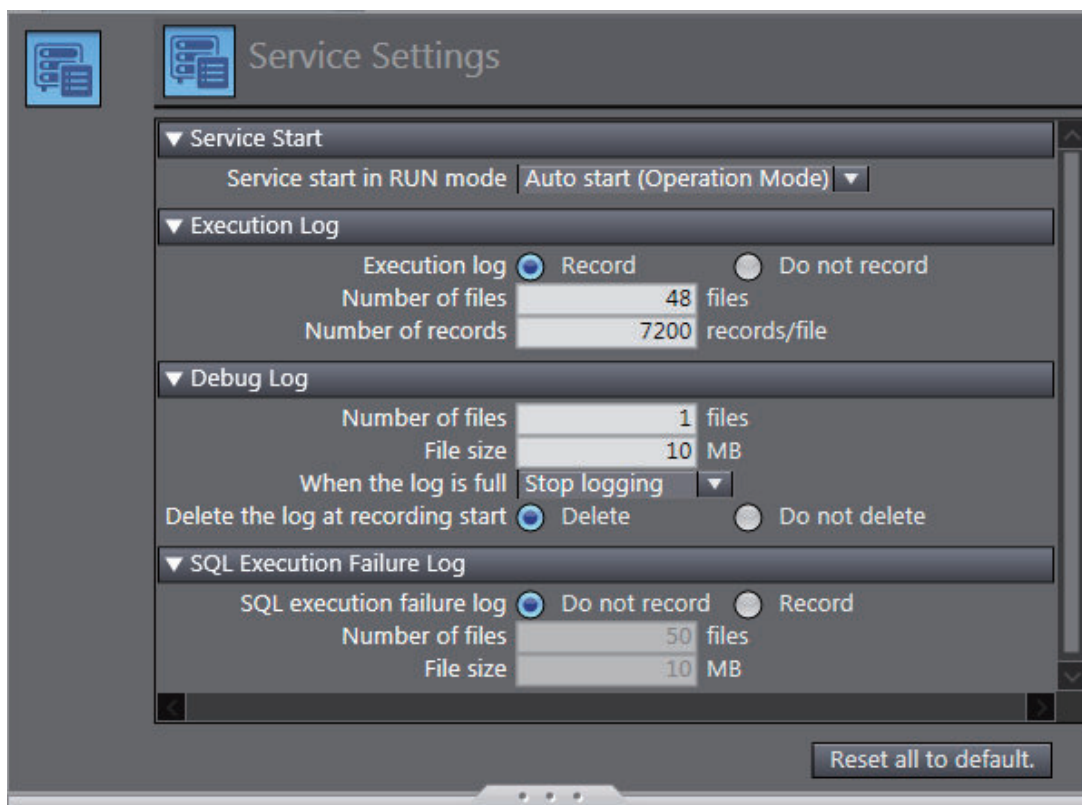


Service Settings

Make a setting for **Service Start**, Execution Log, Debug Log, and SQL Execution Failure Log in the Service Settings.

Refer to *4-1 Run Mode of DB Connection Service and Start/Stop Procedures* on page 4-2 for details on how to start the DB Connection Service.

Refer to *Section 6 How to Use Operation Logs* on page 6-1 for details on the Operation Logs.



Set the following items.

Category	Item	Description	Values
Service Start	Service start in RUN mode	Set whether to automatically start the DB Connection Service when the operating mode of the CPU Unit is set to RUN mode.	- Auto start (Operation Mode)*¹ (Default) - Auto start (Test Mode)*² - Do not start automatically
Execution Log	Execution Log	Set whether to record the Execution Log.	- Record (Default) - Do not recorded
	Number of files	Set the maximum number of files of the Execution Log. When the maximum number of files is reached, the oldest file is deleted and a new file is created.	2 to 100 files (Default: 48 files)
	Number of records	Set the number of log records that can be contained in each Execution Log file. When the maximum number of records is reached, a new file is created.	100 to 65536 records (Default: 7200 records)
Debug Log	Number of files	Set the maximum number of files of the Debug Log.	1 to 100 files (Default: 1 files)
	File size	Set the maximum file size. When the maximum file size is exceeded or when the number of records exceeds 65,536 records in a file, a new file is created.	1 to 100 MB (Default: 10 MB)
	When the log is full	Set the action to be taken when the log has reached the maximum number of files.	- Continue logging (Delete the oldest file) - Stop logging (Default)
	Delete the log at recording start	Set whether to delete the Debug Log contained in the SD Memory Card when recording is started.	- Delete (Default) - Do not delete

Category	Item	Description	Values
SQL Execution Failure Log	SQL execution failure log	Set whether to record the SQL Execution Failure Log.	- Record (Default) - Do not recorded
	Number of files	Set the maximum number of files of the SQL Execution Failure Log. When the maximum number of files is reached, the oldest file is deleted and a new file is created.	2 to 100 files (Default: 50 files)
	File size	Set the maximum file size. When the maximum file size is exceeded or when the number of records exceeds 65,536 records in a file, a new file is created.	1 to 100 MB (Default: 10 MB)

- *1. When a DB Connection Instruction is executed, the DB Connection Service actually accesses the DB.
- *2. When a DB Connection Instruction is executed, the DB Connection Service does not actually access the DB, but the instruction will end normally as if it was executed.



Additional Information

You can calculate the capacity of the Operation Log files that are stored on the SD Memory Card.

If the SD Memory Card often runs out of space, please decrease the values of the following settings.

- Execution Log
Size of each record (Note) x "Number of records" x "Number of log files"
- Debug Log
"File size" x "Number of files"
- SQL Execution Failure Log
"File size" x "Number of files"

Note The maximum value varies by the version of the DB Connection Service.

Version 1.04 or lower: 256 bytes max.

Version 2.00 or higher: 58 KB max.

2-2-2 DB Connection Settings

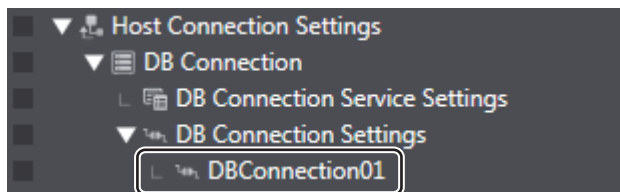
This section describes how to add and rename a DB Connection, and also describes the DB Connection setting procedure and items.

Adding a DB Connection

- 1 Right-click **DB Connection Settings** under **Configurations and Setup - Host Connection Settings - DB Connection** in the Multiview Explorer and select **Add - DB Connection Settings** from the menu. Or, select **DB Connection Settings** from the **Insert Menu**.



A DB Connection is added.*1



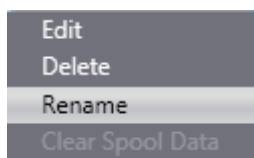
*1. Refer to 1-2-1 *DB Connection Service Specifications* on page 1-5 for the number of DB Connections for each model.

Changing the DB Connection Name

When a DB Connection is created, the following default name is automatically given. "***" is a serial number from 01.

"DBConnection**"

To change the name, right-click the DB Connection in the Multiview Explorer and select **Rename** from the menu.



- You can enter single-byte alphanumeric characters and underscores (_).
- Each DB Connection name can be up to 16 bytes.

● Editing or Deleting the DB Connection Settings

Right-click the DB Connection in the Multiview Explorer and select **Edit** or **Delete** from the menu.

Connection Settings

This section describes how to make a setting of each DB Connection and how to perform a communications test.

● DB Connection Settings

Double-click each DB Connection that you added and make the settings in the **Connection Settings**.

The screenshot shows the 'DBConnection01' window with the following settings:

- Connection name:** DBConnection01
- Database type:** SQL Server
- Encrypted communication:** ☒ Use
- Server certificate:** ServerCertificate.crt, Client_uaexpert.cer, Certificate.pem (Certificate.pem is selected)
- Server specification method:** ☒ IP address
- IP address:** 192.168.250.10
- Host name:** (Empty)
- Instance name/Port No.:** 1521 (Can be omitted)
- Service name/Database name:** XEXDB (Can be omitted)
- User name:** jdbc
- Password:** *****
- Password (for confirmation):** *****
- Login timeout:** 20 s
- Query execution timeout:** 30 s
- Comment:** (Empty)
- DB Communications Test:** Communications Test (Selected)

A 'Reset all to default.' button is located at the bottom right of the window.

Set the following items.

Category	Item	Description	Values
DB Connection	Connection Name	The DB Connection name is displayed.	You can change the DB Connection name. To change the name, right-click the DB Connection in the Multiview Explorer and select Rename from the menu.
	Database type	Set the database type.	- NX701-□□20, NX102-□□20, NJ501-1□20, or NJ101-□□20 - Oracle - SQL Server (Default) - DB2 - MySQL - Firebird - PostgreSQL - NJ501-4320 - Oracle - SQL Server (Default) - MySQL
	Encrypted communication* ¹	Specify whether to enable or disable encrypted communication for each connection.	Do not use or Use Default: Do not use
	Server certificate* ¹	Select a server certificate (including CA/root certificate). (More than one certificate can be selected. Up to five certificates per connection)	File name of the selected server certificate Default: Blank Do not use: Selection disabled Use: Server certificate file name Cannot be omitted for Oracle
	Server specification method	Select the specification method of the server. Select IP address or Host name.	- IP address (Default) - Host name
	IP address	Set the IP address of the server.	Default: Blank This setting cannot be omitted when IP address is selected for Server specification method.
	Host name	Set the host name of the server.* ²	Default: Blank This setting cannot be omitted when Host name is selected for Server specification method.

Category	Item	Description	Values
	Instance name/Port No.	Set the instance name or port number of the server.	• Oracle: Port No. (Can be omitted)*³ e.g. 1521 • SQL Server: Instance name or Port No. (Can be omitted) e.g. INSTANCE1 or 1433 • DB2 Port No. (Can be omitted) e.g. 50000 • MySQL: Port No. (Can be omitted) e.g. 3306 • Firebird: Port No. (Can be omitted) e.g. 3050 • PostgreSQL Port No. (Can be omitted) e.g. 5432 Maximum number of characters for instance name: 64 characters Port No.: 1 to 65535 Default: Blank When omitted, the default port number is used. • Oracle: 1521 • SQL Server: 1433 • MySQL: 3306 • Firebird: 3050 • PostgreSQL: 5432
	Service name/Data-base name	Set the service name or database name in the server.	• Oracle: Service name (Can be omitted)*³ • SQL Server: Database name (Can be omitted) • DB2: Database name (Cannot be omitted) • MySQL: Database name (Cannot be omitted) • Firebird: Database path (Cannot be omitted) e.g., C:/Firebird/OMRON.FDB Or e.g., C:\Firebird\OMRON.FDB • PostgreSQL: Database name (Cannot be omitted) Maximum number of bytes: 127 bytes When omitted, • Oracle: Default service • SQL Server: Default database

Category	Item	Description	Values
	User name	Set the user name for the server.	- DB2: Windows user name of the server - Other DBs: DB user name of the server Maximum number of characters: 127 characters Default: Blank
	Password	Set the password for the server.	- DB2: Windows password of the server - Other DBs: DB password of the server Maximum number of characters: 127 characters Default: Blank
	Login time-out	Set the timeout to be applied when connecting to the DB.	1 to 60 seconds Default: 10 seconds
	Query execution time-out	Set the timeout to be applied at the SQL execution.	1 to 600 seconds Default: 30 seconds
	Comment	Enter a comment.	Maximum number of bytes: 1,024 bytes Default: Blank The comment can be omitted.

*1. This function is displayed for the DB Connection Service version 2.00 or higher.

*2. When you specify a server by its host name, you need to set "DNS to Use" or make the "host settings" in the Built-in EtherNet/IP Port Settings. Refer to the *NJ/NX-series CPU Unit Built-in EtherNet/IP Port User's Manual (Cat. No. W506)* for details on the settings.

*3. The setting cannot be omitted in the DB Connection Service version 2.00 or higher if the encrypted communication is set to *Use*.



Version Information

The supported database types are different for models with the combination of the DB Connection Service version of the CPU Unit and the DB Connection Service version set in the Sysmac Studio project.

For the relationship between the unit version of the CPU Unit and the unit version set in the Sysmac Studio project, refer to *A-4-3 Unit Version, DB Connection Service Version, and Unit Version Set in the Sysmac Studio Project* on page A-27.

● Communications Test

You can test the connection to the DB according to the settings made in the Connection Settings*1 of Sysmac Studio.

*1. This is not the DB Connection Settings that have been transferred to the Controller.

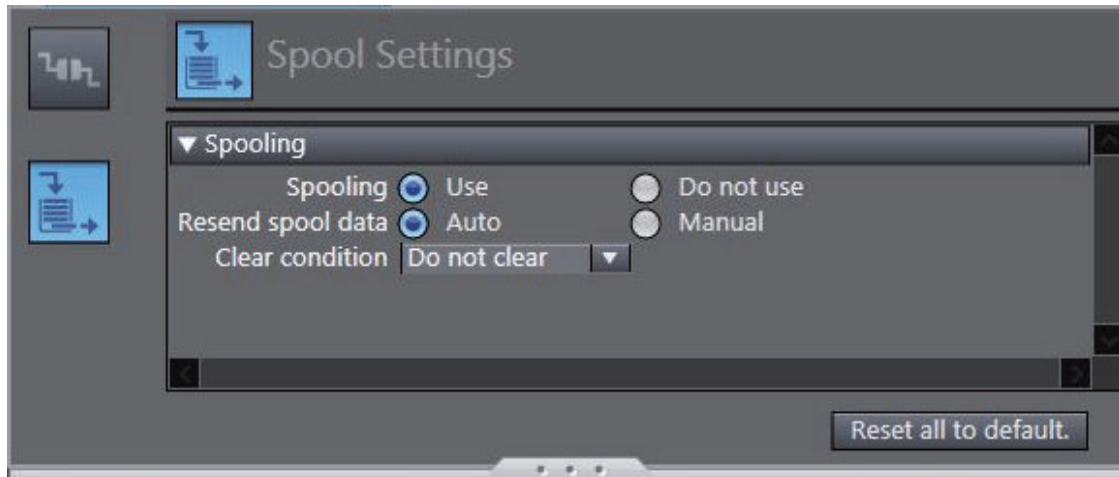
You can perform the communications test while Sysmac Studio is online with the Controller.

- 1** Use the Synchronization function to transfer the DB Connection settings from the computer to the Controller.
- 2** Click the **Communications Test** Button under **DB Communications Test**.
- 3** The result of the communications test is displayed in the text box under the **Communications Test** Button.
When the connection to the server failed from any cause, the SQL status, error code, and detailed error message will be displayed.

SQL status:	Error code defined in the SQL Standards (ISO/IEC 9075).
Error code:	Error code specific to the vendor of DB to connect. When a network failure has occurred, 0 is displayed for error code in some cases. When 0 is displayed, check its SQL status.
Detailed error message:	Error message specific to the vendor of DB to connect.

Spool Settings

Make the settings related to Spool function in the **Spool Settings**.



Refer to 5-2 *Spool Function* on page 5-5 for detailed settings.

3

Programming the DB Connection Function

3

This section describes programming procedure from variable creation to DB access after making the DB Connection settings.

3-1	DB Access Procedure.....	3-2
3-2	Creating a Structure Data Type.....	3-3
3-2-1	Overview.....	3-3
3-2-2	Specifications of Structure Data Type for DB Access.....	3-3
3-2-3	How to Create a Structure Data Type for DB Access.....	3-13
3-3	Creating a DB Map Variable	3-17
3-3-1	DB Map Variables and DB Mapping	3-17
3-3-2	Registration and Attributes of DB Map Variables.....	3-18
3-3-3	Restrictions on DB Map Variables	3-19
3-4	Specifying the Table and Applying the Mapping.....	3-20
3-4-1	DB Mapping by Executing a Create DB Map Instruction	3-20
3-4-2	Clearing the Mapping of DB Map Variables	3-20
3-4-3	Restrictions on DB Mapping	3-20
3-5	Programming and Transfer	3-24
3-5-1	Programming the DB Connection Service	3-24
3-5-2	Displaying DB Connection Instructions on Sysmac Studio.....	3-25
3-5-3	DB Connection Instruction Set.....	3-25
3-5-4	System-defined Variables	3-26
3-5-5	Simulation Debugging of DB Connection Instructions.....	3-27
3-5-6	Transferring the DB Connection Settings and User Program	3-28
3-6	Debugging in Design, Startup, and Operation Phases.....	3-29
3-6-1	Design Phase	3-29
3-6-2	Startup Phase	3-29
3-6-3	Operation Phase.....	3-29

3-1 DB Access Procedure

This section describes a specific programming procedure for using the DB Connection Service. Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for the general programming procedure.

Use the following procedure to access the DB using DB Connection Instructions after making the DB Connection settings.

After the DB mapping^{*1}, use DB connection instructions to execute database operation.

DB mapping ^{*1}	Create a structure data type for DB access.	3-2 <i>Creating a Structure Data Type</i> on page 3-3
		
	Create a variable called "DB Map Variable" using the above structure.	3-3 <i>Creating a DB Map Variable</i> on page 3-17
		
	Establish a DB Connection by executing a DB_Connect (Establish DB Connection) instruction.	4-2 <i>Establishing/Closing a DB Connection</i> on page 4-6
		
	By executing the DB_CreateMapping (Create DB Map) instruction or DB_AttachProcedure instruction (Generate DB Stored Procedure Handle), the specified database table or stored procedure's arguments, return value, and result set is mapped to the DB Map Variables for each SQL type.	3-4 <i>Specifying the Table and Applying the Mapping</i> on page 3-20 * For the stored procedure call function, refer to 5-3-4 <i>Specifying the Table and Applying the Mapping</i> on page 5-20



DB operation	Execute DB Connection Instruction	3-5 <i>Programming and Transfer</i> on page 3-24

*1. "DB mapping" refers to an operation that members of structure-type data used for accessing a database are being associated with columns and arguments, return values, and result sets of stored procedures on the database table. You need to execute the DB mapping for each SQL type and each stored procedure.

3-2 Creating a Structure Data Type

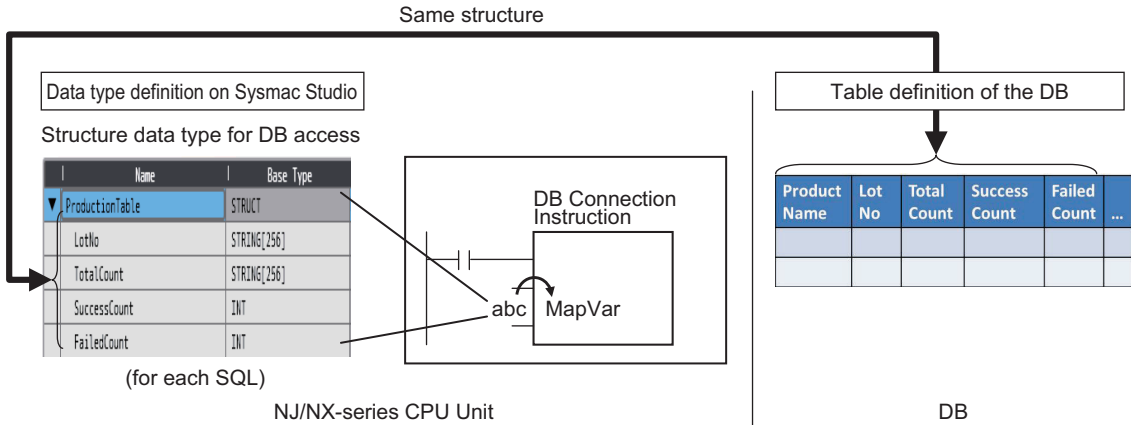
To access a DB, you need to create a user-defined structure data type according to the table definition of the DB.

This section describes the specifications and creation procedure of the structure data type.

3-2-1 Overview

You create a user-defined structure data type on Sysmac Studio based on the data type of the table to access. Register all or some of the columns of the table as structure members.

Each structure member name and data type must match the corresponding column name and data type of the table.



When creating a variable called “DB Map Variable”, you specify the structure as its data type.

3-2-2 Specifications of Structure Data Type for DB Access

Item	Specifications
Structure name	You can specify any name for the structures.
Offset specification for structure members	For all NJ/NX-series Controllers, specify "NJ" for "Offset Type".
Structure members	Register all or some of the columns of the table as members.
Structure member name	Define the same name as the corresponding column of the table. The names are case sensitive.
Structure member's data type	Define a data type that matches the data type of the corresponding column of the table. Refer to the <i>Correspondence of Data Types between NJ/NX-series Controllers and DB</i> on page 3-4 below. However, you cannot specify the following data types and attribute for structure members. - Derivative data types - Array attribute



Precautions for Correct Use

Restrictions on Table's Column Names:

You need to specify the same name for structure members to be used in NJ/NX-series Controllers as the column names of the table to access.

There are following restrictions on structure member names in the NJ/NX-series Controllers.

Therefore, make the column names satisfy the following conditions.

Item	Description
Usable characters	0 to 9, A to Z, a to z Single-byte Japanese kana _ (underscores) Multi-byte characters (e.g., Japanese)
Characters that cannot be used together	• A text string that starts with a number (0 to 9) • A text string that starts with "P_" • A text string that starts with an underscore (_) character • A text string that contains more than one underscore (_) character • A text string that ends in an underscore (_) character • Any text string that consists of an identifier and has a prefix or postfix which contains more than one extended empty space character (i.e., multi-byte spaces or any other empty Unicode space characters)

● Correspondence of Data Types between NJ/NX-series Controllers and DB

The correspondence of data types between NJ/NX-series Controllers and DB is given in the following tables.

- Oracle

Data type category	Data type in DB	Data type in NJ/NX-series Controllers
Characters	VARCHAR2	STRING* ¹
	NVARCHAR2	STRING* ¹
	CHAR	STRING* ¹
	NCHAR	STRING* ¹
	LONG	None
	CLOB	None
	NCLOB	None
Numbers* ²	NUMBER(1)	* ³
	NUMBER(3)	BOOL
	NUMBER(5)	SINT
	NUMBER(10)	INT
	NUMBER(19)	DINT
	NUMBER(3)	LINT
	NUMBER(5)	USINT
	NUMBER(10)	UINT
	NUMBER(20)	UDINT
	NUMBER(19)	ULINT
	NUMBER(19)	TIME* ⁴
	BINARY_FLOAT	REAL
	BINARY_DOUBLE	LREAL
	FLOAT	REAL
	INTEGER	DINT

Data type category	Data type in DB	Data type in NJ/NX-series Controllers
Date	DATE	DATE
	TIMESTAMP	DATE DATE_AND_TIME
	TIMESTAMP WITH TIME ZONE	DATE_AND_TIME
	TIMESTAMP WITH LOCAL TIME ZONE	DATE_AND_TIME
	INTERVAL YEAR TO MONTH	None
	INTERVAL DAY TO SECOND	None
Binary	RAW	None
	LONG RAW	None
	BLOB	None
Others	BFILE	None
	ROWID	None
	UROWID	None
	XMLTYPE	None

- *1. A NULL character is attached to the end of each text string. Therefore, you need to set the value that is one byte bigger than the number of bytes of the DB's data type for the number of bytes to be used in STRING data. You need to set an appropriate value for the number of bytes used in the STRING data according to the data type and character code in the DB. In NJ/NX Series, text strings are handled as UTF-8. One byte is used for each single-byte alphanumeric character and multiple bytes are used for each multi-byte character. Three bytes are used for each Japanese character as a guide.
- *2. The NUMBER(p[,s]) is expressed in the short form where the number of digits after the decimal point (s) is omitted. When the short form is used, the number of digits after the decimal point (s) is 0. If the number of digits after the decimal point (s) is not omitted and 1 or greater numerical value is set, only the integer portion of the value is applicable.
- *3. Digit overflow may occur even in the above data types due to the difference in the valid range.
Example: When the data type in DB is NUMBER(3) and the data type in NJ/NX-series Controllers is USINT:
 - NUMBER(3)'s range: 0 to 999
 - USINT's range: 0 to 255
- *4. Integer in units of nanoseconds.

- SQL Server

Data type category	Data type in DB	Data type in NJ/NX-series Controllers
Numbers ^{*1}	bigint	LINT UDINT
		TIME ^{*2}
	bit	BOOL
	decimal(1) decimal(3) decimal(5) decimal(10) decimal(19) decimal(20) decimal(3) decimal(5) decimal(10) decimal(19)	^{*3} BOOL SINT INT DINT LINT ULINT USINT UINT UDINT
		TIME
	int	DINT UINT
	money	LREAL ^{*4}
	numeric(1) numeric(3) numeric(5) numeric(10) numeric(19) numeric(20) numeric(3) numeric(5) numeric(10) numeric(19)	^{*3} BOOL SINT INT DINT LINT ULINT USINT UINT UDINT
		TIME
	smallint	INT USINT
	smallmoney	REAL ^{*5}
	tinyint	USINT
	float	LREAL
	real	REAL
Date and time	date	DATE
	datetime2	DATE_AND_TIME ^{*6}
	datetime	DATE_AND_TIME
	datetimeoffset	DATE_AND_TIME ^{*6}
	smalldatetime	DATE_AND_TIME
	time	TIME_OF_DAY ^{*6}
String	char	STRING ^{*7}
	text	STRING ^{*7}
	varchar	STRING ^{*7}
	nchar	STRING ^{*7}
	ntext	STRING ^{*7}
	nvarchar	STRING ^{*7}

Data type category	Data type in DB	Data type in NJ/NX-series Controllers
Binary	binary	None
	image	None
	varbinary	None
Others	cursor	None
	hierarchyid	None
	sql_variant	None
	table	None
	uniqueidentifier	None
	xml	None

- *1. The decimal (p[,s]) and numeric (p[,s]) are expressed in the short form where the number of digits after the decimal point (s) is omitted. When the short form is used, the number of digits after the decimal point (s) is 0. If the number of digits after the decimal point (s) is not omitted and 1 or greater numerical value is set, only the integer portion of the value is applicable.
- *2. Integer in units of nanoseconds.
- *3. Digit overflow may occur even in the above data types due to the difference in the valid range.
Example: When the data type in DB is decimal(3) and the data type in NJ/NX-series Controllers is USINT:
 - decimal(3)'s range: 0 to 999
 - USINT's range: 0 to 255
- *4. The significant figures are 15 digits. When the data is written to the DB by a DB Connection Instruction, a value rounded to four decimal places is written.
Example: When 1.79769 is written to the DB, 1.7977 is written.
- *5. The significant figures are 7 digits. When the data is written to the DB by a DB Connection Instruction, a value rounded to four decimal places is written.
Example: When 1.79769 is written to the DB, 1.7977 is written.
- *6. The accuracy is milliseconds.
- *7. A NULL character is attached to the end of each text string. Therefore, you need to set the value that is one byte bigger than the number of bytes of the DB's data type for the number of bytes to be used in STRING data.
You need to set an appropriate value for the number of bytes used in the STRING data according to the data type and character code in the DB. In NJ/NX Series, text strings are handled as UTF-8. One byte is used for each single-byte alphanumeric character and multiple bytes are used for each multi-byte character. Three bytes are used for each Japanese character as a guide.

- DB2

Data type category	Data type in DB	Data type in NJ/NX-series Controllers
Numbers	INT	DINT
	INTEGER	DINT
	BIGINT	LINT TIME
	SMALLINT	INT

Data type category	Data type in DB	Data type in NJ/NX-series Controllers
Fixed-decimal points ^{*1}	DECIMAL(1)	*2 BOOL SINT INT DINT LINT USINT UINT UDINT ULINT
	DECIMAL(3)	
	DECIMAL(5)	
	DECIMAL(10)	
	DECIMAL(20)	
	DECIMAL(3)	
	DECIMAL(5)	
	DECIMAL(10)	
	DECIMAL(20)	
	DECIMAL(20)	TIME
Real numbers	FLOAT	REAL LREAL
	REAL	REAL
	DOUBLE	LREAL
Date	DATE	DATE
	TIME	TIME_OF_DAY
	TIMESTAMP	DATE_AND_TIME
String	CHAR	STRING ^{*3}
	CHARACTER	STRING ^{*3}
	VARCHAR	STRING ^{*3}
	CHAR VARYING	STRING ^{*3}
	CHARACTER VARYING	STRING ^{*3}
	LONG VARCHAR	STRING ^{*3}
	CLOB	None
Binary string	BLOB	None
Others	GRAPHIC	None
	VARGRAPHIC	None
	LONG VARGRAPHIC	None
	DBCLOB	None
	DATALINK	None

*1. The DECIMAL(p[,s]) is expressed in the short form where the number of digits after the decimal point (s) is omitted. When the short form is used, the number of digits after the decimal point (s) is 0. If the number of digits after the decimal point (s) is not omitted and 1 or greater numerical value is set, only the integer portion of the value is applicable.

*2. Digit overflow may occur even in the above data types due to the difference in the valid range.
Example: When the data type in DB is DECIMAL(3) and the data type in NJ/NX-series Controllers is USINT:

- DECIMAL(3)'s range: 0 to 999
- USINT's range: 0 to 255

*3. A NULL character is attached to the end of each text string. Therefore, you need to set the value that is one byte bigger than the number of bytes of the DB's data type for the number of bytes to be used in STRING data.

You need to set an appropriate value for the number of bytes used in the STRING data according to the data type and character code in the DB. In NJ/NX Series, text strings are handled as UTF-8. One byte is used for each single-byte alphanumeric character and multiple bytes are used for each multi-byte character. Three bytes are used for each Japanese character as a guide.

- MySQL:

Data type category	Data type in DB	Data type in NJ/NX-series Controllers
Numbers ^{*1}	BIT	BOOL
	BOOL BOOLEAN	BOOL
	TINYINT	SINT USINT
	SMALLINT	INT UINT
	MEDIUMINT	DINT UDINT
	INT	DINT UDINT
	BIGINT	LINT ULINT TIME
		*2
	DECIMAL(1)	BOOL
	DECIMAL(3)	SINT
	DECIMAL(5)	INT
	DECIMAL(10)	DINT
	DECIMAL(20)	LINT
	DECIMAL(3)	USINT
	DECIMAL(5)	UINT
	DECIMAL(10)	UDINT
	DECIMAL(20)	ULINT
	DECIMAL(20)	TIME
	FLOAT	REAL
	DOUBLE	LREAL
Date and time	DATE	DATE
	DATETIME	DATE_AND_TIME
	TIMESTAMP	DATE_AND_TIME
	TIME	TIME_OF_DAY
String	CHAR	STRING ^{*3}
	VARCHAR	STRING ^{*3}
	TINYTEXT	STRING ^{*3}
	TEXT	STRING ^{*3}
	MEDIUMTEXT	STRING ^{*3}
	LONGTEXT	STRING ^{*3}
Binary	BINARY	None
	VARBINARY	None
	TINYBLOB	None
	BLOB	None
	MEDIUMBLOB	None
	LOB	None

Data type category	Data type in DB	Data type in NJ/NX-series Controllers
Others	ENUM	None
	YEAR	None
	SET	None

- *1. The DECIMAL(p[,s]) is expressed in the short form where the number of digits after the decimal point (s) is omitted. When the short form is used, the number of digits after the decimal point (s) is 0. If the number of digits after the decimal point (s) is not omitted and 1 or greater numerical value is set, only the integer portion of the value is applicable.
- *2. Digit overflow may occur even in the above data types due to the difference in the valid range.
Example: When the data type in DB is DECIMAL(3) and the data type in NJ/NX-series Controllers is USINT:
- DECIMAL(3)'s range: 0 to 999
 - USINT's range: 0 to 255
- *3. A NULL character is attached to the end of each text string. Therefore, you need to set the value that is one byte bigger than the number of bytes of the DB's data type for the number of bytes to be used in STRING data.
You need to set an appropriate value for the number of bytes used in the STRING data according to the data type and character code in the DB. In NJ/NX Series, text strings are handled as UTF-8. One byte is used for each single-byte alphanumeric character and multiple bytes are used for each multi-byte character. Three bytes are used for each Japanese character as a guide.

- Firebird:

Data type category	Data type in DB	Data type in NJ/NX-series Controllers
Numbers	INTEGER	DINT
	BIGINT	LINT TIME
	SMALLINT	INT
Fixed-decimal points *1	DECIMAL(1)	*2 BOOL
	DECIMAL(3)	SINT
	DECIMAL(5)	INT
	DECIMAL(10)	DINT
	DECIMAL(18)	LINT*3
	DECIMAL(3)	USINT
	DECIMAL(5)	UINT
	DECIMAL(10)	UDINT
	DECIMAL(18)	ULINT*3
	NUMERIC(1)	*2 BOOL
	NUMERIC(3)	SINT
	NUMERIC(5)	INT
	NUMERIC(10)	DINT
	NUMERIC(18)	LINT*3
	NUMERIC(3)	USINT
	NUMERIC(5)	UINT
	NUMERIC(10)	UDINT
	NUMERIC(18)	ULINT*3
Real numbers	FLOAT	REAL
	DOUBLE PRECISION	LREAL

Data type category	Data type in DB	Data type in NJ/NX-series Controllers
Date	DATE	DATE
	TIME	TIME_OF_DAY
	TIMESTAMP	DATE_AND_TIME
String	CHAR	STRING*4
	VARCHAR	STRING*4
Others	BLOB	None

- *1. The DECIMAL(p[,s]) and NUMERIC(p[,s]) are expressed in the short form where the number of digits after the decimal point (s) is omitted. When the short form is used, the number of digits after the decimal point (s) is 0. If the number of digits after the decimal point (s) is not omitted and 1 or greater numerical value is set, only the integer portion of the value is applicable.
- *2. Digit overflow may occur even in the above data types due to the difference in the valid range.
Example: When the data type in DB is DECIMAL(3) and the data type in NJ/NX-series Controllers is USINT:
 - DECIMAL(3)'s range: 0 to 999
 - USINT's range: 0 to 255
- *3. The DB can handle up to 18 digits. If an over-18-digit value is written by a DB Connection Instruction, an error will occur.
- *4. A NULL character is attached to the end of each text string. Therefore, you need to set the value that is one byte bigger than the number of bytes of the DB's data type for the number of bytes to be used in STRING data.
You need to set an appropriate value for the number of bytes used in the STRING data according to the data type and character code in the DB. In NJ/NX Series, text strings are handled as UTF-8. One byte is used for each single-byte alphanumeric character and multiple bytes are used for each multi-byte character. Three bytes are used for each Japanese character as a guide.

- PostgreSQL

Data type category	Data type in DB	Data type in NJ/NX-series Controllers
Numbers	boolean	BOOL
	smallint	INT
	integer	DINT
	bigint	LINT TIME
	serial	UDINT
	bigserial	ULINT

Data type category	Data type in DB	Data type in NJ/NX-series Controllers
Fixed-decimal points* ¹	decimal(3)	* ² SINT
	decimal (5)	INT
	decimal (10)	DINT
	decimal (20)	LINT
	decimal (3)	USINT
	decimal (5)	UINT
	decimal (10)	UDINT
	decimal (20)	ULINT
	numeric (3)	* ² SINT
	numeric (5)	INT
	numeric (10)	DINT
	numeric (20)	LINT
	numeric (3)	USINT
	numeric (5)	UINT
	numeric (10)	UDINT
	numeric (20)	ULINT
Real numbers	real	REAL
	double precision	LREAL
Date	timestamp [(p)] [without time zone]	DATE_AND_TIME
	timestamp [(p)] with time zone	DATE_AND_TIME
	date	DATE
	time [(p)] [without time zone]	TIME_OF_DAY
	time [(p)] with time zone	TIME_OF_DAY
String	character(n), char(n)	STRING* ³
	character varying(n), varchar(n)	STRING* ³
	text	STRING* ³

Data type category	Data type in DB	Data type in NJ/NX-series Controllers
Others	bit [(n)]	None
	bit varying [(n)]	None
	Box	None
	Bytea	None
	Cidr	None
	Circle	None
	Inet	None
	interval [fields] [(p)]	None
	Line	None
	Lseg	None
	macaddr	None
	money	None
	path	None
	point	None
	polygon	None
	tsquery	None
	tsvector	None
	txid_snapshot	None
	uuid	None
	xml	None

- *1. The decimal (p[,s]) and numeric (p[,s]) are expressed in the short form where the number of digits after the decimal point (s) is omitted. When the short form is used, the number of digits after the decimal point (s) is 0. If the number of digits after the decimal point (s) is not omitted and 1 or greater numerical value is set, only the integer portion of the value is applicable.
- *2. Digit overflow may occur even in the above data types due to the difference in the valid range.
Example: When the data type in DB is DECIMAL(3) and the data type in NJ/NX-series Controllers is USINT:
- DECIMAL(3)'s range: 0 to 999
 - USINT's range: 0 to 255
- *3. A NULL character is attached to the end of each text string. Therefore, you need to set the value that is one byte bigger than the number of bytes of the DB's data type for the number of bytes to be used in STRING data.
- You need to set an appropriate value for the number of bytes used in the STRING data according to the data type and character code in the DB. In NJ/NX Series, text strings are handled as UTF-8. One byte is used for each single-byte alphanumeric character and multiple bytes are used for each multi-byte character. Three bytes are used for each Japanese character as a guide.



Precautions for Correct Use

- When a data type that is not listed in the above tables is used in the NJ/NX-series Controller, the data may not be converted correctly.
- When reading a value from a database using a DB Connection Instruction, an instruction error (SQL Execution Error) may occur because the data type cannot be converted due to the following reasons.
 - a) The retrieved record contains a column whose value is NULL.
 - b) The combination of data types is not listed in the above tables.

3-2-3 How to Create a Structure Data Type for DB Access

You can use the following procedures for creating a structure data type for accessing a DB.

- Entering the Data on the Data Type Editor
- Pasting the Data from Microsoft Excel onto the Data Type Editor

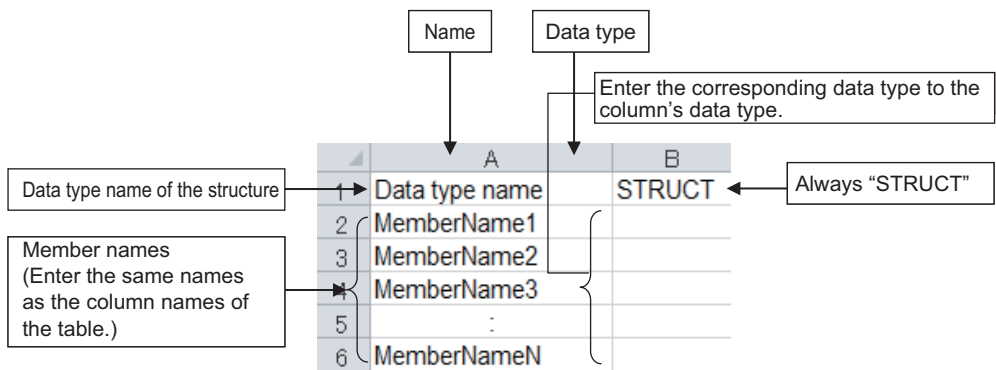
This section gives brief explanation for the operations. Refer to the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for detailed operations.

Entering the Data on the Data Type Editor

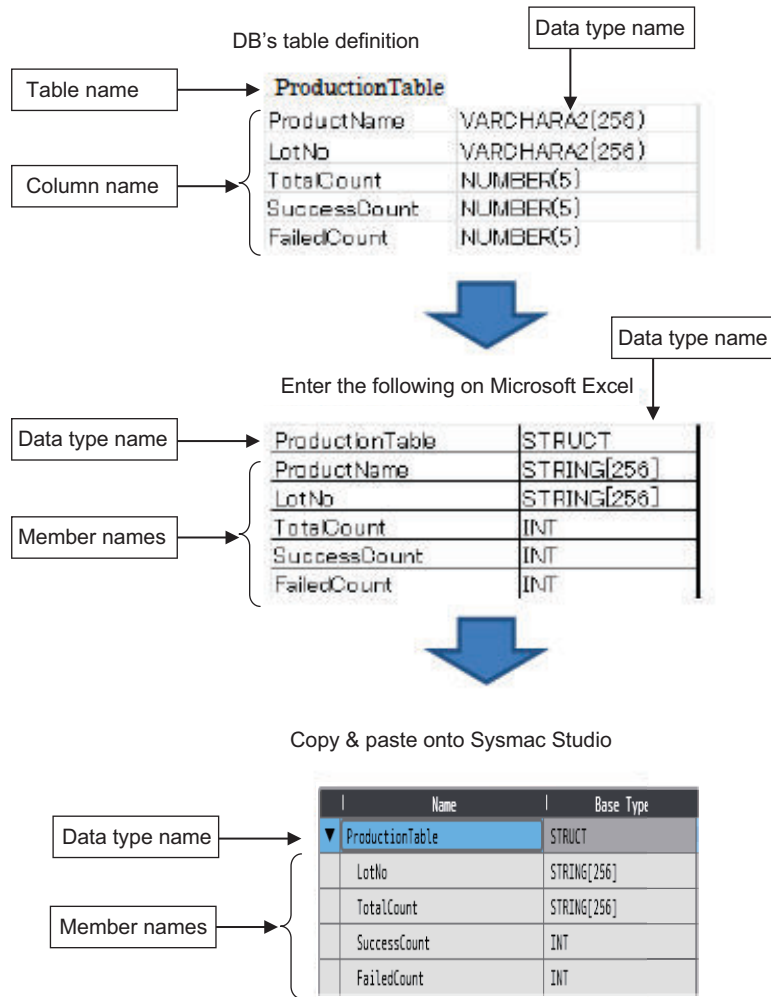
- 1 Double-click **Data Types** under **Programming - Data** in the Multiview Explorer.
- 2 Click the **Structures Side** Tab of the Data Type Editor.
- 3 Enter a data type name on the Structure Data Type Editor.
- 4 Right-click the structure name and select **Create New Member** from the menu. Then, enter a name and data type for each member.

Pasting the Data from Microsoft Excel onto the Data Type Editor

- 1 Use two columns on Microsoft Excel to enter names and data types from the left.
- 2 In the 1st column, enter the data type name of the structure on the 1st line and each member name from the 2nd line.
In the 2nd column, always enter "STRUCT" on the 1st line to create a structure.



- 3 Copy the data area in the Name and Data type columns on Microsoft Excel.
- 4 Paste the data onto the Name and Base Type columns of the Structure Data Type Editor.
Example:



Precautions for Correct Use

You cannot paste the data type onto the Structure "Data Type" Editor in the following cases.

- When a structure member is selected on the editor
- When nothing is selected on the editor

When executing the Paste operation on the Structure Data Type Editor, please select a structure data type, not a member.



Additional Information

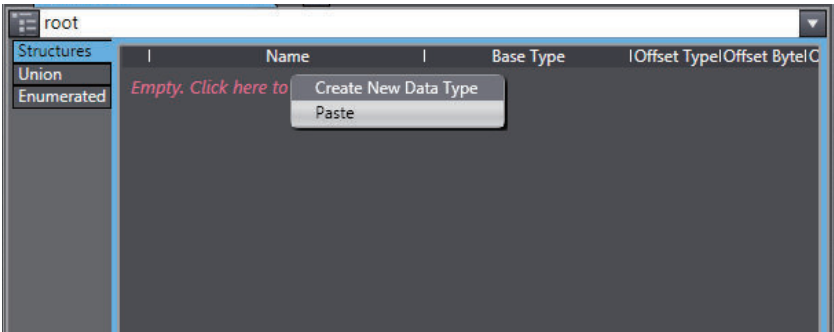
You can reuse table definition data of your DB development tool to create a structure data type for DB access.

Use the following procedure.

- 1) Copy the column name and data type on the table definition data of the DB development tool.
- 2) Create a Column Name column and a Data Type column on Microsoft Excel or other spreadsheet software.
- 3) Change the data type of each column to the corresponding data type for variables of NJ/NX-series CPU Units.
- 4) Insert a line above the data of column names and data types and enter the name of the structure data type.
- 5) Enter "STRUCT" in the Data Type column on the inserted line.
- 6) Copy the data area under the Column Name and Data Type as shown below.

	A	B	C
1			
2	Column Name	Data Type	
3	MyTable	STRUCT	<-Table name
4	MYCOLUMN1	INT	<- Column 1
5	MYCOLUMN2	STRING[20]	<- Column 2
6	MYCOLUMN3	STRING[20]	<- Column 3
7			<- ...

- 7) Right-click on the Structure "Data Type" Editor and select "Paste" from the menu.



A structure data type is created as shown below.

	Name	Base Type	Offset Type	Offset Byte	Offset Char
MyTable	STRUCT	NJ			
MYCOLUMN1	INT				
MYCOLUMN2	STRING[20]				
MYCOLUMN3	STRING[20]				

3-3 Creating a DB Map Variable

After creating a user-defined structure data type for DB access, you create a variable using the data type. The variable is called "DB Map Variable".

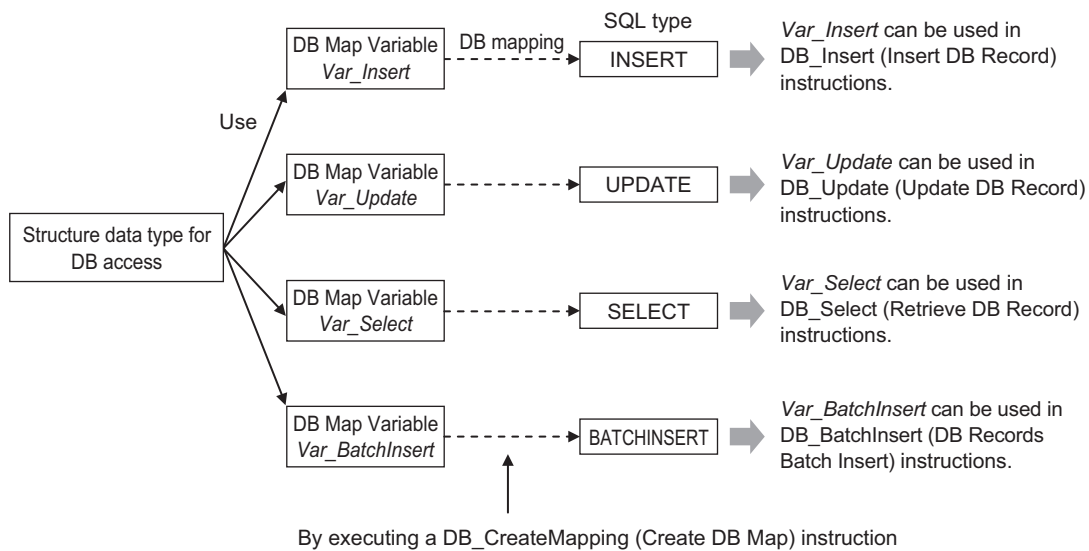
This section describes the specifications and creation procedure of DB Map Variables.

3-3-1 DB Map Variables and DB Mapping

Each DB Map Variable uses a structure data type for DB access as its data type.

By executing a DB_CreateMapping (Create DB Map) instruction for a DB Map Variable in each SQL Type, a DB mapping^{*1} is created.

After creating the DB mapping, you can use the record processing instructions to execute the record processing and stored procedure using DB Map Variables.^{*2}



*1. "DB mapping" refers to the operation of associating each member of DB Map Variables to the columns of a DB table with the arguments, return value, and result set of a stored procedure. You need to execute a DB mapping in each record processing.

*2. The DB mapping for calling a stored procedure is created by executing a DB_AttachProcedure (Generate DB Stored Procedure Handle) instruction. Refer to 5-3 *Stored Procedure Call Function* on page 5-16 for details.

You can map more than one DB Map Variable for a DB Connection.

The following table specifies the record processing operation after creating a structure that contains some of the database columns as its members.

Record processing	Operation
Inserting records (INSERT)	The record values are written to the specified columns of the DB. NULL is entered in the unspecified columns. You need to make a setting for allowing NULL in the DB.
Updating records (UPDATE)	Values are updated only in the specified columns. Values are not changed in the unspecified columns.
Retrieving records (SELECT)	Values are retrieved only from the specified columns. You need to specify only the columns that do not contain NULL.
Inserting all records (BATCHINSERT)	The record values are written to the specified columns of the mapped database. The operation for a single record out of multiple records is same as that for executing a record insertion (INSERT). NULL is entered in the unspecified columns.



Precautions for Correct Use

If you retrieve a record that includes a column of NULL value when executing a DB_Select (Retrieve DB Record) instruction, the instruction will result in an instruction error (SQL Execution Error).



Additional Information

When a DB_CreateMapping (Create DB Map) instruction is executed to create a mapping for a DB Map Variable, it is not checked whether the structure members match the table's columns. The execution result of the record processing instruction results in an error.

3-3-2 Registration and Attributes of DB Map Variables

You can specify the following variable types and attributes for DB Map Variables.

Item		Available type/settings	Restrictions
Registration of variables		Global variable Local variable for a program Local variable for a function block	A local variable for a function cannot be specified.*1
Attributes	Variable name	Any	Refer to the <i>NJ/NX-Series CPU Unit Software User's Manual (Cat. No. W501)</i> for the restrictions on the variable names and other program-related names.
	Data type	Structure data type for DB access	Refer to 3-2 <i>Creating a Structure Data Type</i> on page 3-3.
	AT	Any	
	Retain	Any	
	Initial Value	Any	
	Constant	Any	This attribute cannot be specified for SELECT. A compiling error will occur for DB_Select (Retrieve DB Record) instructions.
	Network Publish	Any	
	Edge	This attribute cannot be specified.	
Array specification		Array can be specified for SELECT and BATCH-INSERT	Array cannot be specified for INSERT nor UPDATE. An instruction error will occur for DB_CreateMapping (Create DB Map) instructions. Refer to 3-3-3 <i>Restrictions on DB Map Variables</i> on page 3-19 for details.

*1. The DB Map Variables cannot be used in any function POU because the DB_CreateMapping (Create DB Map) instruction is a function block type of instruction.

**Precautions for Correct Use**

When a DB Connection Instruction is used in a function block and an in-out variable of the function block is specified as a DB Map Variable, system-defined initial values for the data types are applied to the members of the DB Map Variable when the DB Connection Instruction is executed. Do not specify an in-out variable of a function block as a DB Map Variable.

If you need to use an in-out variable for a DB Connection Instruction, specify an internal variable of the function block as a DB Map Variable and transfer the data between in-out variable and internal variable using a MOVE or other instruction before executing a DB_Insert, DB_Update, or DB_BatchInsert instruction or after executing a DB_Select instruction.

3-3-3 Restrictions on DB Map Variables

This section describes the restrictions on DB Map Variables.

Array Specification for Data Type of DB Map Variables by SQL Type

Whether you can specify a structure array for DB Map Variables depends on SQL type. The following table shows the details.

SQL type	Specifying a structure array for DB Map Variable
INSERT	Not possible
UPDATE	
SELECT	Possible
BATCHINSERT	

Mapping Cannot be Created for a DB Map Variable

Mapping cannot be created for a DB Map Variable in the following cases. The DB_CreateMapping (Create DB Map) instruction ends in an error.

- When the data type of the DB Map Variable is not a structure
- When a derivative data type is contained in structure members of the DB Map Variable
- When a structure array is specified for a DB Map Variable though INSERT or UPDATE is specified for the SQL type in the instruction.
- When a structure variable is specified for the BATCHINSERT DB Map Variable

An Error Occurs when a Record Processing Instruction is Executed

No error is detected when a mapping is created for a DB Map Variable by executing a DB_CreateMapping (Create DB Map) instruction. The execution result of the record processing instruction results in an error.

- When the DB cannot be connected
- When the specified table does not exist in the DB
- When a member name of the DB Map Variable does not match a column in the table
- When a member's data type does not match the data type of the corresponding column

3-4 Specifying the Table and Applying the Mapping

You need to create a mapping from a DB Map Variable to the DB for each SQL type before you can execute a record processing instruction.

This section describes how to create and clear a DB mapping for record processing instructions, as well as the restrictions.

3-4-1 DB Mapping by Executing a Create DB Map Instruction

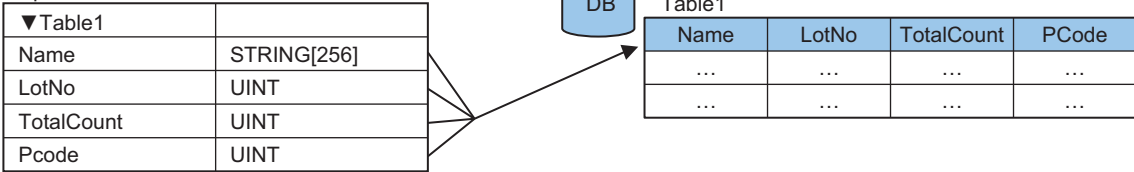
Execute a DB_CreateMapping (Create DB Map) instruction for mapping a DB Map Variable to the connected DB.

Specify the "Table Name", "DB Map Variable", and "SQL Type" in the DB_CreateMapping (Create DB Map) instruction.

By doing so, you can map the DB Map Variable to a database for each SQL type.

Refer to *DB_CreateMapping (Create DB Map)* on page 7-13 for details.

Structure data type definition used by a DB Map Variable



3-4-2 Clearing the Mapping of DB Map Variables

Mapping of DB Map Variables is automatically cleared by the following operations.

- When the DB Connection is closed
- When the DB Connection Service is stopped*1
- When the DB Connection Service is shut down
- When another mapping is applied to a DB Map Variable that has already been mapped by the DB_CreateMapping (Create DB Map) (i.e. mapping to another table or using a different SQL type)

*1. Refer to 4-1-3 *DB Connection Service is Stopped or Cannot be Started* on page 4-4 for details on the stop of the DB Connection Service.



Precautions for Correct Use

Mapping to the DB is automatically cleared when the DB Connection is closed. Therefore, write the user program so that a DB_Connect (Establish DB Connection) instruction is executed before mapping to the DB.

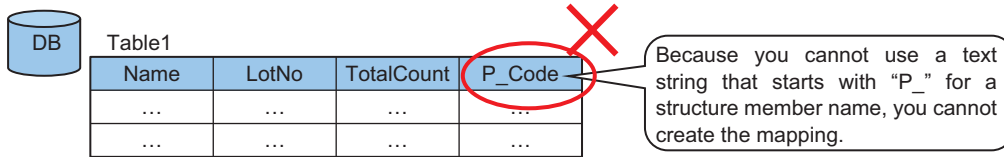
3-4-3 Restrictions on DB Mapping

The DB mapping has the following restrictions.

- Restrictions on Table's Column Names:
When a character that cannot be specified for structure member names is used in a column name of the table, you cannot create the mapping. You need to change the column name of the table.

Example:

When a column name is *P_Code*



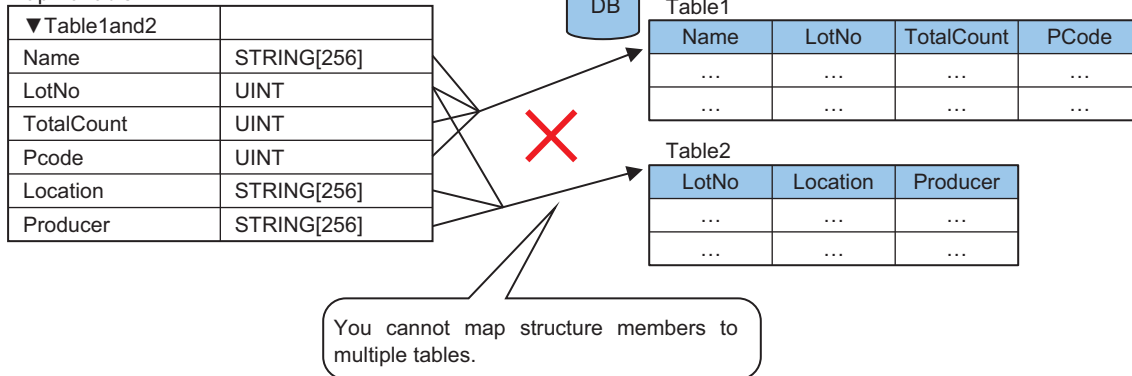
Refer to Precautions for Correct Use: 3-2-2 *Specifications of Structure Data Type for DB Access* on page 3-3 for the characters that cannot be specified for structure member names.

• Restrictions on Mapping to Multiple Tables:

You cannot map the members of a DB Map Variable to columns of different tables.

Example:

Structure data type definition used by a DB Map Variable

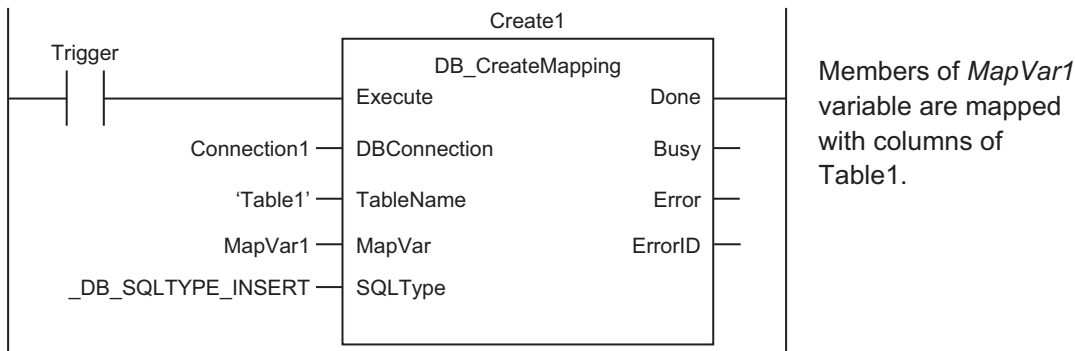


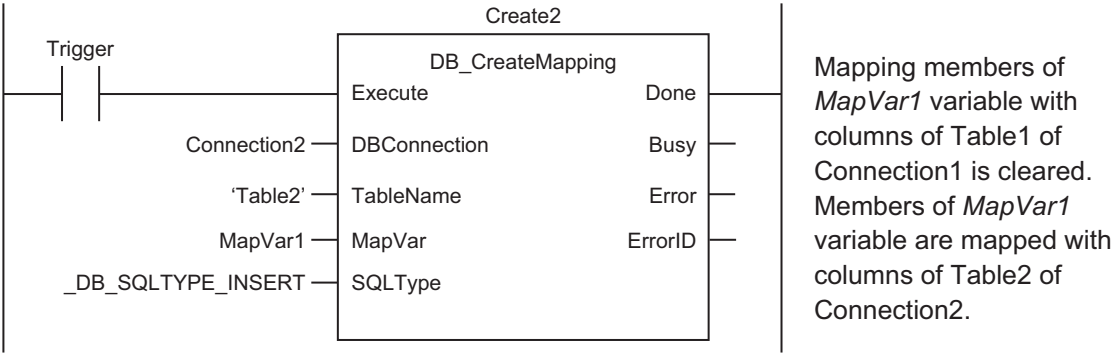
• Restrictions on Mapping to Multiple Tables:

You cannot map a DB Map Variable to two or more tables.

If you execute multiple DB_CreateMapping (Create DB Map) instructions so as to map a single DB Map Variable to two or more tables, the mapping made by the last DB_CreateMapping (Create DB Map) instruction takes effect.

Example:

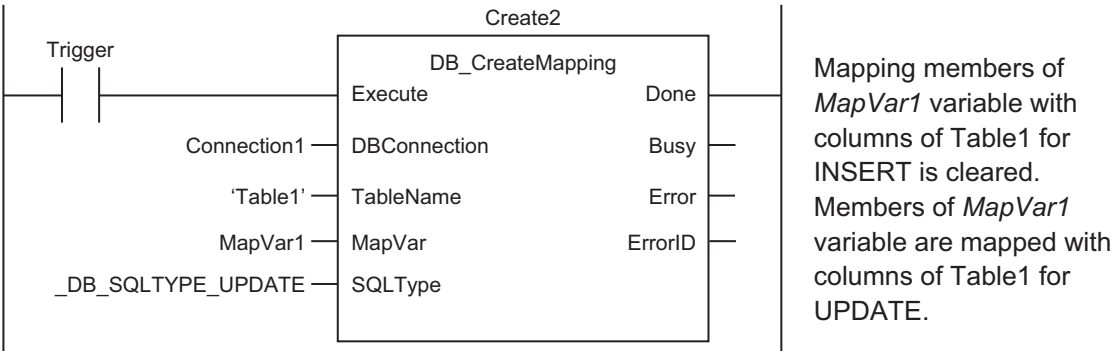
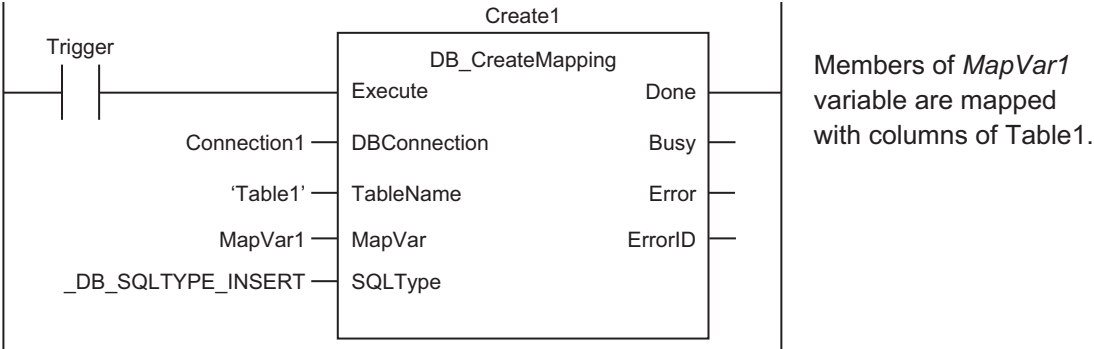




• Restrictions on Mapping to Multiple SQL Types

You cannot map a DB Map Variable for two or more SQL types. If you execute multiple DB_CreateMapping (Create DB Map) instructions so as to map a single DB Map Variable for two or more SQL types, the mapping made by the last DB_CreateMapping (Create DB Map) instruction takes effect.

Example:



• Maximum Number of DB Map Variables For Which a Mapping Can Be Created

The maximum number of DB Map Variables for which you can create a mapping in all connections depends on the database type to connect. Refer to *1-2 DB Connection Service Specifications and System* on page 1-5 for the maximum number of DB Map Variables supported for each DB. When the upper limit is exceeded, an instruction error (Data Capacity Exceeded) will occur when a DB_CreateMapping (Create DB Map) instruction is executed. However, even if the number of DB Map Variables has not reached the upper limit, an instruction error (Data Capacity Exceeded) will occur when the total number of members of the structure definition used as a data type of DB Map Variables in all DB Connections exceeds 10,000 members.

• Definition of DB Map Variables

When a record processing instruction is executed in a POU instance that is different from the POU instance where the DB_CreateMapping (Create DB Map) instruction is executed, the DB Map Variable needs to be defined as a global variable.

3-5 Programming and Transfer

This section describes how to program the DB Connection Service, DB Connection Instruction set, and system-defined variables.

For the actual programming examples, refer to the sample programming for each instruction in *Section 7 DB Connection Instructions* on page 7-1.

3-5-1 Programming the DB Connection Service

Use the following procedure to program the DB Connection Service.

- 1** Select a DB Connection Instruction from the "DB Connect" instruction category of the Toolbox to the right of the program editor of Sysmac Studio. Write the DB Connection Instructions in the following order.*¹
 - 1) Initial Processing
 - Write a DB_ControlService (Control DB Connection Service) instruction when you start the DB Connection Service using the instruction.*²
 - Write a DB_Connect (Establish DB Connection) instruction.
 - Write a DB_CreateMapping (Create DB Map) instruction or DB_AttachProcedure (Generate DB Stored Procedure Handle) instruction.
 - 2) Processing during Operation*³
 - Write record processing and stored procedure instructions, etc.
 - 3) End Processing
 - Write a DB_Close (Close DB Connection) instruction.
 - 4) Power OFF Processing*⁴
 - Write a DB_Shutdown (Shutdown DB Connection Service) instruction.

*1. Refer to 3-5-3 *DB Connection Instruction Set* on page 3-25 for the list of DB connection instructions.

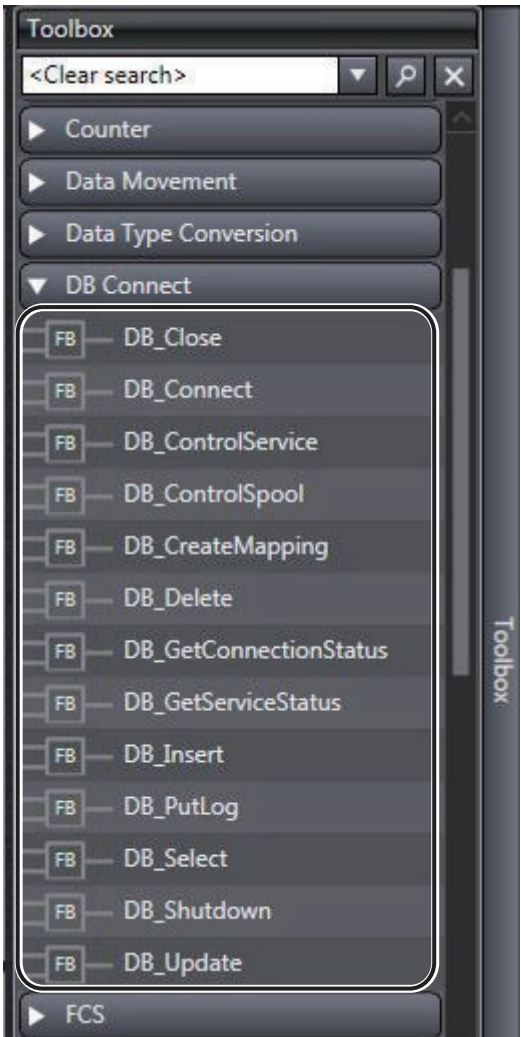
*2. When the DB Connection Service is set to **Auto start**, the DB Connection Service starts automatically by changing the operating mode of the CPU Unit from PROGRAM mode to RUN mode.

*3. When you continuously execute instructions such as record processing and stored procedure instructions, repeat (2) Processing during Operation.

*4. Be sure to execute a DB_Shutdown (Shutdown DB Connection Service) instruction before you turn OFF the power supply to the system. If the power supply is turned OFF without executing a DB_Shutdown (Shutdown DB Connection Service) instruction, the Operation Log file may be corrupted or its contents may be lost.
- 2** Check the status of the DB Connection Service with a system-defined variable.
The status can be Running in Operation Mode, Running in Test Mode, Idle, Error, or Shutdown.
- 3** Transfer the DB Connection settings and user program.
Transfer the DB Connection settings and user program to an NJ/NX-series CPU Unit.
- 4** Cycle the power supply to the Controller.
When you have changed the database type to connect, cycle the power supply to the Controller.

3-5-2 Displaying DB Connection Instructions on Sysmac Studio

The DB Connection Instructions are displayed in the "DB Connect" instruction category of Toolbox of Sysmac Studio.



3-5-3 DB Connection Instruction Set

The following set of DB Connection Instructions is supported.
Refer to *Section 7 DB Connection Instructions* on page 7-1 for details on the DB Connection Instruction.

Category	Instruction	Name	Function
Connection/ disconnection	DB_Connect	Establish DB Con- nection	Connects to a specified DB.
	DB_Close	Close DB Connec- tion	Closes the connection with the DB established by a DB_Connect (Establish DB Connection) instruc- tion.
Mapping	DB_CreateMap- ping	Create DB Map	Creates a mapping from a DB Map Variable to a table of a DB.

Category	Instruction	Name	Function
Record processing	DB_Insert	Insert DB Record	Inserts values of a DB Map Variable to a table of the connected DB as a record.
	DB_Update	Update DB Record	Updates the values of a record of a table with the values of a DB Map Variable.
	DB_Select	Retrieve DB Record	Retrieves records from a table to a DB Map Variable.
	DB_Delete	Delete DB Record	Deletes the records that match the conditions from a specified table.
	DB_BatchInsert	DB Records Batch Insert	Collectively inserts values of array elements for a DB Map Variable into a database table as a single record.
Stored procedure call* ¹	DB_AttachProcedure	Generate DB Stored Procedure Handle	Performs preparation for calling a stored procedure of a database.
	DB_ExecuteProcedure	Execute DB Stored Procedure	Calls a stored procedure using the procedure handle obtained by DB_AttachProcedure.
	DB_DetachProcedure	Release DB Stored Procedure Handle	Releases the stored procedure that was obtained by DB_AttachProcedure.
Others	DB_ControlService	Control DB Connection Service	Starts/stops the DB Connection Service or starts/finishes recording to the Debug Log.
	DB_GetServiceStatus	Get DB Connection Service Status	Gets the current status of the DB Connection Service.
	DB_GetConnectionStatus	Get DB Connection Status	Gets the status of a DB Connection.
	DB_ControlSpool	Resend/Clear Spool Data	Resends or clears the SQL statements spooled by DB_Insert (Insert DB Record) and DB_Update (Update DB Record) instructions.
	DB_PutLog	Record Operation Log	Puts a user-specified record into the Execution Log or Debug Log.
Shut down	DB_Shutdown	Shutdown DB Connection Service	Shuts down the DB Connection Service.* ²

*1. Refer to *5-3 Stored Procedure Call Function* on page 5-16 for details on the stored procedures.

*2. Be sure to execute a DB_Shutdown (Shutdown DB Connection Service) instruction before you turn OFF the power supply to the system. If the power supply is turned OFF without executing a DB_Shutdown (Shutdown DB Connection Service) instruction, the Operation Log file may be corrupted or its contents may be lost.

Refer to *Section 7 DB Connection Instructions* on page 7-1 for details and sample programming of each instruction.

3-5-4 System-defined Variables

You can use the following system-defined variable in the DB Connection Service. A user program performs a processing appropriate for the operation status or the version of DB Connection Service by referencing to a value of system-defined variable in the user program.

Variable name	Data type	Name	Function	Initial Value
Member name				
_DBC_Status	_sDBC_STATUS	DB Connection Service Status	Shows the operation status of the DB Connection Service. For details of the operation status of the DB Connection Service, refer to <i>4-3-1 Operation Status of the DB Connection Service</i> on page 4-7.	
Run	BOOL	Running flag	TRUE when the DB Connection Service is running in Operation Mode or Test Mode.	FALSE
Test	BOOL	Test Mode	TRUE when the DB Connection Service is running in Test Mode.	FALSE
Idle	BOOL	Idle	TRUE when the operation status of the DB Connection Service is Idle.	FALSE
Error	BOOL	Error Stop Flag	TRUE when the operation status of the DB Connection Service is Error.	FALSE
Shutdown	BOOL	Shutdown	TRUE when the operation status of the DB Connection Service is shutdown.	FALSE
_DBC_Version*1	ARRAY[0..1] OF USINT	DB Connection Service Version	The DB Connection Service version is stored. *2 The integer part of the version is stored in the element number 0. The decimal part of the version is stored in the element number 1.	
_JRE_Version*1	ARRAY[0..1] OF USINT	JRE Version	The JRE version is stored. *3 The integer part of the version is stored in the element number 0. The decimal part of the version is stored in the element number 1.	
_DBC_Un-used*4	BOOL	DB Connection Input Variable Omitted	If the stored procedure's argument, return value, or result set does not exist, this variable is specified for omitting the input variable for the DB_AttachProcedure instruction.	FALSE

*1. You can use this system-defined variable with the DB Connection Service version 1.03 or higher.

*2. Example 1) In the case of the DB Connection Service version 1.00, "1" is stored in the element number 0 and "0" is stored in the element number 1.

Example 2) In the case of the DB Connection Service version 1.10, "1" is stored in the element number 0 and "10" is stored in the element number 1.

*3. Example 1) In the case of the JRE version 1.00, "1" is stored in the element number 0 and "0" is stored in the element number 1.

Example 2) In the case of the JRE version 1.10, "1" is stored in the element number 0 and "10" is stored in the element number 1.

*4. You can use this system-defined variable with the DB Connection Service version 2.00 or higher.

3-5-5 Simulation Debugging of DB Connection Instructions

You can perform operation check of the user program using the Simulation function of Sysmac Studio. The DB Connection Instructions perform the following operations during simulation.

- The DB_Connect, DB_Close, DB_Insert, DB_BatchInsert, and other instructions that do not retrieve data will end normally.
- The DB_Select, DB_ExecutePrecedure, and other instructions that retrieve data will end normally as if there was no applicable data.

3-5-6 Transferring the DB Connection Settings and User Program

You transfer the DB Connection settings and user program to an NJ/NX-series CPU Unit using the Synchronization function of Sysmac Studio.

You can specify the following comparison unit for the DB Connection Service in the Synchronization Window.

Synchronization data name	Level	Number	Detailed comparison	Remarks
Host Connection Settings	2	1	Not supported	
DB Connection	3	1	Not supported	
DB Connection Service Settings	4	1	Not supported	
DB Connection Settings	4	1	Not supported	

The DB Connection settings are reflected when the DB Connection Service is started.



Precautions for Correct Use

- If an operation failure or communications error occurs when you execute an operation from Sysmac Studio, retry the operation after performing the following:
 - a) Check the cable connection.
 - b) Check the communications settings.
 - c) Increase the response monitoring time in the Communications Setup.
 - d) Increase the system service execution time ratio.
 - e) Check that the operation status of the DB Connection Service is not "Initializing", "Error", or "Shutdown".
For details of the operation status of the DB Connection Service, refer to *4-3-1 Operation Status of the DB Connection Service* on page 4-7.
- When Sysmac Studio cannot go online, refer to the *NJ/NX-series Troubleshooting Manual (Cat. No. W503)*.

3-6 Debugging in Design, Startup, and Operation Phases

You can use the following debugging procedures according to the phase and actual device environment.

3-6-1 Design Phase

This section gives the debugging procedure in the design phase.

Actual device environment		Debugging method	
CPU Unit	DB	Check item	Operation
Exist	Not exist, or not connected	Checking the executions of DB Connection Instructions on the physical CPU Unit	- Start the DB Connection Service in "Test Mode". - Execute DB Connection Instructions. Note In "Test Mode", SQL statements are not sent actually, but the processing ends as if they were sent normally. - Check the Operation Logs (i.e., Execution Log and Debug Log).

3-6-2 Startup Phase

This section gives the debugging procedure in the startup phase.

Actual device environment		Debugging method	
CPU Unit	DB	Check item	Operation
Exist	Connected	Connection to the DB	- Start the DB Connection Service in "Operation Mode". - Check the status of the DB Connection Service and each DB Connection from Sysmac Studio.
		Checking the DB read/write and timing	- Execute DB Connection Instructions. - Check the Operation Logs (i.e., Execution Log, Debug Log, and SQL Execution Failure Log). (including the check of connection to the DB, executions of SQL statements, and responses)

3-6-3 Operation Phase

This section gives the troubleshooting procedure in the operation phase.

Actual device environment		Debugging method	
CPU Unit	DB	Check item	Operation
Exist	Connected	Regular check	• Check the event logs. • Check the Operation Logs (i.e., Execution Log and SQL Execution Failure Log). • Check the status of the DB Connection Service and each DB Connection from Sysmac Studio. • Check the status of the DB Connection Service and each connection using a DB Connection Instruction.

4

Basic Operations and Status Check

This section describes how to start and stop the DB Connection Service, how to establish and close a DB Connection, and how to check the status of the DB Connection Service and each DB Connection.

4

4-1	Run Mode of DB Connection Service and Start/Stop Procedures	4-2
4-1-1	Run Mode of the DB Connection Service	4-2
4-1-2	How to Start/Stop the DB Connection Service	4-2
4-1-3	DB Connection Service is Stopped or Cannot be Started	4-4
4-1-4	Changing the Run Mode of the DB Connection Service.....	4-5
4-2	Establishing/Closing a DB Connection.....	4-6
4-3	Checking the Status of DB Connection Service and each DB Connection.....	4-7
4-3-1	Operation Status of the DB Connection Service.....	4-7
4-3-2	Checking the Status of the DB Connection Service	4-8
4-3-3	Connection Status of each DB Connection	4-10
4-3-4	Checking the Status of each DB Connection.....	4-11

4-1 Run Mode of DB Connection Service and Start/Stop Procedures

This section describes the Run mode of the DB Connection Service and start/stop procedures.

4-1-1 Run Mode of the DB Connection Service

The DB Connection Service has two Run modes, "Operation Mode" and "Test Mode". You can change the Run mode according to whether to actually access the DB.

This section describes the operations and usage of each Run mode of the DB Connection Service.

Run Mode of the DB Connection Service

You can change the Run mode according to the purpose. In Test Mode, you can test the operations of the DB Connection Service without connecting to the DB.

Run mode	Description	Usage	Environment
Test Mode	- SQL statements are not sent to the DB when DB Connection Instructions are executed. - DB Connection Instructions end normally. However, the instructions for retrieving from the DB do not output anything to the specified DB Map Variable. - Spool function is disabled.	Operation check of user program using DB Connection Instructions when the DB is not connected.	When the DB does not exist, or when the DB exists, but not connected
Operation Mode	- SQL statements are sent to the DB when DB Connection Instructions are executed. - Spool function is enabled.	Practical or trial operation of the system when the DB is connected	When the DB is connected

4-1-2 How to Start/Stop the DB Connection Service

You can use the following three methods to start or stop the DB Connection Service.

- Starting the service automatically when the operating mode of the CPU Unit is changed to RUN mode.
- Starting/stopping the service by online operation from Sysmac Studio.
- Executing a DB_ControlService (Control DB Connection Service) instruction.

Please note that the Run mode of the DB Connection Service cannot be changed while the service is running.

To change the Run mode, you need to stop the DB Connection Service, and then start the service again.

Starting the Service Automatically when Operating Mode of the CPU Unit is Changed to RUN Mode

Double-click **DB Connection Service Settings** under **Configurations and Setup - Host Connection Settings - DB Connection** in the Multiview Explorer. Then, set **Service** start in "Run" mode to "Auto start (Operation Mode)" or "Auto start (Test Mode)" in the Service Settings. (Default: "Auto start (Operation Mode)")

When the operating mode of the CPU Unit is changed from PROGRAM mode to RUN mode, the DB Connection Service is automatically started.



Precautions for Correct Use

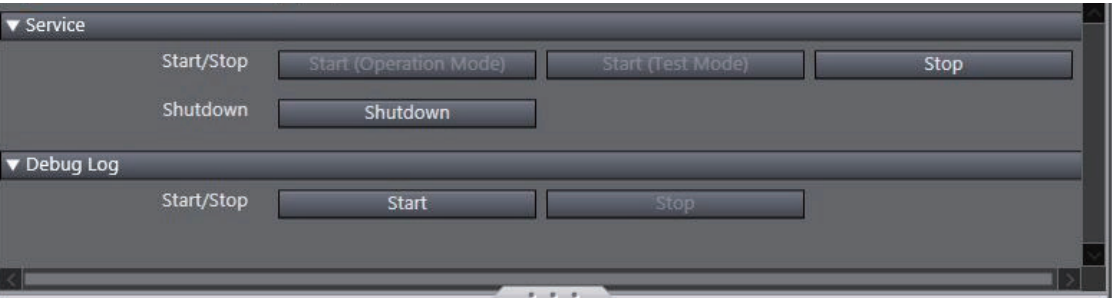
Even if you set "Auto Start" for the DB Connection Service, you cannot execute the DB Connection Instructions until the startup processing of the DB Connection Service is completed. An Instruction Execution Error will occur.
Therefore, write the user program so that the DB Connection Instructions are executed after confirming the status of the DB Connection Service is Running with the "_DBC_Status.Run" system-defined variable ("Running" flag of the DB Connection Service Status).
User program example:

```
IF _DBC_Status.Run = FALSE THEN
    RETURN; (* Abort the processing because the DB Connection Service is
not running *)
END_IF;
(* Execution of DB Connection Instructions *)
(Omitted after this)
```

Starting/Stopping the Service by Online Operation from Sysmac Studio

- 1 Right-click **DB Connection Service Settings** under **Configurations and Setup - Host Connection Settings - DB Connection** in the Multiview Explorer and select **Online Settings** from the menu while online with an NJ/NX-series CPU Unit.

The following **Online Settings** Tab Page is displayed.



You can start or stop the DB Connection Service by clicking a button.

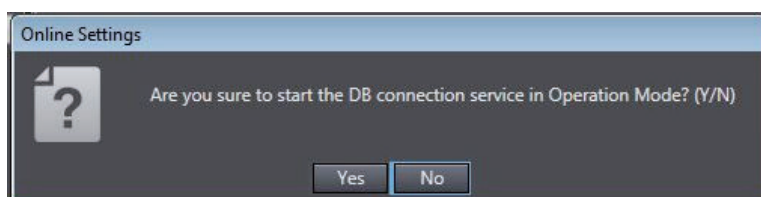
Category	Item	Button	Operation
Service	Start/Stop	Start (Operation Mode)	The DB Connection Service is started in Operation Mode.
		Start (Test Mode)	The DB Connection Service is started in Test Mode.
		Stop	The DB Connection Service is stopped.

- 2** To start the DB Connection Service:
Click the **Start (Operation Mode)** or **Start (Test Mode)** Button.

To stop the DB Connection Service:

Click the **Stop** Button.

A confirmation message is displayed. The following is an example dialog box to be displayed when starting the DB Connection Service in Operation Mode.



- 3** Click the **Yes** Button.

Note You can start or stop the DB Connection Service regardless of the operating mode of the CPU Unit.



Additional Information

You can shut down the DB Connection Service by clicking the **Shutdown Button**. Refer to *5-5 DB Connection Service Shutdown Function* on page 5-26 for details.

Executing a DB_ControlService (Control DB Connection Service) Instruction

Specify one of the following commands in the Cmd input variable of the DB_ControlService (Control DB Connection Service) instruction.

- Start the service in Operation Mode
- Start the service in Test Mode
- Stop the service

Refer to *Section 7 DB Connection Instructions* on page 7-1 for details of the DB_ControlService (Control DB Connection Service) instruction.

4-1-3 DB Connection Service is Stopped or Cannot be Started

In the following conditions, the DB Connection Service cannot be started or the service is stopped.

● **DB Connection Service cannot be Started**

The DB Connection Service cannot be started in the following cases.

- When the DB Connection Service settings are invalid
- When the operation status of the DB Connection Service is "Initializing".
- When the operation status of the DB Connection Service is "Shutdown".

- When the format of the server certificate is corrupt or invalid for either of the connections.

● DB Connection Service is Stopped

The DB Connection Service is stopped in the following cases.

- When the DB Connection Service is stopped by a DB_ControlService (Control DB Connection Service) instruction or Sysmac Studio.
- When the operating mode of the CPU Unit is changed to PROGRAM mode.
- When the Synchronization (download) operation is executed (regardless of whether the DB Connection settings are transferred)
- When the Clear All Memory operation is executed
- When the Restore Controller operation is executed from Sysmac Studio
- When a major fault level Controller error has occurred
- When the DB Connection Service is shut down



Additional Information

- If you stop the DB Connection Service when it is waiting for a response from the DB after sending an SQL statement, the DB Connection Service is stopped after it receives the response from the DB or a communications error is detected.
- If a DB Connection has been established when the DB Connection Service is stopped, the DB Connection is closed.

4-1-4 Changing the Run Mode of the DB Connection Service

You cannot change the Run mode of the DB Connection Service between Operation Mode and Test Mode while the service is running.

To change the Run mode, stop the DB Connection Service and then start the service again.

4-2 Establishing/Closing a DB Connection

After starting the DB Connection Service, you establish or close a DB Connection using an instruction as shown below.

- **Establishing a DB Connection**

Use a DB_Connect (Establish DB Connection) instruction to establish a DB Connection with a specified name.



Precautions for Correct Use

Mapping to the DB is automatically cleared when the DB Connection is closed.
Therefore, write the user program so that a DB_Connect (Establish DB Connection) instruction is executed before a DB_CreateMapping (Create DB Map) instruction is executed.

- **Closing a DB Connection**

Specify the DB Connection name given in the DB_Connect (Establish DB Connection) instruction in a DB_Close (Close DB Connection) instruction and execute the instruction.

Refer to *Section 7 DB Connection Instructions* on page 7-1 for details of each instruction.

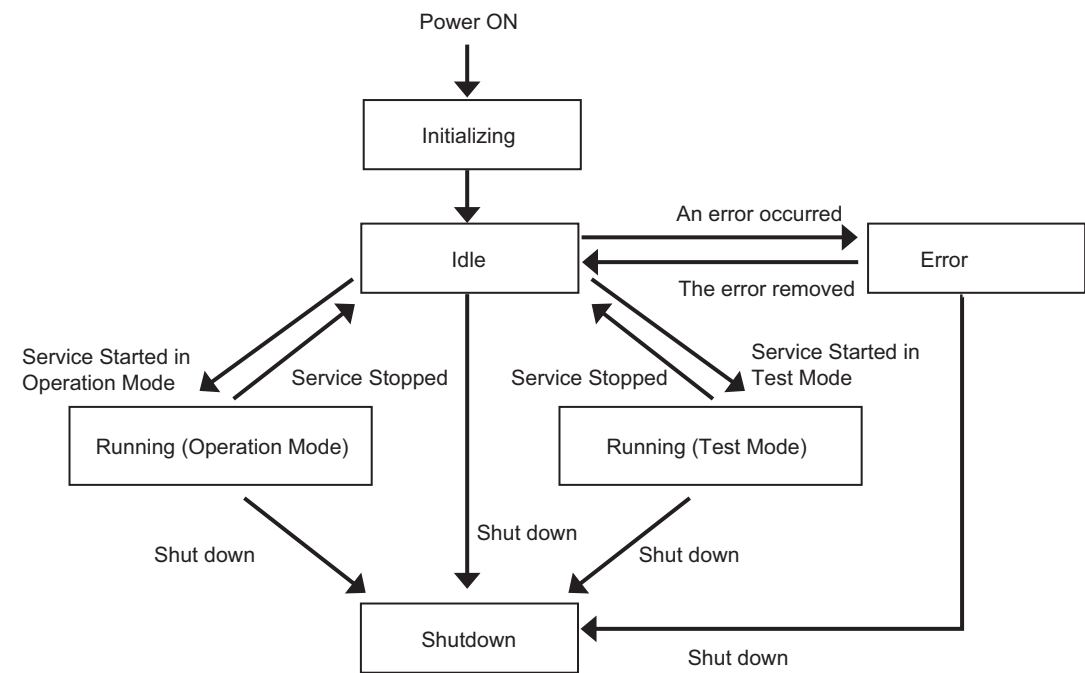
4-3 Checking the Status of DB Connection Service and each DB Connection

This section describes how to check the following status.

- DB Connection Service
- Each DB Connection

4-3-1 Operation Status of the DB Connection Service

This section describes the operation status of the DB Connection Service.



The DB Connection Service has six operation statuses, "Initializing", "Idle", "Running (Operation Mode)", "Running (Test Mode)", "Error", "Shutdown".

After the power supply to the CPU Unit is turned ON, the DB Connection Service enters the "Initializing" status. When the initialization processing is completed, the service enters the "Idle" status. If the DB Connection Service settings are invalid in the "Idle" status, the service enters the "Error" status. When the error is removed, the service returns to the "Idle" status.

When the DB Connection Service is started, the service enters the "Running (Operation Mode)" or "Running (Test Mode)" status according to the Run mode of the DB Connection Service.

When the DB Connection Service is stopped in the "Running (Operation Mode)" or "Running (Test Mode)" status, the service enters the "Idle" status.

When the DB Connection Service shutdown function is executed, the service enters the "Shutdown" status.

The following table gives the details of each status.

Status	Description	Remarks
Initializing	The DB Connection Service was started but has not entered the Idle status after the power supply to the CPU Unit was turned ON.	The DB Connection Service cannot be started.
Idle	The DB Connection Service is not running without having any error.	The DB Connection settings can be changed. The DB Connection Instructions cannot be executed.
Running (Operating Mode)	The DB Connection Service is running in Operation Mode.	The DB Connection settings cannot be changed. The DB Connection Instructions can be executed.
Running (Test Mode)	The DB Connection Service is running in Test Mode.	The DB Connection settings cannot be changed. The DB Connection Instructions can be executed (, but SQL statements are not sent to the DB).
Error	The DB Connection Service cannot run due to an error.	The status changes to Error in the following case. • When the DB Connection Service settings are invalid.
Shutdown	The DB Connection Service is already shut down.	The status changes to Shutdown when the DB Connection Service is shut down by an instruction or Sysmac Studio operation. After the shutdown processing of the DB Connection Service is completed, you can safely turn OFF the power supply to the CPU Unit. You cannot start the DB Connection Service again until you execute the Reset Controller operation or cycle the power supply to the CPU Unit.

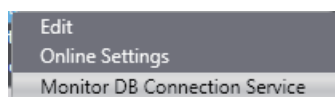
4-3-2 Checking the Status of the DB Connection Service

You can use the following methods to check the status of the DB Connection Service.

- DB Connection Service Monitor of Sysmac Studio
- DB_GetServiceStatus (Get DB Connection Service Status) instruction
- System-defined variable

Checking the Status with DB Connection Service Monitor of Sysmac Studio

Right-click **DB Connection Service Settings** under **Configurations and Setup - Host Connection Settings - DB Connection** in the Multiview Explorer and select **Monitor DB Connection Service** from the menu while online with an NJ/NX-series CPU Unit.



The following **DB Connection Service Monitor** Tab Page is displayed.

Service Settings	
▼ Operation Information	
Operation status	Running (Operation Mode)
Operating time	0:00:01:42
▼ Operation Log	
Debug log	OFF
▼ Query Execution	
Number of normal executions	0
Number of error executions	0
▼ Spooling	
Number of spool data	0

You can check the following in the monitor unless the operation status of the DB Connection Service is "Initializing" or "Shutdown".

Category	Item	Description	Values
Operation Information	Operation status	Operation status of the DB Connection Service.	- Running (Operating Mode) - Running (Test Mode) - Idle - Error Refer to 4-3-1 <i>Operation Status of the DB Connection Service</i> on page 4-7 for details.
	Operating time	Time elapsed since the DB Connection Service was started.	Duration (Unit: d:h:m:s)
Operation Log	Debug log	ON while the Debug Log is recorded.*1	ON/OFF
Query Execution	Number of normal executions	Total number of times in all connections when an SQL statement is normally executed. Including the number of times when a spooled SQL statement is resent. This value is cleared when the DB Connection Service is started.	Number of normal executions
	Number of error executions	Total number of times in all connections when an SQL statement execution failed. This is the number of times when an SQL statement is not spooled, but discarded. The number of times when a statement is spooled is not included. This value is cleared when the DB Connection Service is started.	Number of error executions
Spooling	Number of spool data	Number of spooled SQL statements in all connections.	Number of Spool data

*1. The Debug log flag remains ON even if recording to the log is stopped in the following cases.

- When the When the log is full parameter is set to "Stop logging" in the Service Settings, and the maximum number of files is reached
- When the SD Memory Card capacity is insufficient
- When writing to the SD Memory Card failed

Checking the Status using a Get DB Connection Service Status Instruction

You can check the following operation information of the DB Connection Service using a DB_GetServiceStatus (Get DB Connection Service Status) instruction.

Information	Description
Debug Log flag	TRUE while the Debug Log is recorded.*1
Operating time	Time elapsed since the DB Connection Service was started. When the DB Connection Service is stopped, the time from start to stop is retained. This value is cleared the next time the DB Connection Service is started.
Number of normal executions	Total number of times in all connections when an SQL statement is normally executed. Including the number of times when a spooled SQL statement is resent. This value is cleared when the DB Connection Service is started.
Number of error executions	Total number of times in all connections when an SQL statement execution failed. This value is cleared when the DB Connection Service is started.
Number of Spool data	Number of spooled SQL statements in all connections.

*1. The Debug log flag remains TRUE even if recording to the log is stopped in the following cases.

- When the When the log is full parameter is set to "Stop logging" in the Service Settings, and the maximum number of files is reached
- When the SD Memory Card capacity is insufficient
- When writing to the SD Memory Card failed

Checking the Status with a System-defined Variable

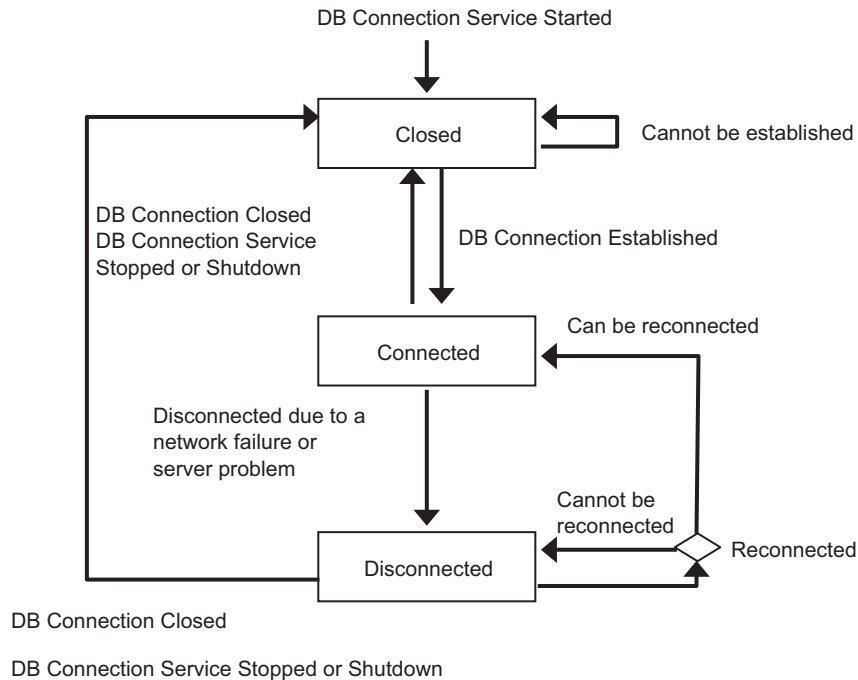
You can check the operation status of the DB Connection Service with the `_DBC_Status` system-defined variable.

Use this variable when checking the status of the DB Connection Service from the user program or checking the shutdown of the DB Connection Service from an HMI.

<code>_DBC_Status</code> system-defined variable		Status					
Member	Meaning	Initializing	Running (Operation Mode)	Running (Test Mode)	Idle	Error	Shutdown
Run	Running flag	FALSE	TRUE	TRUE	FALSE	FALSE	FALSE
Test	Test mode	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE
Idle	Idle	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
Error	Error stop flag	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE
Shutdown	Shutdown	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE

4-3-3 Connection Status of each DB Connection

This section describes the connection status of each DB Connection.



Each DB Connection has three statuses, "Closed", "Connected", and "Disconnected".

After the DB Connection Service is started, each DB Connection enters the "Closed" status. When the DB Connection is established in the "Closed" status, the DB Connection enters the "Connected" status. If the DB Connection cannot be established, it remains in the "Closed" status.

When a network failure or server problem occurs in the "Connected" status, the DB Connection enters the "Disconnected" status.

The DB Connection tries reconnection periodically in the "Disconnected" status. The DB Connection enters the "Connected" status if the DB can be reconnected and remains in the "Disconnected" status if the DB cannot be reconnected.

When the DB Connection is disconnected or the DB Connection Service is stopped or shutdown in the "Connected" or "Disconnected" statuses, the DB Connection enters the "Closed" status.

The following table gives the details of each status.

Status	Description	Remarks
Closed	The DB is not connected.	
Connected	The DB is connected.	You can execute SQL statements such as INSERT and SELECT using instructions.
Disconnected	The DB was disconnected due to a network failure, server's problem, or other causes.	If the DB Connection enters this status during instruction execution, the SQL statement is spooled. Reconnection is attempted periodically.

4-3-4 Checking the Status of each DB Connection

You can use the following methods to check the status of each DB Connection.

- Connection Monitor Table of Sysmac Studio
- DB_GetConnectionStatus (Get DB Connection Status) instruction

Checking the Status with Connection Monitor Table of Sysmac Studio

Right-click **DB Connection Settings** under **Configurations and Setup - Host Connection Settings** - **DB Connection** in the Multiview Explorer and select **Connection Monitor Table** from the menu while online with an NJ/NX-series CPU Unit.

The following **Connection Monitor Table** Tab Page is displayed.

Connection Name	DBConnection01	
▼ Connection Status		
Connection	Closed	
Connected time	0:00:54:43.481	
Disconnected time	0:00:00:00.000	
Disconnection date/time	1/1/1970 0:00:00.000	
▼ Query Execution		
Number of normal executions	0	
Number of error executions	0	
Response time	0:00:00:00.000	
▼ Spooling		
Number of spool data	0	
Spool usage	0%	
▼ Connection Error		
SQL status		
Error code		
Error message		

You can monitor the following of each DB Connection unless the operation status of the DB Connection Service is "Idle" or "Shutdown".

Category	Item	Description	Values
Connection Status	Connection	Status of the DB Connection.	• Closed • Connected • Disconnected Refer to 4-3-3 <i>Connection Status of each DB Connection</i> on page 4-10.
	Connected time	Total time when the DB is connected. This value is cleared when Connection changes from Closed to Connected.	Duration (Unit: d:h:m:s.ms)
	Disconnected time	Disconnected time Total time when the DB is disconnected. This value is cleared when the status changes from Closed to Connected.	Duration (Unit: d:h:m:s.ms)
	Disconnection date/time	Date and time when the DB is disconnected due to a network failure, server's problem, or other causes.*1 This value is cleared when the DB Connection Service is started.	Date and time

Category	Item	Description	Values
Query Execution	Number of normal executions	Number of times when an SQL statement is normally executed. Including the number of times when a spooled SQL statement is resent. This value is cleared when the DB Connection Service is started.	Number of normal executions
	Number of error executions	Number of times when an SQL statement execution failed. This is the number of times when an SQL statement is not spooled, but discarded. The number of times when a statement is spooled is not included. This value is cleared when the DB Connection Service is started.	Number of error executions
	Response time	Time elapsed since the CPU Unit sent the SQL statement until the CPU Unit received its SQL execution result in the latest execution of SQL statement.*2 The response time is stored only when normal response is returned from the DB. If a DB Connection Instruction Execution Timeout has occurred, the response time is not stored when the execution of the instruction is completed (i.e. when the Error output variable changes from FALSE to TRUE). The response time is stored when a normal response is returned from the DB after the DB Connection Instruction Execution Timeout occurred. This value is cleared when the DB Connection Service is started.	Duration (Unit: d:h:m:s.ms)
Spooling	Number of spool data	Number of SQL statements stored in the Spool memory.	Number of Spool data
	Spool usage	Use rate of the Spool memory for each DB Connection.	Spool usage in percentage (%)
Connection Error	SQL status	Error code defined in SQL Standards (ISO/IEC 9075) to be shown when a network failure or an SQL Execution Error occurred.*3 The value of the latest error in the connection is stored. This value is cleared when the DB Connection Service is started.	---
	Error code	Error code that is specific to DB vendor to be shown when a network failure or an SQL Execution Error occurred.*3 When a network error has occurred, 0 is displayed for error code in some cases. When 0 is displayed, check its SQL status. The code of the latest error in the connection is stored. This value is cleared when the DB Connection Service is started.	---
	Error message	Error message that is specific to DB vendor to be shown when a network failure or an SQL Execution Error occurred.*3 The message of the latest error in the connection is stored. This value is cleared when the DB Connection Service is started.	---

*1. The date and time information follows the time zone set when the power supply to the Controller is turned ON. After you change the time zone, cycle the power supply.

- *2. It refers to a record processing instruction, a stored procedure call instruction, or resending spool data (automatic or manual transmission by using the DB_ControlSpool instruction). Refer to 3-5-3 *DB Connection Instruction Set* on page 3-25 for details on the instructions.
- *3. The value may differ by unit version of the CPU Unit.
The value of connection error to SQL Server was changed in the unit version 1.08 of the CPU Units.

Checking the Status using a Get DB Connection Status Instruction

You can check the connection status and information of each DB Connection using a DB_GetConnectionStatus (Get DB Connection Status) instruction.

Information		Description
Connection status of the DB Connection		Connection status (Closed, Connected, or Disconnected) of the DB Connection.
Connection information of the DB Connection	Connected time	Total time when the DB is connected. This value is cleared when the status changes from Closed to Connected.
	Disconnected time	Disconnected time Total time when the DB is disconnected. This value is cleared when the status changes from Closed to Connected.
	Number of normal executions	Number of times when an SQL statement is normally executed. Including the number of times when a spooled SQL statement is re-sent. This value is cleared when the DB Connection Service is started.
	Number of error executions	Number of times when an SQL statement execution failed. This is the number of times when an SQL statement is not spooled, but discarded. The number of times when a statement is spooled is not included. This value is cleared when the DB Connection Service is started.
	Number of Spool data	Number of SQL statements stored in the Spool memory. This value returns to 0 when the Spool data is cleared.
	Spool usage	Use rate of the Spool memory for the DB Connection in percentage (%). This value returns to 0 when the Spool data is cleared.
	Disconnection date/time	Date and time when the DB is disconnected due to a network failure, server's problem, or other causes.*1 This value is cleared when the DB Connection Service is started.
	SQL status	Error code defined in SQL Standards (ISO/IEC 9075) to be shown when a network failure or an SQL Execution Error occurred.*2 This value is cleared when the DB Connection Service is started.
	Error code	Error code that is specific to DB vendor to be shown when a network failure or an SQL Execution Error occurred.*2 When a network error has occurred, 0 is displayed for error code in some cases. When 0 is displayed, check its SQL status. This value is cleared when the DB Connection Service is started.
	Error message	Error message that is specific to DB vendor to be shown when a network failure or an SQL Execution Error occurred.*2 This value is cleared when the DB Connection Service is started.

*1. The date and time information follows the time zone set when the power supply to the Controller is turned ON. After you change the time zone, cycle the power supply.

*2. The value may differ by unit version of the CPU Unit.
The value of connection error to SQL Server was changed in the unit version 1.08 of the CPU Units.

Other Functions

This section describes other functions of the DB Connection Service.

5-1	Examples of Using Functions	5-3
5-2	Spool Function	5-5
5-2-1	Overview	5-5
5-2-2	Spooling System	5-5
5-2-3	Applicable Instructions and Spooling Execution Conditions	5-5
5-2-4	Memory Area Used by the Spool Function	5-6
5-2-5	Spool Function Settings	5-8
5-2-6	How to Resend the SQL Statements Stored in the Spool Memory	5-8
5-2-7	Clearing the SQL Statements from the Spool Memory	5-9
5-2-8	Relationship with the DB Connection Instructions	5-11
5-2-9	How to Estimate the Number of SQL Statements that can be Spooled	5-13
5-3	Stored Procedure Call Function	5-16
5-3-1	Overview	5-16
5-3-2	Specifications of the Stored Procedure Call Function for Databases	5-17
5-3-3	How to Execute the Stored Procedure Call Function	5-19
5-3-4	Specifying the Table and Applying the Mapping	5-20
5-3-5	Errors during Stored Procedure Call	5-22
5-4	Batch Insert Function	5-24
5-4-1	Overview	5-24
5-4-2	How to Execute the Batch Insert Function	5-25
5-5	DB Connection Service Shutdown Function	5-26
5-5-1	Overview	5-26
5-5-2	Shutdown System	5-26
5-5-3	How to Execute the Shutdown Function	5-27
5-5-4	How to Check the Shutdown of the DB Connection Service	5-27
5-6	How to Prevent Losing SQL Statements at Power Interruption	5-28
5-6-1	Overview	5-28
5-6-2	Procedures	5-28
5-7	Timeout Monitoring Functions	5-33
5-7-1	Timeout Monitoring Functions	5-33
5-7-2	Login Timeout	5-33
5-7-3	Query Execution Timeout	5-34
5-7-4	Communications Timeout	5-34
5-7-5	Instruction Execution Timeout	5-35
5-7-6	Keep Alive Monitoring Time	5-35
5-8	Other Functions	5-36

5-8-1 Backup/Restore Function in the DB Connection Service 5-36

5-8-2 Operation Authority Verification in the DB Connection Service 5-37

5-8-3 Encrypted Communication..... 5-37

5-1 Examples of Using Functions

This section explains examples of using functions described in this chapter.

DB Connection Service has various functions designed to handle various events that will occur during data exchange with a relational database in the server.

The typical events that can occur, the outline of their countermeasures, and the relationship between the events and DB Connection Service functions are shown below. For details on how to deal with it, refer to the following items described in this section.

No.	Typical events	Effects of the event when it is occurred	Outline of the countermeasures	DB Connection Service functions	Reference in this section
1	For tasks requiring complex processing, such as the summary of data from multiple tables	If all the processing is performed on the controller, the overall processing may become inefficient.	Implement measures to prevent this from happening by using the stored procedure call function for calling a stored procedure on the relational database and run multiple processing in a single instruction.	Stored procedure call function	5-3 <i>Stored Procedure Call Function</i> on page 5-16
2	When it is necessary to insert multiple records	The user program could become complex and inefficient, and it is possible that the data saving does not complete within the time required by the application.	Use the batch insert function to prevent them from happening.	Batch insert function	5-4 <i>Batch Insert Function</i> on page 5-24
3	When the interruption of the power supply to the Controller occurred	- Possibility of loss of SQL statements to be sent - As a result, the possibility of missing data stored in the relational database	When resending the SQL statement the next time the power is turned ON, take measures on the user program in combination with the spool function.	Spool function	5-2 <i>Spool Function</i> on page 5-5 5-6 <i>How to Prevent Losing SQL Statements at Power Interruption</i> on page 5-28
			Use an uninterruptible power supply for the power supply of the Controller. If the power supply using the uninterruptible power supply can not be maintained, shut down the DB Connection Service.	DB Connection Service shutdown function	5-5 <i>DB Connection Service Shutdown Function</i> on page 5-26
		Possibility of loss of Operation Log data	Use an uninterruptible power supply for the power supply of the Controller. If the power supply using the uninterruptible power supply can not be maintained, shut down the DB Connection Service.	DB Connection Service shutdown function	5-5 <i>DB Connection Service Shutdown Function</i> on page 5-26

No.	Typical events	Effects of the event when it is occurred	Outline of the countermeasures	DB Connection Service functions	Reference in this section
4	When a load in the server temporarily increased	• Possibility of DB Connection Service delay • As a result, the possibility of missing data stored in the relational database	Implement the countermeasures using the timeout monitoring functions.	• Timeout monitoring functions • Spool function	5-7 <i>Timeout Monitoring Functions</i> on page 5-33 5-2 <i>Spool Function</i> on page 5-5
5	When a response speed in the server relatively continued to decrease for a long time	Possibility of missing data stored in relational database due to insufficient spool capacity	To prevent missing data, implement one of the followings until the spooled SQL statement is resent and the capacity shortage is resolved. • Pause or slow down operation of the equipment • Data evacuation to user-defined variables in the user program	• Spool function	5-2 <i>Spool Function</i> on page 5-5
6	When a server failure has occurred for a long time Example: • Ethernet network disconnection or noise • Power loss of the network equipment or the server • The DB is stopped in the server. • Hardware failure of the server	Possibility of missing data stored in the relational database	• Implement the countermeasures shown in above No. 2 and 3. • Use an uninterruptible power supply for the server. If the power supply using the uninterruptible power supply can not be maintained, shut down the DB Connection Service.	• DB Connection Service shutdown function	5-5 <i>DB Connection Service Shutdown Function</i> on page 5-26

5-2 Spool Function

This section describes spooling of unsent SQL statements in the DB Connection Service.

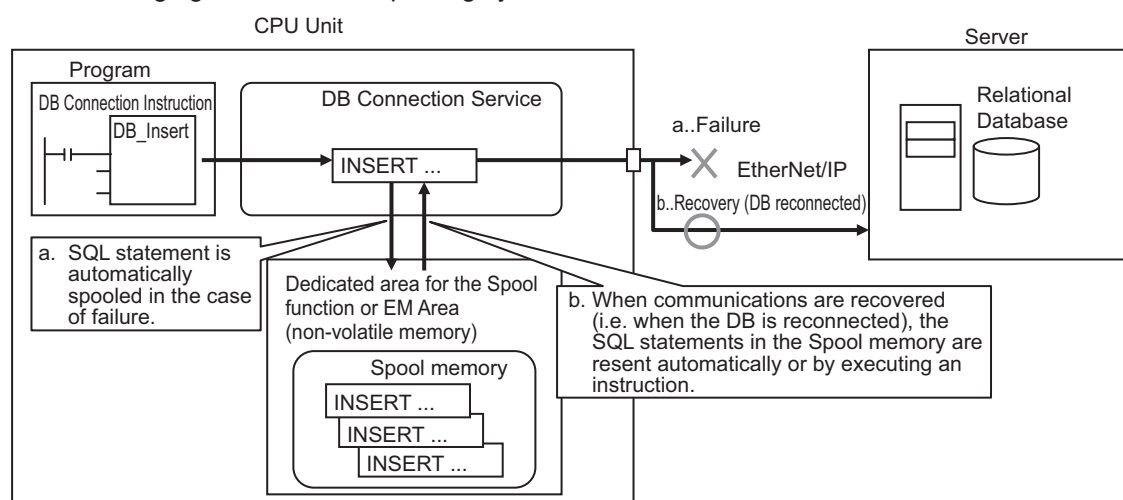
5-2-1 Overview

When a failure occurred in information exchange between DB Connection Service and DB, the unsent SQL statements are stored in a memory used for CJ-series Units and resent when the problem is solved.

You can set whether to enable or disable the Spool function for each DB Connection.

5-2-2 Spooling System

The following figure shows the spooling system.



a. When a failure occurred in information exchange between DB Connection Service and DB, the unsent SQL statements are automatically stored in the Spool memory (a dedicated area for the Spool function for an NX-series Controller and the EM Area of the memory used for CJ-series CPU Units for an NJ-series Controller).

b. When communications are recovered from the failure and the DB is reconnected, the SQL statements in the Spool memory are resent automatically or by executing an instruction.

5-2-3 Applicable Instructions and Spooling Execution Conditions

Applicable Instructions

The following two instructions are applicable to this function.

- DB_Insert (Insert DB Record) instruction
- DB_Update (Update DB Record) instruction



Precautions for Correct Use

Only the processing for inserting or updating records is spooled. For the other processing, you need to execute the instruction again.

Spooling Execution Conditions

SQL statements are spooled in the following cases.

- When an applicable instruction is executed, the SQL statement cannot be sent due to a network failure.
- When an applicable instruction is executed, the response from the DB cannot be received due to a network failure.
- When an applicable instruction is executed, the DB is stopped due to a server's problem or other causes.
- When an applicable instruction is executed, one or more SQL statements are already stored in the Spool memory.
- When an applicable instruction is executed, a DB Connection Instruction Execution Timeout occurs.



Precautions for Correct Use

- The following error codes are applicable to the spooling execution conditions when the instructions end in an error. When the instructions end in an error with other error codes, the SQL statement is not stored in the Spool memory.
3011 hex: DB Connection Disconnected Error Status
3012 hex: DB Connection Instruction Execution Timeout
3014 hex: Data Already Spooled
3016 hex: DB in Process
- If an instruction error (SQL Execution Error) occurs, the transmitted SQL statement itself can be the cause of the SQL Execution Error. Therefore, the SQL statement is not stored in the Spool memory because the SQL Execution Error may occur again when the SQL statement is resent.
- Even if a response cannot be received from the DB, the transmitted SQL statement may have been processed in the DB.

5-2-4 Memory Area Used by the Spool Function

The following provides the memory areas that are used by the Spool function. The memory area differs for the NX-series Controllers and NJ-series Controllers.

NX701-□□20 and NX102-□□20

The following memory area is used by the Spool function.

Memory area	Description	
Dedicated area for the Spool function	• The unsent SQL statements are stored in the dedicated area for the Spool function.	• Total capacity of Spool memory: NX701-□□20: 2 MB max. NX102-□□20: 192 KB max. • Spool capacity for each DB Connection: Total capacity is equally divided by DB Connections for which the Spool function is enabled.

You can prevent losing the SQL statements stored in the Spool memory even if a power interruption occurred in the CPU Unit because the dedicated area for the Spool function is non-volatile memory.



Precautions for Correct Use

- The data in the dedicated area for the Spool function is retained by a battery. If the battery is not mounted or weak, the CPU Unit detects a Battery-backup Memory Check Error. In that case, the Spool data is cleared.
- The spool data will be cleared in the following cases:
 - a) When "Use" is selected in the "Spool Settings" and the project is downloaded. In this case, the spool data will be cleared regardless of the "Spool Settings" of the project to be downloaded.
 - b) When restoring backup data.

NJ501-□□20 and NJ101-□□20

The following memory area is used by the Spool function.

Memory area	Description	
Memory used for CJ-series Units (EM)	• The unsent SQL statements are stored in the following EM Area of the memory used for CJ-series Units. EM Banks: NJ501-□□20: 16 EM banks from No. 9 hex to 18 hex. NJ101-□□20: 3 EM banks from No. 1 hex to 3 hex.	• Total capacity of Spool memory: NJ501-□□20: 1 MB max. NJ101-□□20: 192 KB max. • Spool capacity for each DB Connection: Total capacity is equally divided by DB Connections for which the Spool function is enabled.

You can prevent losing the Spool data even if a power interruption occurred in the CPU Unit because the EM Area of the memory used for CJ-series Units is non-volatile memory.

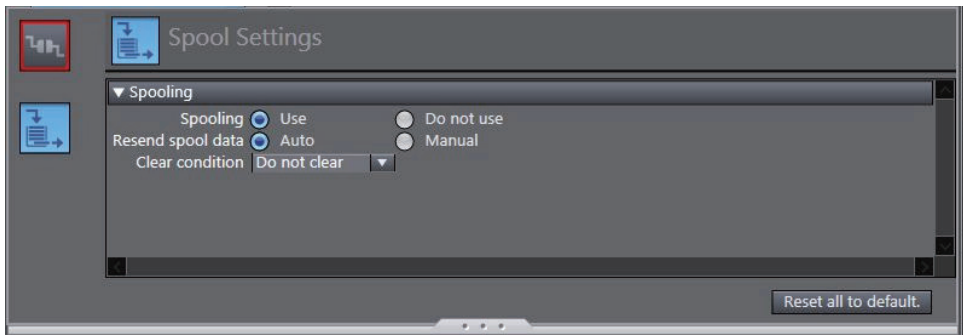


Precautions for Correct Use

- When the Spool function is enabled, the DB Connection Service uses EM Banks. Please design the system so that the EM Banks used by the DB Connection Service are not used for the following purposes because the Spool data is corrupted if used.
 - a) AT specification of user-defined variables
 - b) I/O memory address specification of tags for tag data link
 - c) Access by communications commands
 - d) Access from HMI
 - e) Specification of "Expansion Area words allocated to Special Units" for CJ-series Special Units
- The data values in the EM Area of the memory used for CJ-series Units are retained by a battery. If the battery is not mounted or weak, the CPU Unit detects a Battery-backup Memory Check Error. In that case, the Spool data is cleared.
- In the "DB Connection settings", the default setting of "Spooling" is "Use". If you do not use the Spool function, be sure to set "Spooling" to "Do not use" in the Spool Settings of the DB Connection settings and then download the DB Connection settings when you add a DB Connection. If you download the DB Connection settings while Spooling is set to "Use", the values stored in the EM banks used by the DB Connection Service will be overwritten by the initialization processing of the Spool function.
- If you select "DM, EM and Holding Memory used for CJ-series Units" for the memory type when backing up or restoring variables or memory on Sysmac Studio, the Spool data will be also backed up or restored. If you don't need the Spool data after executing a restore operation, clear the SQL statements from the Spool memory. Refer to 5-2-7 *Clearing the SQL Statements from the Spool Memory* on page 5-9 for the procedure.

5-2-5 Spool Function Settings

Right-click a DB Connection name under **Configurations and Setup - Host Connection Settings - DB Connection - DB Connection Settings** in the Multiview Explorer and select **Edit** from the menu. Set the **Spool function** in the **Spool Settings**.



Set the following items for the Spool function.

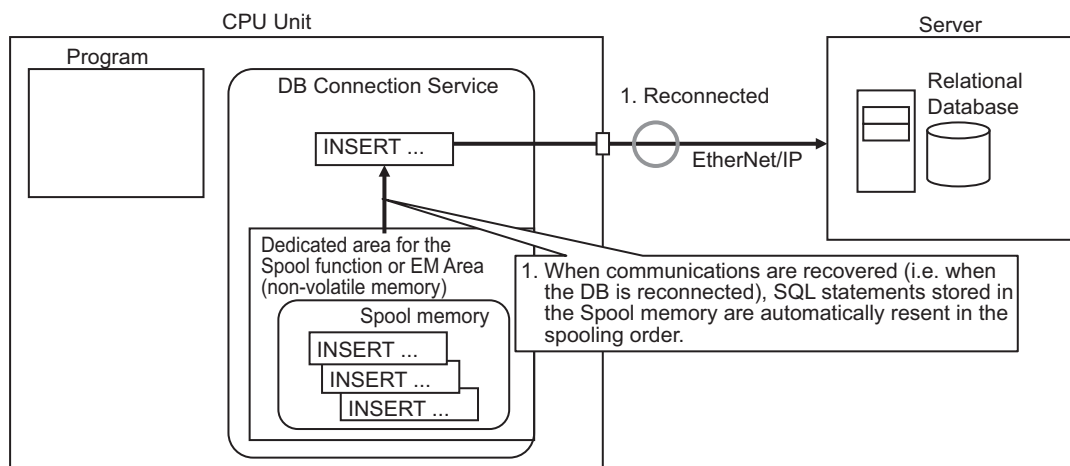
Item	Description	Values
Spooling	Set whether to use the Spool function.	• Use (Default) • Do not use
Resend spool data	Set this item when you select "Use" for "Spooling". Set whether to resend the SQL statements stored in the Spool memory automatically or manually.	• Auto (Default) • Manual
Clear condition	Set this item when you select "Auto" for "Resend spool data". Set the condition for clearing the SQL statements from the Spool memory.	• Do not clear (Default) • At power ON • When DB connection service started • When DB connection established

5-2-6 How to Resend the SQL Statements Stored in the Spool Memory

You can resend the SQL statements stored in the Spool memory automatically or manually, which can be selected in the "Resend Spool Data" of the Spool Settings.

Auto Resend

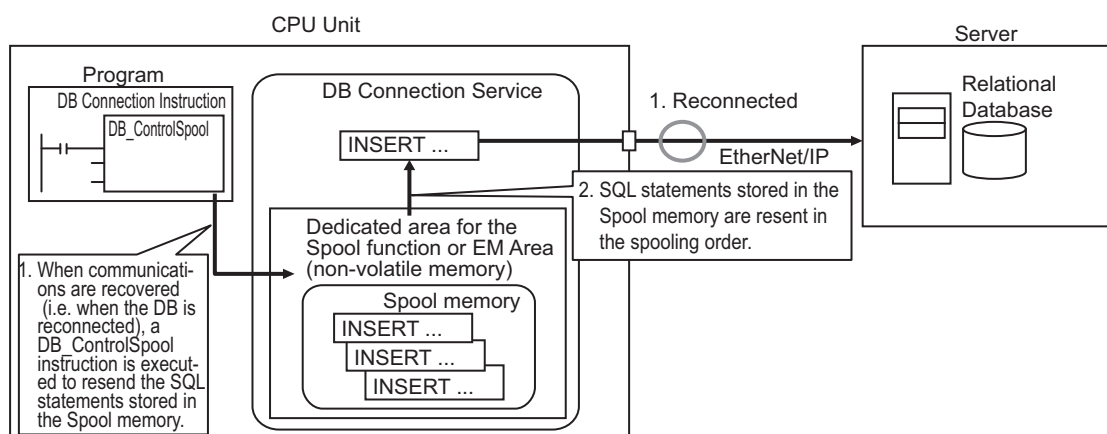
The SQL statements stored in the Spool memory are automatically resent when the DB is reconnected.



Manual Resend

The SQL statements stored in the Spool memory are resent when a DB_ControlSpool (Resend/Clear Spool Data) instruction is executed.

All of the SQL statements stored in the Spool memory are sent in the spooling order by one execution of the DB_ControlSpool (Resend/Clear Spool Data) instruction.



If a Failure Occurred in Information Exchange with the DB when Resending the SQL Statements

If a failure occurred again when the SQL statements stored in the Spool memory are resent, the unsent SQL statements are kept in the Spool memory. The SQL statements are resent again by auto resend or manual resend. The resend order is not changed.

5-2-7 Clearing the SQL Statements from the Spool Memory

The SQL statements are cleared from the Spool memory in the following cases.

- When the specified clear condition is met.
- When a DB_ControlSpool (Resend/Clear Spool Data) instruction is executed
- When the Clear Spool Data operation is executed from Sysmac Studio
- When the automatic clear condition is met



Version Information

If the version of the DB connection service is Ver.1.04 or higher, *Spool Cleared (Information)* will be registered in the event log once the spooled SQL statements are cleared.

When the Specified Clear Condition is Met

When Auto is selected for Resend Spool Data in the Spool Settings, you can set the condition for clearing the SQL statements from the Spool memory for each DB Connection in **Clear condition** under **DB Connection Settings - Spool Settings** on Sysmac Studio. Select from the following options.

Clear condition	Description
Do not clear (Default)	The SQL statements stored in the Spool memory are not cleared.
At power ON	The SQL statements are cleared from the Spool memory when the power supply to the CPU Unit is turned ON.
When DB connection service started	The SQL statements are cleared from the Spool memory when the DB Connection Service is started.
When DB connection established	The SQL statements are cleared from the Spool memory when the DB Connection is established (i.e. when the status changes from Closed to Connected). If you select this option, the SQL statements are cleared from the Spool memory without being resent.

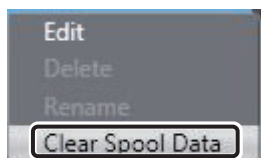
When a DB_ControlSpool (Resend/Clear Spool Data) Instruction is Executed

You can clear the SQL statements from the Spool memory by executing a DB_ControlSpool (Resend/Clear Spool Data) instruction.

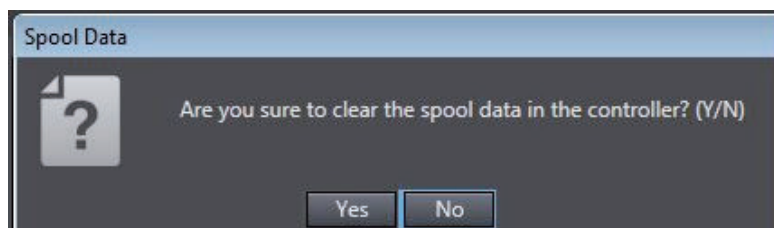
When the Clear Spool Data Operation is Executed from Sysmac Studio

You can clear the SQL statements from the Spool memory by the following operation from Sysmac Studio.

- 1 Right-click a DB Connection in the Multiview Explorer and select **Clear Spool Data** from the menu while online with an NJ/NX-series CPU Unit.



The following message is displayed.



2 Click the **Yes** Button.

When the Automatic Clear Condition is Met

The SQL statements are automatically cleared from the Spool memory regardless of the Resend spool data setting in the following cases.

- When you execute the Clear All Memory operation
- When a "Battery-backup Memory Check Error" occurred
- When you execute the Restore operation of the SD Memory Card backup function or Sysmac Studio Controller backup function.
- When you restore the memory using the Restore Variables/Memory function of Sysmac Studio
- When the Synchronization (download) operation is executed on Sysmac Studio (Note)

Note However, even the Synchronization (download) operation is executed when the operation status of the DB Connection Service is shutdown (Shutdown is TRUE), the spooled data is not cleared. To clear the data, disconnect the database physically such as disconnecting an Ethernet cable after synchronization, and then cycle the power supply to the CPU Unit. Then, execute the **Clear Spool Data** by online operation from Sysmac Studio.



Version Information

If the version of the DB connection service is Ver.1.03 or lower, the spooled SQL statements will not be cleared automatically when the Synchronization (download) operation is executed without modifying the DB connection settings.

5-2-8 Relationship with the DB Connection Instructions

This section describes the operations of DB Connection Instructions to be performed when one or more SQL statements are already stored in the Spool memory and the impacts to the spooling operations to be performed when an Instruction Execution Timeout occurred for a DB Connection Instruction.

Executing DB Connection Instructions when SQL Statements are Already Stored in the Spool Memory

This section describes the operation to be performed when each DB Connection Instruction is executed for a DB Connection that already has one or more SQL statements in the Spool memory.

Instruction	Operation
DB_Insert (Insert DB Record)	The SQL statement (INSERT) is spooled.*1 The instruction ends in an error. (Error = TRUE, SendStatus = _DBC_SEND_SPOOLED) Refer to <i>Section 7 DB Connection Instructions</i> on page 7-1 for ErrorID of the instruction execution error.
DB_Update (Update DB Record)	The SQL statement (UPDATE) is spooled.*1 The instruction ends in an error. (Error = TRUE, SendStatus = _DBC_SEND_SPOOLED) Refer to <i>Section 7 DB Connection Instructions</i> on page 7-1 for ErrorID of the instruction execution error.
DB_Select (Retrieve DB Record)	The SQL statement (SELECT) is not sent to the DB. An instruction execution error occurs. (Error = TRUE) Refer to <i>Section 7 DB Connection Instructions</i> on page 7-1 for ErrorID of the instruction execution error.
DB_Delete (Delete DB Record)	The SQL statement (DELETE) is not sent to the DB. An instruction execution error occurs. (Error = TRUE) Refer to <i>Section 7 DB Connection Instructions</i> on page 7-1 for ErrorID of the instruction execution error.

*1. If the remaining Spool memory area is not enough when the SQL statement is spooled, the SQL statements will be discarded without being stored in the Spool memory.

Instruction	Operation
DB_Insert (Insert DB Record)	The SQL statement (INSERT) is not sent to the DB. An instruction execution error occurs. (Error = TRUE, SendStatus=_DBC_SEND_SENDING) Refer to <i>Section 7 DB Connection Instructions</i> on page 7-1 for ErrorID of the instruction execution error.
DB_Update (Update DB Record)	The SQL statement (UPDATE) is not sent to the DB. An instruction execution error occurs. (Error = TRUE, SendStatus=_DBC_SEND_SENDING) Refer to <i>Section 7 DB Connection Instructions</i> on page 7-1 for ErrorID of the instruction execution error.

Operations of Instructions and DB Connection Service in the Case of DB Connection Instruction Execution Timeout

If the Spool function is enabled, the transmitted SQL statement is stored in the Spool memory when a DB Connection Instruction Execution Timeout occurs.

The DB Connection Service waits for a response from the DB for the time set in the Query execution timeout parameter plus 10 seconds*1 after the DB Connection Instruction is executed.

When a response is returned from the DB, the SQL statement stored in the Spool memory is deleted. If no response has been returned from the DB when the time set in the Query execution timeout parameter plus 10 seconds*1 has elapsed, the DB Connection is changed to the "Disconnected" status.

When the record processing instruction or the stored procedure instruction is executed while the DB Connection Service is waiting for a response from the server-side database, the instruction will end abnormally (DB in Process).

**Precautions for Correct Use**

If the time set in the Query execution timeout parameter has elapsed after execution of a DB Connection Instruction, a cancel request of the applicable SQL operation is sent to the DB. The details of the SQL operation cancel processing are given below.

1. When the cancel processing is completed within 10 seconds*¹:
 - The instruction will be terminated due to an error (SQL Execution Error).
2. When the cancel processing is not completed within 10 seconds*¹:
 - A communications timeout will occur. When the communications timeout has occurred, the instruction will be terminated due to an error (DB Connection Disconnected Error Status) and the DB Connection is changed to the "Disconnected" status.
 - In the case of DB_Insert (Insert DB Record) or DB_Update (Update DB Record) instruction, the SQL statement is stored in the Spool memory.
 - If resending of Spool data and disconnection of DB Connection occur repeatedly, increase the time set in the Query execution timeout parameter or review the SQL operation to make an adjustment so that the communications timeout does not occur. Refer to 5-7 *Timeout Monitoring Functions* on page 5-33 for timeout monitoring.

*1. The time differs by the DB type and DB status.

● DB_Insert (Insert DB Record) or DB_Update (Update DB Record) Instruction

If the Spool function is enabled, the SQL statement to send is spooled.

Regardless of the Resend spool data setting, the spooled SQL statement is sent after the response to the previous DB Connection Instruction is returned.

● Record Processing Instructions other than DB_Insert (Insert DB Record) and DB_Update (Update DB Record) Instructions

To execute the record processing instructions other than DB_Insert (Insert DB Record) and DB_Update (Update DB Record) instructions after a response from the previously executed DB Connection instruction is returned, write a user program so that the instructions are retried until they normally complete.

5-2-9 How to Estimate the Number of SQL Statements that can be Spooled

The number of SQL statements that can be spooled depends on the user program.

This section describes how to estimate the number of SQL statements that can be spooled.

Calculation of the Number of Bytes of each SQL Statement

The method of calculating the number of bytes for each SQL statement varies by the version of the DB Connection Service. For details of the procedure to check the version of the DB Connection Service, refer to *Versions* on page 21.

You can check the contents of SQL statements with the Debug Log.

Refer to 6-3 *Debug Log* on page 6-15 for the information on the Debug Log.

● For DB Connection Service version 1.04 or higher

Instruction	SQL statement	Calculating formula of the number of bytes of each SQL statement*1
DB_Insert (Insert DB Record)	<TableName><DBMapVariableName><DBMapVariableValue>	38 + (Number of bytes of <TableName>) + (Number of bytes of <DBMapVariableName>) + (Number of bytes of <DBMapVariableValue>)
DB_Update (Update DB Record)	<TableName><DBMapVariableName><DBMapVariableValue> <RetrievalCondition>	38 + (Number of bytes of <TableName>) + (Number of bytes of <DBMapVariableName>) + (Number of bytes of <DBMapVariableValue>) + (Number of bytes of <RetrievalCondition>)

*1. Text strings of SQL statements are handled as UTF-8. One byte is used for each single-byte alphanumeric character and multiple bytes are used for each multi-byte character. Three bytes are used for each Japanese character as a guide.

● For DB Connection Service version 1.03 or lower

Instruction	SQL statement	Calculating formula of the number of bytes of each SQL statement*1
DB_Insert (Insert DB Record)	insert into <TableName> (<ColumnName1>, <ColumnName2>, <ColumnName3>..., <ColumnNameN>) values(<Value1>, <Value2>, <Value3>..., <ValueN>)	50 + (Number of bytes of <TableName>) + (Number of bytes of <ColumnName1>) + (2 + Number of bytes of <ColumnName2>) + (2 + Number of bytes of <ColumnName3>) ... + (2 + Number of bytes of <ColumnNameN>) + (Number of bytes of <Value1>) + (2 + Number of bytes of <Value2>) + (2 + Number of bytes of <Value3>) ... + (2 + Number of bytes of <ValueN>)
DB_Update (Update DB Record)	update <TableName> set <ColumnName1>=<Value1>, <ColumnName2>=<Value2>..., <ColumnNameN>=<ValueN> where <RetrievalCondition>	45 + (Number of bytes of <TableName>) + (3 + Number of bytes of <ColumnName1> + Number of bytes of <Value1>) + (5 + Number of bytes of <ColumnName2> + Number of bytes of <Value2>) + (5 + Number of bytes of <ColumnName3> + Number of bytes of <Value3>) ... + (5 + Number of bytes of <ColumnNameN> + Number of bytes of <ValueN>) + (Number of bytes of <RetrievalCondition>)

*1. Text strings of SQL statements are handled as UTF-8. One byte is used for each single-byte alphanumeric character and multiple bytes are used for each multi-byte character. Three bytes are used for each Japanese character as a guide.

Calculation of the Number of SQL Statements that Can be Spooled

You can estimate the number of SQL statements that can be spooled using the following formulae.

Number of SQL statements that can be spooled =

Pool capacity per DB Connection (bytes) ÷ Number of bytes of each SQL statement

Pool capacity per DB connection (bytes) =

Capacity of the entire Spool memory (2,097,152 bytes for NX701-□□20, 1,048,576 bytes for NJ501-□□20, or 196,608 bytes for NX102-□□20 and NJ101-□□20) ÷ Number of DB Connections for which the Spool function is enabled

5-3 Stored Procedure Call Function

This section describes the stored procedure call function.

The stored procedure call function is available for version 2.00 or higher of the DB Connection Service.

For details on the relationship between the DB Connection Service versions and the DB Connection function, refer to *A-4 Version Information* on page A-26.

5-3-1 Overview

The stored procedure call function enables the controller to call stored procedures or stored functions defined in the other databases inside the network.

With this function, complex arithmetic processing can be performed on the server side. This function not only simplifies the user programming for the controller but also shortens the processing time by using the server with high computing capacity to perform processing, ensures the data consistency on the server side, and enables operations that are difficult to realize with the INSERT, UPDATE, and DELETE instructions.

You can use the stored procedure call function with the DB Connection Instructions. Under normal circumstances, DB Connection Instructions are called in the order of DB_AttachProcedure, DB_ExecuteProcedure, and DB_DetachProcedure.

Refer to *1-2-2 DB Connection System* on page 1-11 for details on how the stored procedure call function works.

The specification overview of the stored procedure call function is described below.

- The overview contains terminology that is related to stored procedures.
For the terminology on stored procedures, refer to the manual of the corresponding database provider.
- Refer to *5-3-2 Specifications of the Stored Procedure Call Function for Databases* on page 5-17 for details on the specifications of each database type.

Item	Specifications
Supported database type	SQL Server Oracle Database MySQL Community Edition PostgreSQL
Argument of stored procedure	Up to 256 variables (vary by a single structure member or an array variable of basic data type) • If you use the argument, use the <i>ArgIn</i>, <i>ArgOut</i> and <i>ArgInOut</i> input variables for the DB_AttachProcedure (Generate DB Stored Procedure Handle) instruction. Refer to <i>DB_AttachProcedure (Generate DB Stored Procedure Handle)</i> on page 7-107 for details. • If you omit the argument, assign the <i>_DBC_Used</i> system-defined variable to the input variable.
Return value of stored procedure	One variable (basic data type) • If you use the return value, use the <i>ReturnVal</i> input variable for the DB_AttachProcedure (Generate DB Stored Procedure Handle) instruction. Refer to <i>DB_AttachProcedure (Generate DB Stored Procedure Handle)</i> on page 7-107 for details. • If you omit the return value, assign the <i>_DBC_Used</i> system-defined variable to the input variable.

Item	Specifications
Result set of stored procedure	Up to 256 x 65535 variables or equivalent (vary by the structure array variables stored in up to 256 structure definition members) - If you use the result set, use the <i>ResultSet</i> input variable for <i>DB_AttachProcedure (Generate DB Stored Procedure Handle)</i> instruction. Refer to <i>DB_AttachProcedure (Generate DB Stored Procedure Handle)</i> on page 7-107 for details. - if you omit the result set, assign the <i>_DBC_Unused</i> system-defined variable to the input variable.
Spool function	Not supported



Precautions for Correct Use

Before you execute the stored procedure call function, make sure to verify the name of the stored procedure to execute, the processing details, and the argument values. When you use the Operation Logs, you can check an error that occurs during execution. Refer to *Section 6 How to Use Operation Logs* on page 6-1 for details on the Operation Logs.

5-3-2 Specifications of the Stored Procedure Call Function for Databases

This section describes the specifications of the stored procedure call function for each database type. For the data types of stored procedure arguments, return values, and result sets that are supported by the NJ/NX-series Controllers per database type, refer to *Correspondence of Data Types between NJ/NX-series Controllers and DB* on page 3-4.

The description contains terminology that is related to stored procedures.

For the terminology on stored procedures, refer to the manual of the corresponding database provider.



Precautions for Correct Use

- Stored procedure names are case-sensitive.
- The prohibited characters for the stored procedure names conform to the specifications of each database type.
- The length of a stored procedure name should be no more than the data size of the procedure name that can be output to an Operation Log.

SQL Server

Item	Description
Supported stored procedure type	Stored procedures and scalar functions generated by Transact-SQL
Stored procedure name specification method	To specify a schema: [schema name].[procedure name] To skip a current schema: [procedure name]

Item	Description
Notes	- The argument name for a stored procedure in SQLServer needs to have '@' at the beginning, but the controller does not allow variable names to start with '@'. For this reason, make sure to use the same text string consisting of the stored procedure argument name without '@' for the structure member variable names that are specified in the <i>ArgIn</i>, <i>ArgOut</i>, <i>ArgInOut</i> and <i>ResultSet</i> input variables for the DB_AttachProcedure instruction. - Specify the variable, which is declared with an OUTPUT keyword as an argument by the stored procedure in SQL Server, to the <i>ArgOut</i> input variable of the DB_AttachProcedure instruction.

Oracle Database

Item	Description
Supported stored procedure type	Standalone procedures and packaged procedures generated by PL/SQL
Stored procedure name specification method	To specify a schema and specify a packaged procedure: [schema name].[package name].[procedure name] To specify a schema: [schema name].[procedure name] To skip a current schema and specify a packaged procedure: [package name].[procedure name] To skip a current schema (other than packaged procedures): [procedure name]
Notes	Set [schema name], [package name], and [procedure name] to be no more than 30 bytes respectively. To return the result set of a stored procedure, use the <i>ResultSet</i> input variable of the DB_AttachProcedure instruction and set the variable to a structure array for which result set needs to be received. On the stored procedure, prepare one OUT argument in the SYS_REFCURSOR type and assign the result set to the cursor.

MySQL Community Edition

Item	Description
Supported stored procedure type	Stored functions generated by PL/pgSQL
Stored procedure name specification method	To specify a database: [database name].[procedure name] To skip a current database: [procedure name]

PostgreSQL

Item	Description
Supported stored procedure type	- Standalone procedures (functions) generated by PL/SQL - Packaged procedures (functions)
Stored procedure name specification method	To specify a schema: [schema name].[procedure name] To skip a current schema: [procedure name]

Item	Description
Notes	- The SQL functions for PostgreSQL are not supported since argument names do not exist. - Argument names can be omitted for PL/pgSQL, but the stored functions with the argument names omitted are not supported. - For the result set, prepare one OUT argument in the REFCURSOR type and assign the result set for the corresponding cursor.

5-3-3 How to Execute the Stored Procedure Call Function

The stored procedure call function is executed by using a procedure handle and DB Connection Instructions. The description of a procedure handle, instructions to use, and the operation flow are given below.

What is a Procedure Handle?

A procedure handle is an identifier (handle) used for uniquely identifying a stored procedure. In the DB Connection Instructions, each stored procedure is distinguished by the procedure handle.

Instructions to Use

The following instructions are used. For details of each instruction, refer to the description of the corresponding instruction.

Instruction	Reference
DB_AttachProcedure	<i>DB_AttachProcedure (Generate DB Stored Procedure Handle)</i> on page 7-107
DB_ExecuteProcedure	<i>DB_ExecuteProcedure (Execute DB Stored Procedure)</i> on page 7-112
DB_DetachProcedure	<i>DB_DetachProcedure (Release DB Stored Procedure Handle)</i> on page 7-124

Operation Flow

The operation flow is described below. This operation flow is same for all the database types. For details on the execution processing and input/output variables, refer to the description of the corresponding instruction.

Step	Execution processing	Instruction	Reference	Remarks
1	Establish connection with a database	DB_Connect	<i>DB_Connect (Establish DB Connection)</i> on page 7-6	
2	Obtain a procedure handle	DB_AttachProcedure	<i>DB_AttachProcedure (Generate DB Stored Procedure Handle)</i> on page 7-107	Required only when using the stored procedure call function
3	Call the stored procedure	DB_ExecuteProcedure	<i>DB_ExecuteProcedure (Execute DB Stored Procedure)</i> on page 7-112	
4	Release the procedure handle	DB_DetachProcedure	<i>DB_DetachProcedure (Release DB Stored Procedure Handle)</i> on page 7-124	
5	Disconnect the database	DB_Close	<i>DB_Close (Close DB Connection)</i> on page 7-10	

5-3-4 Specifying the Table and Applying the Mapping

Before you execute the stored procedure call function, you need to map DB Map Variables to a data-base.

This section describes how to create and clear a DB mapping for calling a stored procedure, as well as the restrictions.

Overview

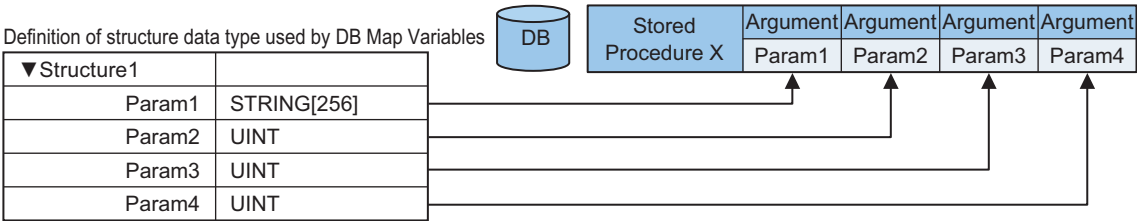
DB Map Variables are mapped to a database using the DB_AttachProcedure (Generate DB Stored Procedure Handle) instruction.

In the DB_AttachProcedure (Generate DB Stored Procedure Handle) instruction, specify "stored procedure name", "arguments", "return value", and "result set", etc. By doing so, DB Map Variables can be mapped to a database.

Refer to *DB_AttachProcedure (Generate DB Stored Procedure Handle)* on page 7-107 for details.

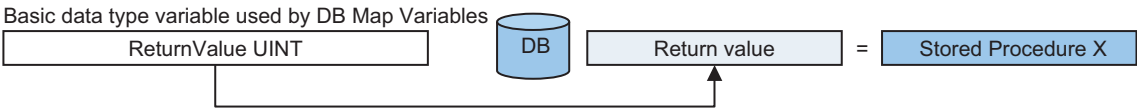
Mapping to Arguments

The mapping to arguments is illustrated below.



Mapping to a Return Value

The mapping to a return value is illustrated below.

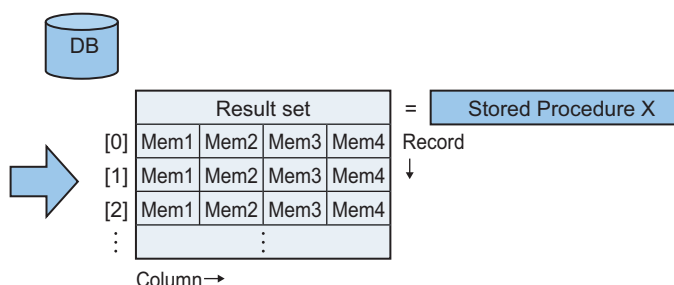


Mapping to Result Sets

The mapping to result sets is illustrated below.

Definition of structure data type used by
DB Map Variables

▼ Structure1[0]	
Mem1	STRING[256]
Mem2	UINT
Mem3	UINT
Mem4	UINT
▼ Structure1[1]	
Mem1	STRING[256]
Mem2	UINT
Mem3	UINT
Mem4	UINT
▼ Structure1[2]	
Mem1	STRING[256]
Mem2	UINT
Mem3	UINT
Mem4	UINT
⋮	



Clearing the Mapping of DB Map Variables

With the stored procedure call function, the DB Map Variables mapped to a database are cleared when the DB_DetachProcedure instruction is executed.

Refer to 3-4-2 *Clearing the Mapping of DB Map Variables* on page 3-20 for details on the cases where mapping is cleared by any other operation.

Relationship Between Stored Procedure Elements and DB Map Variables

The following table shows the relationship between stored procedure elements and DB Map Variables.

Stored procedure elements	Specifications of the DB_AttachProcedure instruction	Remarks
Name	It must be the text string indicating the stored procedure name.	Refer to 5-3-2 <i>Specifications of the Stored Procedure Call Function for Databases</i> on page 5-17.
Argument	It must be a structure variable. The structure member name must be consistent with the argument name. The total number of structure members specified in the IN, INOUT, and OUT arguments of the input variables for a stored procedure must be 256 or less.	The order of structure members can be modified. An error occurs if the number of structure members does not match. Refer to Precautions for Correct Use in 3-2-2 <i>Specifications of Structure Data Type for DB Access</i> on page 3-3 for the characters that cannot be specified for structure member names.
Return value	It must be a variable in the basic data type. * Derivative data type variables (arrays and structures) cannot be used.	

Stored procedure elements	Specifications of the DB_AttachProcedure instruction	Remarks
Result set	It must be a structure variable or a structure array variable. The total number of structure members must be 256 or less. The number of array elements allowed is 65535 elements, which is the upper limit of array variables for the system.	An error occurs if the number of structure members does not match. Refer to Precautions for Correct Use in 3-2-2 <i>Specifications of Structure Data Type for DB Access</i> on page 3-3 for the characters that cannot be specified for structure member names.

5-3-5 Errors during Stored Procedure Call

This section describes errors that could occur during a stored procedure call.

The following shows an example of the stored procedure (procedure name "proc1"), the end positions including exceptions during proc1 execution, the causes of termination, as well as the execution result and troubleshooting of the DB_ExecuteProcedure instruction for each position and cause of termination.

Example of the stored procedure (stored procedure name "proc1")

Outline of the processing	End positions and causes of termination
<pre>CREATE PROCEDURE proc1 Processing 1 BEGIN TRY IF (pre-execution check) BEGIN Processing 2 RETURN 0 END ELSE BEGIN RETURN 1 END END TRY BEGIN CATCH Exception handling RETURN 2 END CATCH;</pre>	1. Processing 1 ended by an exception outside of the TRY-Statement 2. An exception occurred in Processing 2 inside the TRY-Statement 3. Processing normally completed in Processing 2 inside the TRY-Statement 4. Processing ended in the error handling during pre-execution check 5. Processing ended by an exception in the error handling inside the CATCH-Statement 6. Processing ended in the error handling inside the CATCH-Statement 7. No response is returned within the query execution timeout period

Execution result and troubleshooting of the DB Connection Instruction per position and cause of termination

* No. corresponds to the number of *End positions and causes of termination* column in the above chart.

No.	Positions and causes of termination	DB_ExecuteProcedure instruction execution result			Return value	Troubleshooting
		Done	Error	ErrorID		
1	Ended by an exception in Processing 1 outside of the TRY-Statement	False	True	0x300B	Unchanged (same value as before executing the instruction)	- Check the output variable <i>Connection Status</i> by executing the DB_GetConnectionStatus instruction and implement a process that will execute retry according to the database status in the user program. - Check the failure log
2	An exception occurred in Processing 2 inside the TRY-Statement: Go to the CATCH-Statement	---	---	---	---	---
3	Ended normally in Processing 2 inside the TRY-Statement	True	False	0x0000	0	Implement a process for each return value in the user program
4	Ended when the pre-execution check failed	True	False	0x0000	1	Implement a process for each return value in the user program
5	Ended by an exception in the exception handling inside the CATCH-Statement	False	True	0x300B	Unchanged (same value as before executing the instruction)	- Check the output variable <i>Connection Status</i> by executing the DB_GetConnectionStatus instruction and implement a process that will execute retry according to the database status in the user program. - Check the failure log
6	Ended in the exception handling inside the CATCH-Statement	True	False	0x0000	2	Implement a process for each return value in the user program
7	No response is returned within the query execution timeout period	False	True	0x300B	Unchanged (same value as before executing the instruction)	- Check the output variable <i>Connection Status</i> by executing the DB_GetConnectionStatus instruction and implement a process that will execute retry according to the database status in the user program. - Check the failure log
Errors that are not listed above, such as an error in the communication path		False	True	Other than 0x300B	Unchanged (same value as before executing the instruction)	Abnormal end processing is performed based on the ErrorID details

5-4 Batch Insert Function

This section describes the batch insert function.

The batch insert function is available for the DB Connection Service version 2.00 or higher.

For details on the relationship between the DB Connection Service versions and the DB Connection function, refer to *A-4 Version Information* on page A-26.

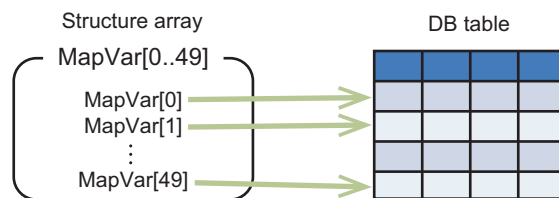
5-4-1 Overview

The batch insert function is used to insert multiple records at once.

By using the batch insert function, the execution time becomes shorter than when executing the insert processing several times.

You can use the batch insert function with the DB Connection Instructions.

The following figure illustrates data preparation and execution processes for the batch insert function.



One structure array element is inserted as one record

- The *MapVar* variables shown in the figure are defined by a structure array.
- When the number of array elements is represented as 'n', the number of records corresponding to 'n' will be collectively inserted to a table mapped by the instruction.

The specifications of the batch insert function are specified below.

Item	Specifications
Supported data-base type	SQL Server Oracle Database MySQL Community Edition PostgreSQL
Supported data and sizes	For the batch insert function, a structure array is used for the data to be inserted collectively. The supported data and sizes for the structure array are defined below. • Maximum number of elements in the structure array: 65,535 • Number of structure members (= number of columns) in a structure array: 1,000 • Structure size (= size of a record) in a structure array: 16 KB If the above conditions are not fulfilled, the execution of the DB_CeateMapping instruction fails.
Spool function	Not supported
Debug Log	It outputs information but not the SQL statement.
SQL Execution Failure Log	The logs are output only when the SQL Execution Failure Log enabled in the DB Connection Service Settings and the <i>SQLFailLog</i> input variable for the DB_BatchInsert instruction is set to TRUE.

5-4-2 How to Execute the Batch Insert Function

The batch insert function is executed by using DB Connection Instructions. The following describes the instructions to use and the operation flow.

Instructions to Use

The following instructions are used. For details of each instruction, refer to the description of the corresponding instruction.

Instruction	Reference
DB_BatchInsert (DB Records Batch Insert)	<i>DB_BatchInsert (DB Records Batch Insert)</i> on page 7-94

Operation Flow

The operation flow is described below. This operation flow is same for all the database types. For details on the execution processing, refer to the description of the corresponding instruction.

Step	Execution processing	Instruction	Reference	Remarks
1	Establish connection with a database	DB_Connect	<i>DB_Connect (Establish DB Connection)</i> on page 7-6	
2	Create a DB mapping	DB_CreateMapping	<i>DB_CreateMapping (Create DB Map)</i> on page 7-13	
3	Set a value to the DB Map Variable			Write a user program and execute the process
4	Execute the batch insert	DB_BatchInsert	<i>DB_BatchInsert (DB Records Batch Insert)</i> on page 7-94	Required only when using the batch insert function
5	Disconnect the database	DB_Close	<i>DB_Close (Close DB Connection)</i> on page 7-10	

5-5 DB Connection Service Shutdown Function

This section describes the shutdown function of the DB Connection Service to prevent losing the Operation Log data.

Refer to 4-3-1 *Operation Status of the DB Connection Service* on page 4-7 for the information on the operation status of the DB Connection Service.

5-5-1 Overview

The DB Connection Service shutdown function (hereinafter called "shutdown function") is used to shut down the DB Connection Service after saving the Operation Log files into the SD Memory Card.

Execute the shutdown function before turning OFF the power supply to the CPU Unit. You can prevent losing the Operation Log data by executing the shutdown function.



Precautions for Correct Use

If the power supply to the CPU Unit is turned OFF without executing the shutdown function while the DB Connection Service is running, the contents of the Operation Logs cannot be guaranteed. The Operation Log files may be corrupted or the data may be lost.

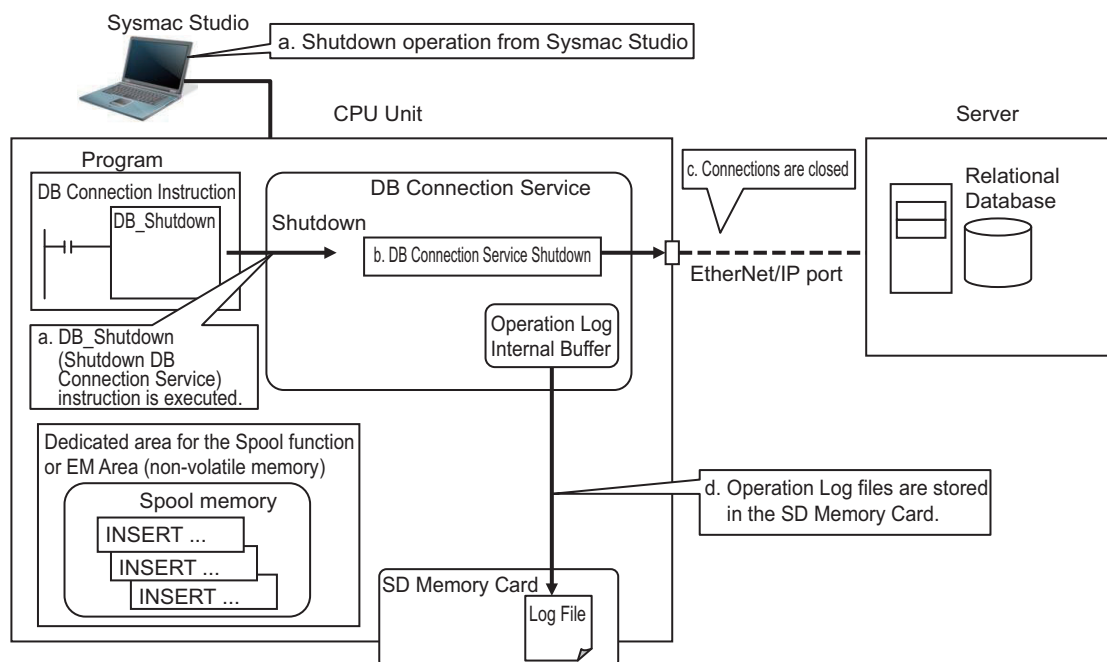


Additional Information

We recommend that you take countermeasures against power interruption such as installation of uninterruptible power supply system to prevent data loss by unexpected power interruption.

5-5-2 Shutdown System

The following figure shows the shutdown system.



- a. The DB Connection Service is shut down by a Sysmac Studio operation or by executing a DB_Shutdown (Shutdown DB Connection Service) instruction.
- b. The DB Connection Service is shut down.
- c. The DB Connections are closed.
- d. The Operation Log files (Execution Log files, Debug Log files, and SQL Execution Failure Log files) are stored in the SD Memory Card.

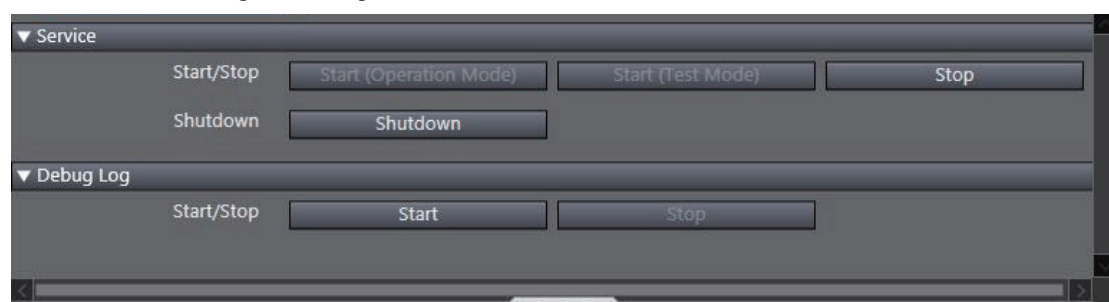
5-5-3 How to Execute the Shutdown Function

You can use the following procedure to execute the shutdown function.

- Sysmac Studio operation
- Instruction execution

Sysmac Studio Operation

Right-click **DB Connection Service Settings** under **Configurations and Setup - Host Connection Settings - DB Connection** in the Multiview Explorer and select **Online Settings** from the menu while online with an NJ/NX-series CPU Unit. Then, click the **Shutdown** Button under **Service - Shutdown** in the Online Settings Tab Page.



Additional Information

When you execute the "Reset Controller" operation on Sysmac Studio, the shutdown function is automatically executed before resetting the Controller.

Instruction Execution

Execute a DB_Shutdown (Shutdown DB Connection Service) instruction.

5-5-4 How to Check the Shutdown of the DB Connection Service

Confirm that the DB Connection Service has been shut down by the following methods before turning OFF the power supply to the CPU Unit.

- Checking with a system-defined variable
Confirm that `_DBC_Status.Shutdown` system-defined variable (Shutdown flag of the DB Connection Service Status) is TRUE.
- Checking by executing an instruction
- Confirm that the `Done` output variable of the DB_Shutdown (Shutdown DB Connection Service) instruction is TRUE.

5-6 How to Prevent Losing SQL Statements at Power Interruption

This section describes how to write the user program so as not to lose the SQL statements at power interruption.

5-6-1 Overview

By using the Spool function^{*1} and the user program, it is possible to avoid losing the SQL statements that the user attempted to transmit.

The workaround for preventing SQL statements from being lost is specified below.

	Workaround
Instructions with the Spool function ^{*1} supported (e.g. DB_Insert)	The problem can be avoided by configuring the Spool Settings to Use and by resending the statements by the user program. ^{*2}
Instructions without the Spool function ^{*1} supported (e.g. DB_BatchInsert)	The problem can be avoided by resending the statements by the user program. ^{*2}

*1. Refer to 5-2 *Spool Function* on page 5-5 for details on the Spool function.

*2. Refer to 5-6-2 *Procedures* on page 5-28 for details.

5-6-2 Procedures

Use the following procedures.

Checking the Progress of the DB Connection Instruction

The progress of the DB Connection Instructions is output to the SendStatus output variable as enumeration data. Use this data to create the user program.

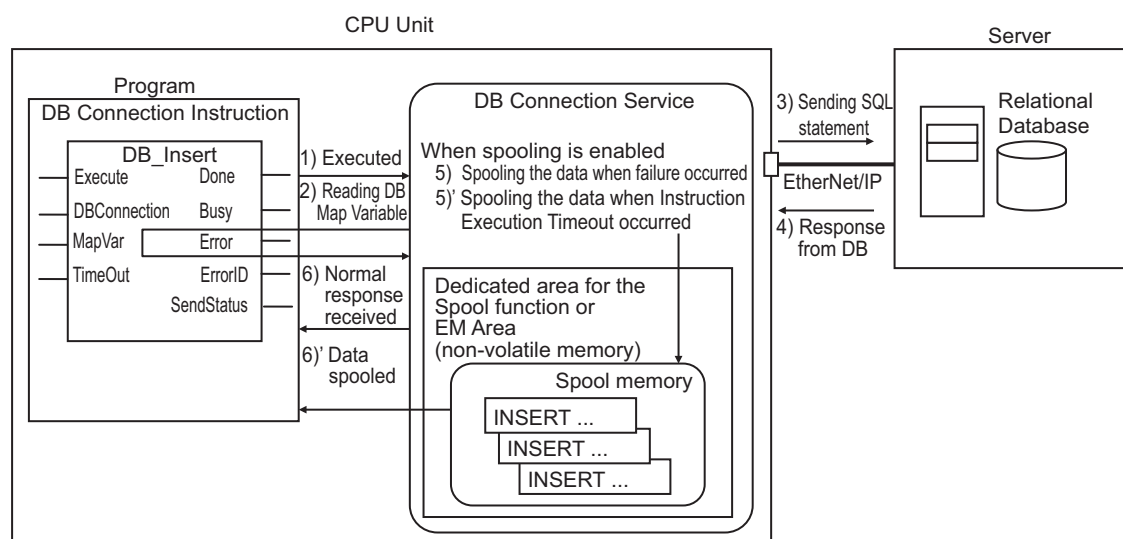
Output variable	Meaning	Data type	Description
SendStatus	Send Status	_eDBC_SEND_STATUS	_DBC_SEND_INIT(0): Initial status _DBC_SEND_UNSENT(1): SQL statement unsent _DBC_SEND_SENDING(2): Sending SQL statement _DBC_SEND_SPOOLED(3): SQL statement spooled _DBC_SEND_COMPLETE(4): SQL statement transmission completed

Variable Settings

- Set the Retain attribute of the input parameter (DB Map Variable) of the MapVar input variable to "Retained".
- Set the Retain attribute of the output parameter of the Busy output variable to "Retained".
- Set the Retain attribute of the output parameter of the SendStatus output variable to "Retained".

Necessary Actions against Power Interruption

You need to take an action against power interruption according to when power interruption occurs. This section describes the necessary actions using the following figure.



The numbers in the following table are corresponding to the numbers in the above figure.

Power interruption timing during execution of a DB Connection Instruction		Value of SendStatus output variable	Action
1) Executed (When instruction execution is started)	Until the DB Connection Service reads the present value of the DB Map Variable after Execute of the DB Connection Instruction changed from FALSE to TRUE	_DBC_SEND_SENDING: Sending SQL statement	Resend by user program
2) Reading DB Map Variable	Until the DB Connection Service sends the SQL statement to the DB after the service started reading the present value of the DB Map Variable		
3) Sending SQL statement	Until the transmission is completed since immediately before the DB Connection Service sends the SQL statement to the DB		
4) Response from DB	Until the response from DB is received after the SQL statement was sent to DB		
5) Spooling the data when failure occurred	While the SQL statement is being spooled because a failure has occurred (when spooling is enabled)		
5) Spooling the data when Instruction Execution Timeout occurred	While the SQL statement is being spooled because an Instruction Execution Timeout has occurred. (when spooling is enabled)	_DBC_SEND_COMPLETE: SQL statement transmission completed	Action not required
6) Normal response received	After normal response is received from the DB		

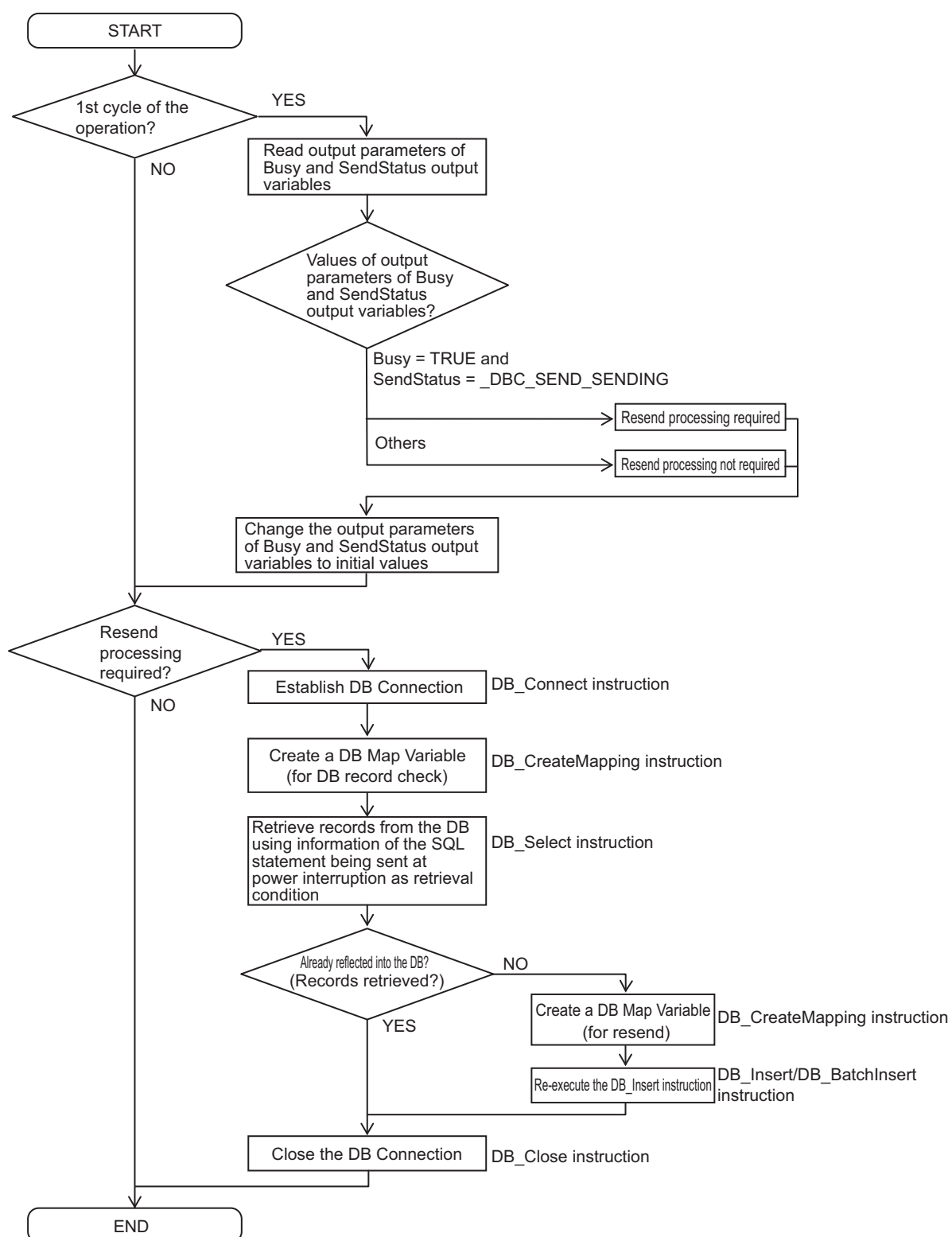
Power interruption timing during execution of a DB Connection Instruction		Value of SendStatus output variable	Action
6)' Data spooled	After the SQL statement is spooled (when spooling is enabled)	_DBC_SEND_SPOOLED: SQL statement spooled	Resend by Spool function (auto resend or manual re-send)

Resend Flow by User Program

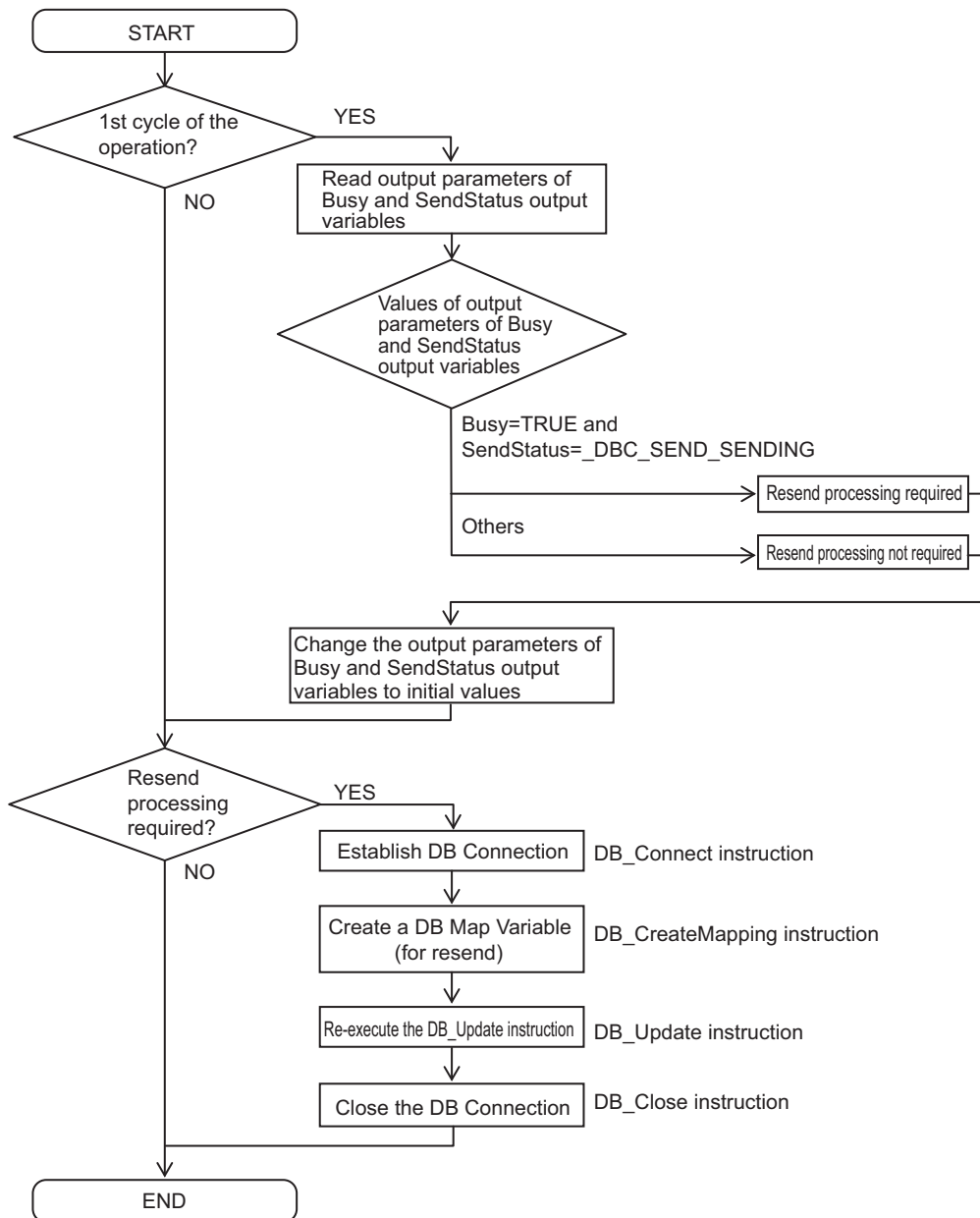
Write the user program to re-execute the instruction that is being executed at the time of power interruption.

The resend flow differs by whether a DB_Insert/DB_BatchInsert or DB_Update instruction is being executed at the time of power interruption.

● When a DB_Insert/DB_BatchInsert instruction is being executed



● When a DB_Update instruction is being executed



Precautions for Correct Use

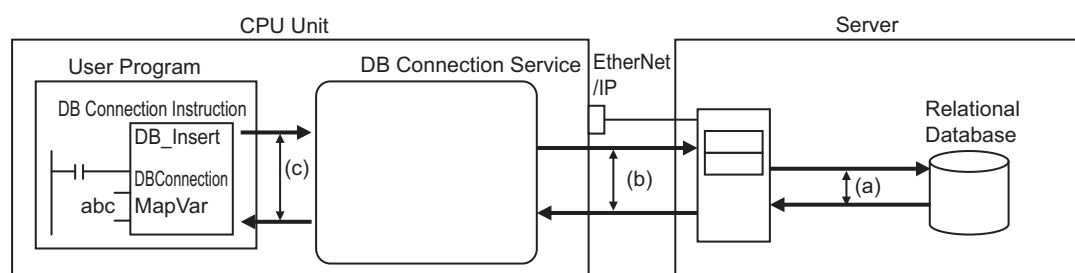
- The value of the SendStatus output variable is overwritten when the value of the Execute input variable is evaluated regardless of the value of the Execute input variable. Therefore, write the user program so that the value of the SendStatus output variable is read before evaluating the value of the Execute input variable of the DB Connection Instruction in the first cycle of the operation.
- The DB Connection Instruction is not executed if the Execute input variable is already TRUE at the operation start. You need to change the Execute input variable to FALSE to execute the instruction.

5-7 Timeout Monitoring Functions

This section describes timeout monitoring for the DB Connection Service.

5-7-1 Timeout Monitoring Functions

The following figure shows the types of timeouts that can be monitored.



Function name	Setting range	Description	Reference
Login timeout	1 to 60 seconds Default: 10 seconds	Time until the DB Connection Service detects a login failure due to a communications failure between DB Connection Service and DB or server's problem	2-2-2 DB Connection Settings on page 2-7
Query execution timeout ((a) in the above figure)	1 to 600 seconds Default: 30 seconds	Time until the DB Connection Service detects an error when the DB takes time for query execution. You can cancel the SQL operation when the DB takes longer than expected for query execution.	2-2-2 DB Connection Settings on page 2-7
Communications timeout ((b) in the above figure)	Time specified for Query execution timeout plus 10 seconds**1	Time until the DB Connection Service detects an error due to a communications failure between DB Connection Service and DB	---
Instruction execution timeout ((c) in the above figure)	Not monitored, or 0.05 to 180 seconds Default: Not monitored	Time until the DB Connection Service detects an error when the execution time of the record processing and stored procedure instructions becomes longer due to a communications failure between the DB Connection Service and the database, or the server-side problems or the load being applied. You can use this when you do not want to extend the takt time (i.e., lower the equipment performance).	A-2-4 Ensuring Equipment Performance (Takt Time) by Monitoring Instruction Execution Time-out on page A-23
Keep Alive monitoring time	1 to 65535 seconds Default: 300 seconds	This function is used to check whether the server is normally connected. When you set this Keep Alive monitoring time, a communications failure can be detected even while the DB Connection Service is waiting for a response from the server because the DB is executing a query.	Refer to the NJ/NX-series CPU Unit Built-in EtherNet/IP Port User's Manual (Cat. No. W506).

*1. The time to detect a communications timeout differs by the DB type and DB status.

5-7-2 Login Timeout

The login timeout is monitored in the following cases.

- When connecting to a DB using a DB_Connect (Establish DB Connection) instruction
- When reconnecting to a DB while a DB Connection is in the "Disconnected" status

The following table shows the operation to be performed when a login timeout has occurred.

When the timeout occurred	DB Connection status after the timeout occurred	Instruction execution result
When executing a DB_Connect instruction	Closed	ErrorID = 3005 hex (DB Connection Failed)
When reconnecting to a DB	Disconnected	---

5-7-3 Query Execution Timeout

The query execution timeout is monitored in the following cases.

- When transmitting an SQL statement to the server-side database using the record processing and stored procedure instructions
- When resending an SQL statement stored in the Spool memory

The following table shows the operation to be performed when a query execution timeout has occurred.

When the timeout occurred	DB Connection status after the timeout occurred	Instruction execution result
When executing a DB_Insert, DB_Update, DB_ExecuteProcedure, or DB_BatchInsert instruction	Connected	ErrorID = 300B hex (SQL Execution Error) SendStatus = _DBC_SEND_COMPLETE The SQL statement is not stored in the Spool memory.*1
When executing a DB_Select or DB_Delete instruction	Connected	ErrorID = 300B hex (SQL Execution Error)
When resending Spool data	Connected	The SQL statement is not stored in the Spool memory again.*1

*1. If an instruction error (SQL Execution Error) occurs, the transmitted SQL statement itself can be the cause of the SQL Execution Error. Therefore, the SQL statement is not stored in the Spool memory because the SQL Execution Error may occur again when the SQL statement is resent.

5-7-4 Communications Timeout

The communications timeout is monitored in the following cases.

- When transmitting an SQL statement to the server-side database using the record processing and stored procedure instructions
- When resending an SQL statement stored in the Spool memory

The following table shows the operation to be performed when a communications timeout has occurred.

When the timeout occurred	DB Connection status after the timeout occurred	Spool function	Instruction execution result
When executing a DB_Insert, DB_Update, DB_Execute-Procedure, or DB_BatchInsert instruction	Disconnected	Enabled	ErrorID = 3011 hex (DB Connection Disconnected Error Status) SendStatus = _DBC_SEND_SPOOLED The SQL statement is stored in the Spool memory.
		Disabled	ErrorID = 3011 hex (DB Connection Disconnected Error Status) SendStatus = _DBC_SEND_SENDING
When executing a DB_Select or DB_Delete instruction	Disconnected	---	ErrorID = 3011 hex (DB Connection Disconnected Error Status) SendStatus = _DBC_SEND_SENDING
When resending Spool data	Disconnected	Enabled	The SQL statement is stored in the Spool memory again.

5-7-5 Instruction Execution Timeout

Refer to 5-2-8 *Relationship with the DB Connection Instructions* on page 5-11 for details on the instruction execution timeout.

5-7-6 Keep Alive Monitoring Time

Whether the server is normally connected is monitored while the DB Connection is in the "Connected" status.

When the connection to the server cannot be confirmed for the time set in the "Keep Alive monitoring time parameter plus 12 seconds" due to a communications failure or server's problem, the DB Connection is closed.

The DB Connection enters into the "Disconnected" status when a record processing instruction or a stored procedure instruction is executed after closing the DB Connection or when the Spool data is resent.

The keep-alive function operates as shown below in the DB Connection Service.

- Regardless of the "Keep Alive" setting, the function is always used.
- Regardless of the Linger option setting, the option is always "specified".

The operation to be performed after the DB Connection is closed by the keep-alive monitoring function is the same as the communications timeout. Refer to 5-7-4 *Communications Timeout* on page 5-34.



Precautions for Correct Use

- The Keep Alive monitoring time is a common setting to the built-in EtherNet/IP port. When you set the Keep Alive monitoring time, confirm that the operations of the following functions in the built-in EtherNet/IP port are not affected before changing the value.
- Socket service, FTP server function, communications with Sysmac Studio, FINS/TCP

5-8 Other Functions

This section describes the operation related to the DB Connection Service, including the backup and restore functions of the NJ/NX-series Controllers, verification of operation authority from Sysmac Studio, as well as encrypted communication.

5-8-1 Backup/Restore Function in the DB Connection Service

The backup function is used to back up the setting data in an NJ/NX-series Controller into an SD Memory Card or a computer.

And the restore function is used to restore the data from an SD Memory Card or a computer to the Controller.

Backup/Restore Function Data

The following table shows whether each data of the DB Connection Service can be backed up and restored by the Controller function.

Data		Backup/ Restore func- tion	Available op- erations	Remarks
DB Connec- tion Settings	DB Connection Service Settings DB Connection Settings	Supported	Backup / Re- store	Data group in the backup function is "User program and settings".
	Server certificate			
Event log				Backup only
Operation Logs		Not supported	---	Refer to the Additional Information be- low.
Spool data				The Spool data is cleared by the Re- store operation.



Additional Information

The Operation Logs cannot be backed up nor restored by the Backup/Restore operation. If you want to keep the Operation Log data after replacement of the CPU Unit, insert the used SD Memory Card to the restore-destination CPU Unit after completion of the Restore operation.

The Combination of CPU Units for Which Backup and Restore Function is Available

The backup and restore function is available for the combination of the CPU Units shown below. However, the function will not be available for the following cases:

- The Unit version of the restore-destination CPU Unit is earlier than the Unit version of the backup-source CPU Unit

- The version of the restore-destination DB Connection Service is lower than the version of the backup-source DB Connection Service

If you try to restore data by using the combination of CPU Units where the backup/restore function is not available, the "Restore Operation Failed to Start" event is registered to the event log.

Backup source	Restore destination
NX701-□□20	NX701-□□20
NX102-1□20	NX102-1□20
NX102-9020	NX102-9020
NJ501-1□20	NJ501-1□20
NJ501-4320	NJ501-4320
NJ101-1□20	NJ101-1□20
NJ101-9020	NJ101-9020

5-8-2 Operation Authority Verification in the DB Connection Service

This function is used to restrict the online operations that can be performed on the CPU Unit from Sysmac Studio according to the operation rights.

This section describes the operation authority verification function related to the DB Connection Service.

Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* and the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for details of the operation authority verification function.

The functions, authorities, and operation restrictions that require verification in the DB Connection Service are given below.

OP: Operation possible VR: Verification required for each operation NP: Operation not possible

Monitoring status	Administrator	Designer	Maintainer	Operator	Observer
DB Connection Service Monitor	OP	OP	OP	OP	OP
Connection Monitor Table	OP	OP	OP	OP	OP

Controller operations	Administrator	Designer	Maintainer	Operator	Observer
Displaying the Operation Logs	OP	OP	OP	OP	NP
Clearing the Operation Logs	OP	OP	OP	NP	NP
Starting/stopping the DB Connection Service	OP	OP	NP	NP	NP
Shutting down the DB Connection Service	OP	OP	NP	NP	NP
Starting/stopping the Debug Log	OP	OP	VR	NP	NP
Clearing the Spool data	OP	OP	NP	NP	NP

DB connection test	Administrator	Designer	Maintainer	Operator	Observer
Communications test	OP	OP	OP	NP	NP

5-8-3 Encrypted Communication

The encrypted communication function is designed to prevent sniffing and tampering by encrypting communication data between the controller and the database.

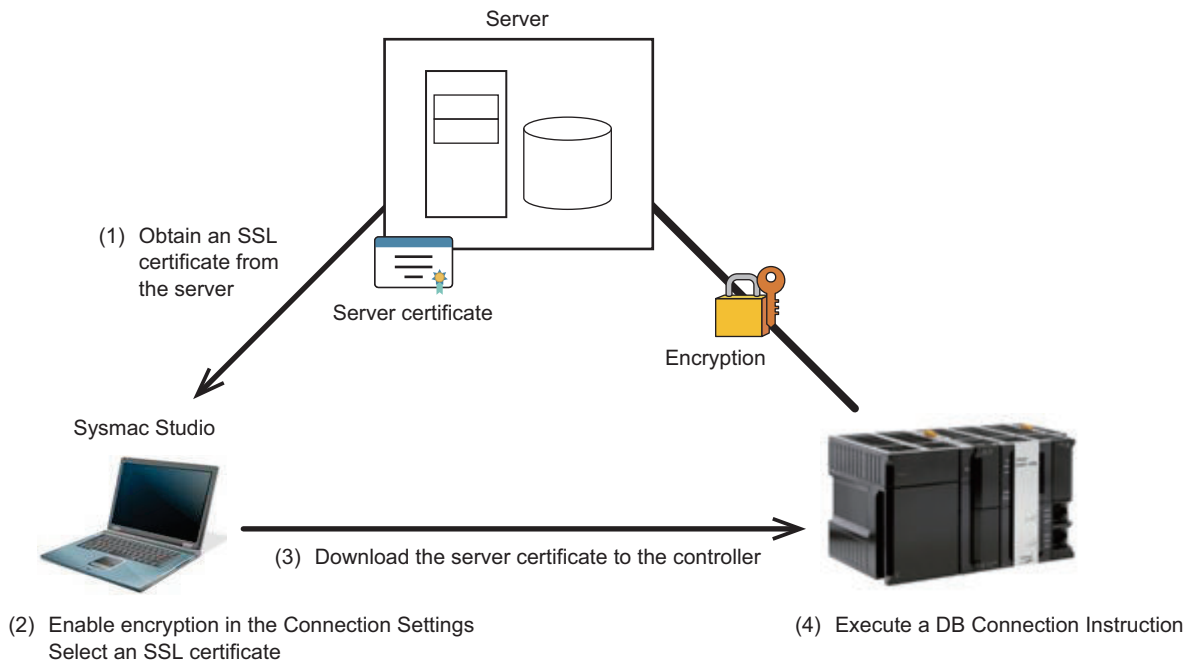
The DB Connection Service uses SSL/TLS for encrypted communication.

Encrypted communication is available for the DB Connection Service version 2.00 or higher with the DB Connection Settings configured on the Sysmac Studio.

Refer to 2-2-2 *DB Connection Settings* on page 2-7 for details on the settings of the encrypted communication.

Mechanism and Specifications of Encrypted Communication

The mechanism of encrypted communication using an SSL certificate is described below.



- (1) Obtain an SSL certificate used by the server and copy it onto a computer where Sysmac Studio is installed.
- (2) In the Connection Settings of Sysmac Studio, enable encryption and select the SSL certificate.
- (3) By using the normal transfer (download) function of Sysmac Studio, download the DB configuration file and the SSL certificate.
- (4) After downloading, run the program and execute the DB Connection Instruction.

[Updating Certificate]

- (5) In the Connection Settings of Sysmac Studio, select the SSL certificate again.
- (6) By using the download function of Sysmac Studio, download the DB configuration and the updated SSL certificate.

[Deleting Certificate]

- (5) In the Connection Settings of Sysmac Studio, disable encryption.
- (6) Download them on Sysmac Studio.

The following table summarizes the specifications of encrypted communication.

Item	Specifications
Supported database type	SQL Server Oracle Database MySQL Community Edition PostgreSQL
Protocol	TLS 1.2

Item		Specifications
Cipher suite		The following cipher suites available in the supported database types can be used: • Mandatory cipher suites for RFC5246 (TLS1.2) • Cipher suites in the highest priority group (most recommended cipher suites)
Communications test		Encrypted communication test is also available
Server certificate	Max. number of certificates (total)	15
	Certificate chain	The following certificates can be used for up to five levels: • Server certificate • Root certificate • Intermediate CA certificate
	Format	X.509-compliant
	Characters allowed for file name	The following characters enclosed with " " are allowed. "-" indicates characters within the range. "0 - 9", "A - Z", "a - z", "%", " ", "-", "_", "@", "!", "(", ")", "~", "#", "&", "^", "{", "}", "=", "[", "]", "+", ";", ":", " ", "\$", " ", "`"
	CA certificate	Supported Specify a certificate in the Connection Settings of Sysmac Studio.
	Backup and restore	Supported



Additional Information

The table below specifies the relationship between the supported database versions and protocols.

For details, refer to the manual of the corresponding database provider.

(S: Supported, N: Not supported, C: Supported with conditions)

DB type	Versions	Protocol (TLS1.2) supported
SQL Server	2012	C (Update required)
	2014	C (Update required)
	2016	S
	2017	S
Oracle Database	11g	N
	12c	S
	18c	S
MySQL Community Edition	5.6	N
	5.7	C (Patching required)
	8.0	S
PostgreSQL	9.4	N
	9.5	S
	9.6	S
	10	S

Operation Flow

The operation flow is described below. This operation flow is same for all the database types.

Refer to 2-2-2 *DB Connection Settings* on page 2-7 for details on the settings of execution processing.

Step	Operation target	Execution processing	Remarks
1	Tool	In the Connection Settings, select Use (initial value: Do not use).	Required only when using encrypted communication
2	Tool	In the Connection Settings, specify a server certificate.	Required only when using encrypted communication
3	Tool	Download the certificate to the controller	
4	Controller	Execute the DB Connection Instruction	Communication with the database is automatically encrypted

Operation When a Server Certification was Set

The following table summarizes the operation when a server certificate is set, if the encrypted communication function is used.

Database types	Setting a server certificate	Operation
Oracle	Required	When SSL/TLS communications are established, if the server certificate that you set does not match the server certificate in the DB server, the DB_Connect instruction fails.
SQL Server	Can be omitted	If a server certificate was set, the server certificate is not used when SSL/TLS communications are established. Therefore, if the server certificate that you set does not match the server certificate in the DB server, the DB_Connect instruction is successful. (The DB connection is possible.)
MySQL		
PostgreSQL		

6

How to Use Operation Logs

This section describes how to use the Operation Logs for tracing the operations of the DB Connection Service.

6-1	Operation Logs.....	6-2
6-2	Execution Log.....	6-3
6-2-1	Overview.....	6-3
6-2-2	Application Procedure.....	6-3
6-2-3	Setting the Execution Log.....	6-3
6-2-4	Checking the Execution Log	6-4
6-2-5	Execution Log File Specifications	6-4
6-3	Debug Log.....	6-15
6-3-1	Overview.....	6-15
6-3-2	Application Procedure.....	6-15
6-3-3	Set the Debug Log.....	6-15
6-3-4	Start Recording to the Debug Log	6-16
6-3-5	Stopping Recording to Debug Log.....	6-17
6-3-6	Checking the Debug Log	6-18
6-3-7	Debug Log File Specifications	6-18
6-4	SQL Execution Failure Log	6-29
6-4-1	Overview.....	6-29
6-4-2	Application Procedure.....	6-29
6-4-3	Setting the SQL Execution Failure Log.....	6-29
6-4-4	Checking the SQL Execution Failure Log.....	6-30
6-4-5	SQL Execution Failure Log File Specifications.....	6-30
6-5	SD Memory Card Operations	6-36
6-5-1	Saving Operation Log Files on SD Memory Card.....	6-36
6-5-2	Directory Used for DB Connection Service.....	6-36
6-5-3	Operation Log Operations in Replacing the SD Memory Card.....	6-37
6-5-4	Guidelines for SD Memory Card Replacement Time.....	6-37
6-5-5	Replacement Timing of SD Memory Card	6-38
6-6	Checking the Operation Logs	6-39
6-6-1	How to Check the Operation Logs	6-39
6-6-2	Checking the Log on the Operation Log Window in Sysmac Studio	6-39
6-6-3	Checking the Log with the SD Memory Card.....	6-41
6-6-4	Checking the Log by Transfer using FTP Client Software	6-41

6-1 Operation Logs

Operation Logs are used to trace the operations of the DB Connection Service on the CPU Unit. The logs are saved on the SD Memory Card mounted in the CPU Unit.

The following three types of Operation Logs are provided.

Operation Log type	Description
Execution Log	Used to record the executions of the DB Connection Service in order to check the execution records of the DB Connection function.
Debug Log	Used to record the contents and results of SQL executions and user-specified logs for debugging.
SQL Execution Failure Log	Used to record the transmitted SQL statements and error information in order to check the information on execution failure of SQL statements in the DB.

6-2 Execution Log

This section describes the "Execution Log" used to trace the executions of the DB Connection Service.

6-2-1 Overview

You can check the start/stop of the DB Connection Service, connection/disconnection with the DB, and success/failure of SQL statement executions with the Execution Log. Thus, you can check whether the expected DB Connection Service processing is executed.

You can record this log by setting **Execution log** to **Record** in the DB Connection Service Settings of Sysmac Studio. You can also record a specified log as Execution Log by executing a DB_PutLog (Record Operation Log) instruction.

When you record this log, the Execution Log file is constantly saved on the SD Memory Card mounted in the CPU Unit while the DB Connection Service is running.

The Execution Log is temporarily recorded in the internal buffer (volatile memory) of the CPU Unit and then saved on the SD Memory Card. While the SD Memory Card is being replaced, the Execution Log is kept in the internal buffer (volatile memory) of the CPU Unit. When you insert an SD Memory Card, the Execution Log temporarily stored in the internal buffer is automatically saved on the SD Memory Card. Refer to 6-5-3 *Operation Log Operations in Replacing the SD Memory Card* on page 6-37 for details.

You can check the contents of this log in the **Execution Log** Tab Page of the Operation Log Window in Sysmac Studio.

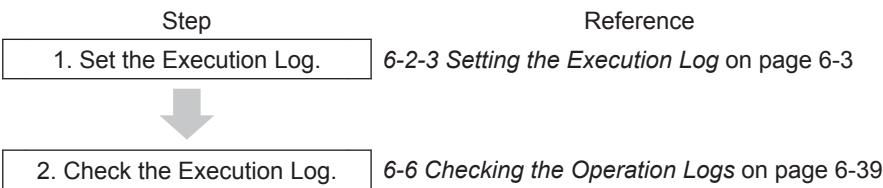


Precautions for Correct Use

When you use the Execution Log, be sure to insert an SD Memory Card into the CPU Unit. The Execution Log is temporarily recorded in the internal buffer of the CPU Unit and then saved on the SD Memory Card. If no SD Memory Card is mounted at power OFF or shutdown processing of the CPU Unit, the Execution Log recorded in the internal buffer will be lost.

6-2-2 Application Procedure

Use the Execution Log according to the following procedure.



6-2-3 Setting the Execution Log

Double-click **DB Connection Service Settings** under **Configurations and Setup - Host Connection Settings - DB Connection** in the Multiview Explorer. Then, set the following in the Service Setting.

Item	Description	Values
Execution Log	Set whether to record the Execution Log.	- Record (Default) - Do not recorded
Number of files	Set the maximum number of files of the Execution Log. When the maximum number of files is reached, the oldest file is deleted and a new file is created.	2 to 100 (Default: 48)
Number of records	Set the number of log records that can be contained in each Execution Log file. When the maximum number of records is reached, a new file is created.	100 to 65536 (Default: 7200)

You can record a specified log as Execution Log using a DB_PutLog (Record Operation Log) instruction. The logs recorded by a DB_PutLog (Record Operation Log) instruction are called "user-specified log".

To record a user-specified log, set Log Type to "Execution Log" and specify the log code, log name, and log message in a DB_PutLog (Record Operation Log) instruction and execute the instruction. Refer to *Section 7 DB Connection Instructions* on page 7-1 for details of the DB_PutLog (Record Operation Log) instruction.

6-2-4 Checking the Execution Log

Refer to *6-6 Checking the Operation Logs* on page 6-39 for how to check the Execution Log.

6-2-5 Execution Log File Specifications

This section describes the specifications of Execution Log files.

- Each Execution Log file is composed of multiple records.
- Each record is expressed in one line.
- The maximum number of records to be contained in each Execution Log file is set in Sysmac Studio.
- The size of a single record is up to 256 bytes for the DB Connection Service version 1.04 and lower. It is up to 58 KB for version 2.00 and higher.
- The following table shows the file name and type.

File name	File type
DB_ExecutionLog.log	Latest log file of the log
DB_ExecutionLog_[year_month_date_hours_minutes_seconds_milliseconds].log* ¹ Example: DB_ExecutionLog_20120724220915040.log	Previous log files
DB_ExecutionLog.fjc	Log control file

*1. The system time of the CPU Unit is used for the time information included in the file name.

- The files are stored in the following directory (of the SD Memory Card).
 - Log files:
/packages/DB_Connection/ExecutionLog/
 - Log control file:
/packages/DB_Connection/System/
- The following is the format of records.
Each record is expressed in one line and composed of multiple parameters. The parameters are separated from each other by a tab.

```
[Serial number]<tab>[Date]<tab>[Time]<tab>[Millisecond]<tab>[Category]<tab>[Log code]<tab>[Log
name]<tab>[Result]<tab>[DB Connection name]<tab>[Serial ID]<tab>[Details]<CR><LF>
```

Parameter	Size	Description
Serial number	1 to 5 bytes	0 to 65535 When exceeding 65535, this value returns to 0. The serial number is given across multiple files. (Even if a new file is created, the serial number is not reset to 0.)
Date	10 bytes (Fixed)	Displays year, month, and date when the log was recorded* ¹ . YYYY-MM-DD Example: 2012-07-23
Time	8 bytes (Fixed)	Displays hours, minutes, and seconds when the log was recorded* ¹ . hh:mm:ss Example: 15:33:45
Millisecond	3 bytes (Fixed)	Displays 3-digit decimal integer (000 to 999) that shows millisecond of the time when the log was recorded* ¹ . Example: 10 ms: 010 623 ms: 623
Category	16 bytes max. (Variable)	Displays the category. Refer to <i>Category</i> on page 6-6 for details.
Log code	4 bytes (Fixed)	Displays a 4-digit decimal code that is a unique identification code in the category. Refer to <i>Log Code</i> on page 6-7 for details.
Log name	32 bytes max. (Variable)	Displays a name that shows the contents of the log. Refer to <i>Log Name</i> on page 6-8 for details.
Result	6 bytes (Fixed)	Displays a 4-digit hexadecimal code that shows the execution result. (e.g., 0x1234) 0x0000: Succeeded Other than 0x0000: Failed (Same code as ErrorID of DB Connection Instruction)
DB Connection name	16 bytes max. (Variable)	Displays a DB Connection name (single-byte alphanumeric characters) *When the category is "DB Connection Service" or "User-specified Log", nothing is displayed.
Serial ID	10 bytes max. (Variable)	Displays the ID code assigned at the execution of record processing and stored procedure instructions. Decimal code consisting of 10 digits max. Possible range: 0 to 2147483647 When this value exceeds 2147483647 or when the power supply to the CPU Unit is turned ON, the value returns to 0. * When the category is "DB Connection Service", "DB Connection", or "User-specified Log", nothing is displayed.

Parameter	Size	Description
Details	Variable	Displays the details of the Execution Log. The contents differ according to the category. In the Details parameter, information items are separated from each other by a tab. Refer to <i>Information Item Details</i> on page 6-8 for details of each information item. [Category string: DB_CONNECTION (Category: DB Connection)] [SQL status]<tab>[DB error code]<tab>[Error message] [Category string: SQL (Category: SQL)] - For INSERT/UPDATE/SELECT/DELETE [Table name]<tab>[DB Map Variable name]<tab>[DB response time]<tab>[DB error code] - For BATCHINSERT Refer to <i>Details for BATCHINSERT</i> on page 6-8. [Category string: PROCEDURE (Category: Stored procedure)] Refer to <i>Details of Stored Procedure</i> on page 6-10. [Category string: SQL_RESEND (Category: SQL resend)] [DB response time]<tab>[DB error code] [Category string: USER (Category: User-specified Log)] [Log Message]
Tab separation	10 bytes in total	
CR+LF	2 bytes	

*1. The date and time information follows the time zone set when the power supply to the Controller is turned ON. After you change the time zone, cycle the power supply.

● Category

Category	Category string
DB Connection Service	DB_SERVICE
DB Connection	DB_CONNECTION
SQL	SQL
Stored procedure	PROCEDURE
SQL Resend	SQL_RESEND
User-specified Log	USER

● Log Code

Category	Code (decimal)	Operation	Log recording timing
DB Connection Service	0001	DB Connection Service Started	When the start processing of the DB Connection Service is completed (succeeded/failed)
	0002	DB Connection Service Stopped	When the stop processing of the DB Connection Service is completed (succeeded/failed)
	0003	DB Connection Service Shutdown	When the shutdown processing of the DB Connection Service is completed (succeeded/failed)
DB Connection	0001	DB Connection Established	When the establishment processing of a DB Connection is completed (succeeded/failed) after the establishment is commanded from Sysmac Studio or the applicable instruction.
	0002	DB Connection Closed	When the close processing of a DB Connection is completed (succeeded/failed) after the close is commanded from Sysmac Studio or the applicable instruction.
	0003	DB Connection Disconnected	When disconnection from the DB is detected.
	0004	DB Connection Reestablished	When the DB Connection status changes from Disconnected to Connected.
SQL	0001	INSERT	When a response (succeeded/failed) is returned to INSERT that is issued from DB Connection Service to DB after execution of a DB_Insert (Insert DB Record) instruction.
	0002	UPDATE	When a response (succeeded/failed) is returned to UPDATE that is issued from DB Connection Service to DB after execution of a DB_Update (Update DB Record) instruction.
	0003	SELECT	When a response (succeeded/failed) is returned to SELECT that is issued from DB Connection Service to DB after execution of a DB_Select (Retrieve DB Record) instruction.
	0004	DELETE	When a response (succeeded/failed) is returned to DELETE that is issued from DB Connection Service to DB after execution of a DB_Delete (Delete DB Record) instruction.
	0005	BATCHINSERT	When a response (succeeded/failed) is returned to BATCHINSERT that is issued from DB Connection Service to DB after execution of a DB_BatchInsert instruction.
PROCEDURE	0001	ATTACH	When a DB_AttachProcedure instruction is executed.
	0002	EXECUTE	When a response (succeeded/failed) is returned to PROCEDURE that is issued from DB Connection Service to DB after execution of a DB_ExecuteProcedure instruction.
	0003	DETACH	When a DB_DetachProcedure instruction is executed.
SQL Re-send	0001	INSERT	When a response (succeeded/failed) is returned to INSERT after resending the INSERT statement stored in the Spool memory.
	0002	UPDATE	When a response (succeeded/failed) is returned to UPDATE after resending the UPDATE statement stored in the Spool memory.
User-specified Log	0000 to 9999 (specified by the user)	DB_PutLog Instruction Executed	When a DB_PutLog (Record Operation Log) instruction is executed

● Log Name

Category	Operation	Log name
DB Connection Service	DB Connection Service Started	Start
	DB Connection Service Stopped	Stop
	Shutdown DB Connection Service	Shutdown
DB Connection	DB Connection Established	Connect
	DB Connection Closed	Close
	DB Connection Disconnected	Disconnect
	DB Connection Reestablished	Reconnect
SQL	INSERT	INSERT
	UPDATE	UPDATE
	SELECT	SELECT
	DELETE	DELETE
	BATCHINSERT	BATCHINSERT
PROCEDURE	EXECUTE	EXECUTE
	DETACH	DETACH
	ATTACH	ATTACH
SQL Resend	INSERT	INSERT
	UPDATE	UPDATE
User-specified Log	DB_PutLog Instruction Executed	Text string specified in the <i>LogName</i> input variable of the DB_PutLog instruction.

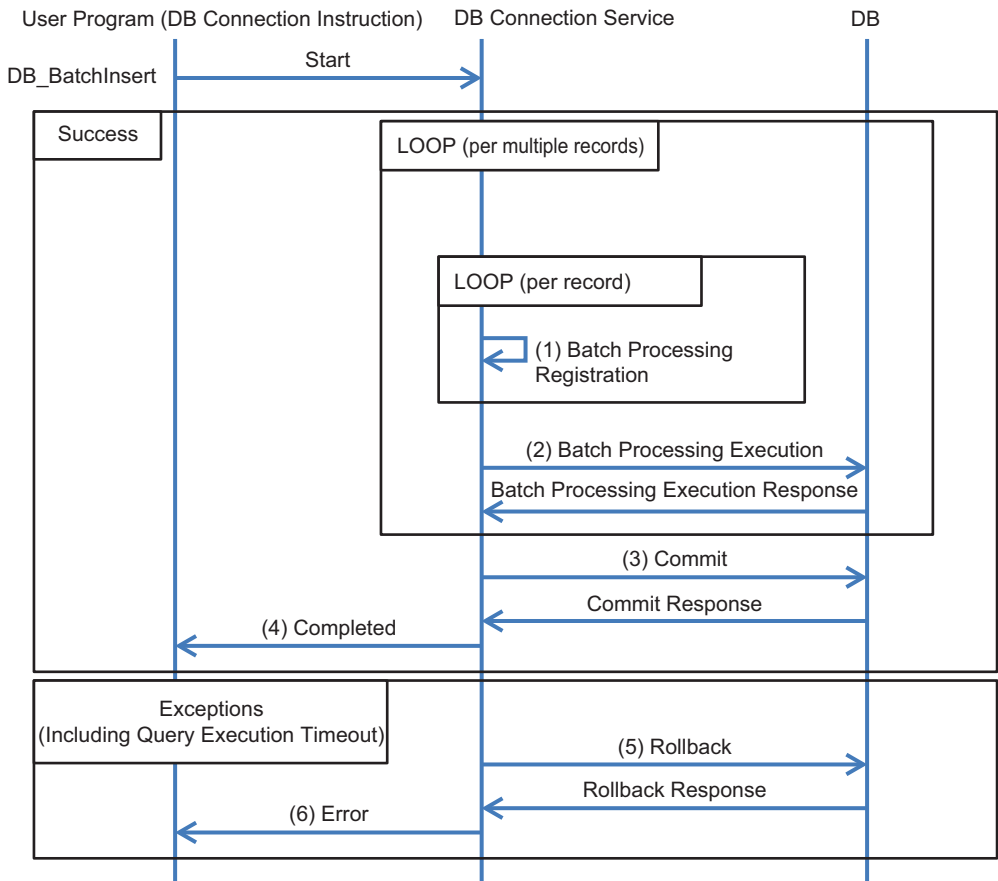
● Information Item Details

Information	Description
SQL status	The SQLSTATE value defined in the SQL Standards (ISO/IEC 9075) is displayed.
DB error code	Error code that is specific to DB vendor of the device to connect. When a network error has occurred, 0 is displayed for DB error code in some cases. When 0 is displayed, check its SQL status.
Error message	The error message is displayed from the first character within the record size (i.e., 256 bytes).
Table name, DB Map Variable name	A maximum of 60 bytes from the beginning are displayed.
DB Map Variable name	Variable name specified in the <i>MapVar</i> input variable (The POU instance name is not displayed. Nothing is displayed for DELETE.)
DB response time	An integer value in milliseconds is displayed.
DB log message	Displays the text string specified in the <i>LogMsg</i> input variable of the DB_PutLog instruction. (128 bytes max.)

Details for BATCHINSERT

The details for BATCHINSERT are described below.

The format of details for BATCHINSERT varies by the timing (1) through (6) shown in the figure below.



The format of details on the timing (1) through (6) is specified below.

Output timing		Category	Log name	Details*1
(2)	Batch Processing Execution	SQL	BATCHINSERT	[Table name]<tab>[DB Map Variable name]<tab>[Batch Insert processing name:REQUEST]<tab>[Insert count]<tab>[Processing array range]
(3)	Commit			[Table name]<tab>[DB Map Variable name]<tab>[Batch Insert processing name:COMMIT]
(4)	Done			[Table name]<tab>[DB Map Variable name]<tab>[DB response time]<tab>
(5)	Rollback			[Table name]<tab>[DB Map Variable name]<tab>[Batch insert processing name:ROLLBACK]
(6)	Error			[Table name]<tab>[DB Map Variable name]<tab>[DB response time]<tab>[DB error code]

*1. Refer to *Information Item Details (for BATCHINSERT)* on page 6-9 for details of each item.

● Information Item Details (for BATCHINSERT)

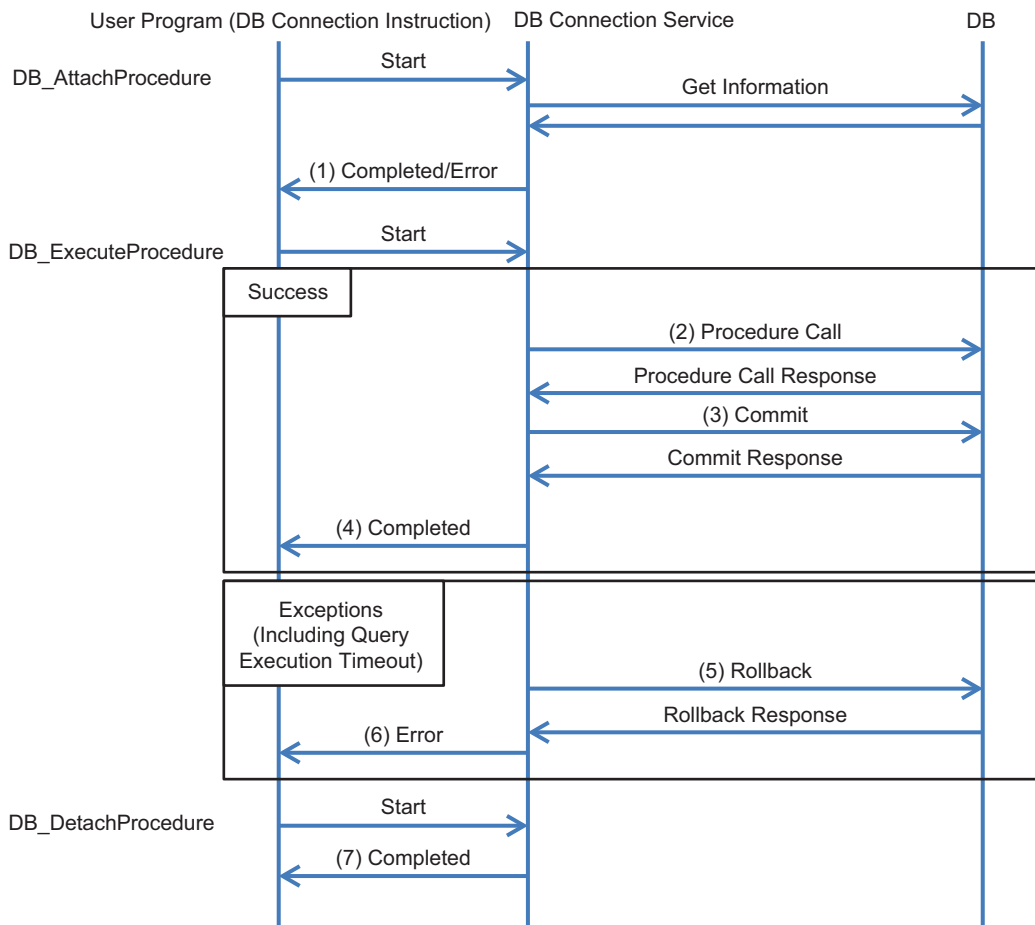
Item	Description
Table name, DB Map Variable name	A maximum of 60 bytes from the beginning are displayed.

Item	Description
Batch Insert processing name	It shows the internal processing of BATCHINSERT. The processing varies by the timing (1), (2), (3), (5) shown in the above figure. (1): APPEND: Batch Processing Registration (2): REQUEST: Batch Processing Execution (3): COMMIT: Commit (5): ROLLBACK: Rollback
Insert count	The value of the <i>InsertCnt</i> input variable is displayed. If the number of array elements in the DB Map Variable is equal to or less than InsertCnt, or if InsertCnt is equal to 0, the number of array elements in the DB Map Variable is displayed. (1 to 5 bytes)
Processing array range	It displays the difference between the start and end numbers of the array elements that were requested to be inserted in the database. Example: For the array elements [0..299], it shows as 0..99, 100..199, 200..299.
DB response time	An integer value in milliseconds is displayed.
DB error code	Error code that is specific to DB vendor of the device to connect. When a network error has occurred, 0 is displayed for DB error code in some cases. When 0 is displayed, check its SQL status.

Details of Stored Procedure

The details for the stored procedure are described below.

The format of details for the stored procedure varies by the timing (1) through (7) shown in the figure below.



The format of details on the timing (1) through (7) is specified below.

Output timing		Category	Log name	Details*1
(1)	Done/ Error	PROCEDURE	ATTACH	Normally completed: [Procedure name]<tab>[Procedure handle]<tab>[Attach error code]<tab>[Procedure IF]<tab>[IN argument map variable name]<tab>[OUT argument map variable name]<tab>[INOUT argument map variable name]<tab>[Return value map variable name]<tab>[Result set map variable name] Ended with error: [Procedure name]<tab><tab>[Attach error code]<tab>[Procedure IF]<tab>[IN argument map variable name]<tab>[OUT argument map variable name]<tab>[INOUT argument map variable name]<tab>[Return value map variable name]<tab>[Result set map variable name]<tab>[Attach error location]<tab>[Attach error member name] Normal completed and ended with error when the the internal buffer of the CPU Unit is used. [Procedure name]<tab>[Procedure handle]<tab>[Attach error code]<tab>...
(2)	Procedure Call	PROCEDURE	EXECUTE	[Procedure name]<tab>[Procedure handle]<tab>[Procedure process name: CALL]
(3)	Commit			[Procedure name]<tab>[Procedure handle]<tab>[Procedure process name: COMMIT]
(4)	Done			[Procedure name]<tab>[Procedure handle]<tab>[Return value]<tab>[DB response time]<tab>
(5)	Rollback			[Procedure name]<tab>[Procedure handle]<tab>[Procedure process name: ROLLBACK]
(6)	Error			[Procedure name]<tab>[Procedure handle]<tab>[DB response time]<tab>[DB error code]
(7)	Done	PROCEDURE	DETACH	[Procedure name]<tab>[Procedure handle]

*1. Refer to *Information Item Details (Stored Procedure)* on page 6-11 for details of each item.

● Information Item Details (Stored Procedure)

Item	Description
Procedure Name	A maximum of 60 bytes from the beginning are displayed. When multi-byte characters are used, the characters are displayed up to the delimiter.
Procedure Handle	Displays the value being output to the <i>ProcHandle</i> output variable for the DB_Attach-Procedure instruction.
Attach error code	Refer to <i>Details of [Attach error code], [Attach error location], and [Attach error member name]</i> on page 6-12.
Procedure IF	Displays the procedure IF retrieved from the database. It is output in the following format: "return data type_procedure name (argument direction_argument 1 data type_argument 1 name,_argument 2 data type_argument 2 name)" *_ is a single-byte space. For the argument direction, IN/OUT/INOUT is output. All the data types output are those for the databases.
IN argument map variable name	Displays the variable name specified for the <i>ArgIn</i> input variable.
OUT argument map variable name	Displays the variable name specified for the <i>ArgOut</i> input variable.

Item	Description
INOUT argument map variable name	Displays the variable name specified for the <i>ArgInOut</i> input variable.
Return value map variable name	Displays the variable name specified for the <i>ReturnVal</i> input variable.
Result set map variable name	Displays the variable name specified for the <i>ResultSet</i> input variable.
Attach error location	Refer to <i>Details of [Attach error code], [Attach error location], and [Attach error member name]</i> on page 6-12.
Attach error member name	Refer to <i>Details of [Attach error code], [Attach error location], and [Attach error member name]</i> on page 6-12.
Procedure process name	Displays the internal process of the stored procedure. The processing varies by the timing (2), (3), (5) shown in the above figure. (2): CALL: Procedure Call (3): COMMIT: Commit (5): ROLLBACK: Rollback
Return value	Displays the return value.
DB response time	An integer value in milliseconds is displayed.
DB error code	Error code that is specific to DB vendor of the device to connect. When a network error has occurred, 0 is displayed for DB error code in some cases. When 0 is displayed, check its SQL status.

● **Details of [Attach error code], [Attach error location], and [Attach error member name]**

Attach error code	Attach error location	Attach error member name	Description
Success	---	---	Succeeded
ExistOnlyOneSide	ArgIn	---	Although the stored procedure contains an IN argument, <i>_DBC_Unused</i> is set to ArgIn of the controller.
	ArgOut	---	Although the stored procedure contains an OUT argument, <i>_DBC_Unused</i> is set to ArgOut of the controller.
	ArgInOut	---	Although the stored procedure contains an INOUT argument, <i>_DBC_Unused</i> is set to ArgInOut of the controller.
	ReturnVal	---	Although the stored procedure does not contain a return value, a value other than <i>_DBC_Unused</i> is set to ReturnVal of the controller.
TypeNotMatch	ArgIn	Member name	The IN argument type of the stored procedure does not match the ArgIn member variable type on the controller.
	ArgOut	Member name	The OUT argument type of the stored procedure does not match the ArgOut member variable type on the controller.
	ArgInOut	Member name	The INOUT argument type of the stored procedure does not match the ArgInOut member variable type on the controller.
	ReturnVal	---	The return value type of the stored procedure does not match the ReturnVal type on the controller.

Attach error code	Attach error location	Attach error member name	Description
CountNotMatch	ArgIn	---	The number of IN arguments of the stored procedure does not match the number of ArgIn member variables on the controller.
	ArgOut	---	The number of OUT arguments of the stored procedure does not match the number of ArgOut member variables on the controller.
	ArgInOut	---	The number of INOUT arguments of the stored procedure does not match the number of ArgInOut member variables on the controller.
NameNotMatch	ArgIn	Member name	The ArgIn member variables on the controller contain a name that does not exist in the IN arguments of the stored procedure.
	ArgOut	Member name	The ArgOut member variables on the controller contain a name that does not exist in the OUT arguments of the stored procedure.
	ArgInOut	Member name	The ArgInOut member variables on the controller contain a name that does not exist in the INOUT arguments of the stored procedure.
DBMSSpecific	---	---	The OUT arguments of the stored procedure contain two or more cursor-type data. (Oracle and PostgreSQL only)

Record Examples

- DB Connection Service Started:

```
1 2012-07-24 21:29:45 267 DB_SERVICE 0001 Start 0x0000
```

- INSERT (Failed):

```
1 2012-07-24 21:29:45 267 SQL 0001 INSERT 0x1234 DBConnection1 45 TableX VarY 10
0 17026
```

- User-specified Log:

```
1 2012-07-24 21:29:45 267 USER 9876 LineA1 0x0000
"ProductionStarted"
```

Log File Example

```
0 2012-07-24 08:29:45 267 DB_SERVICE 0001 Start 0x0000
1 2012-07-24 08:31:52 002 DB_CONNECTION 0001 Connect 0x0000 MyDatabase1
2 2012-07-24 08:31:53 959 DB_CONNECTION 0001 Connect 0x0000 MyDatabase2
3 2012-07-24 09:00:00 052 USER 0001 LineA1 0x0000 "ProductionStarted"
4 2012-07-24 09:00:00 150 SQL 0001 INSERT 0x0000 MyDatabase1 0 TABLE_Production
Production 100 0
5 2012-07-24 09:10:00 150 SQL 0001 INSERT 0x0000 MyDatabase1 1 TABLE_Production
Production 100 0
6 2012-07-24 09:20:00 151 SQL 0001 INSERT 0x0000 MyDatabase1 2 TABLE_Production
Production 100 0
7 2012-07-24 09:30:00 150 SQL 0001 INSERT 0x0000 MyDatabase1 3 TABLE_Production
Production 100 0
8 2012-07-24 09:55:23 422 USER 0002 LineA1 0x0000 "ProductionFinished"
9 2012-07-24 10:15:00 549 SQL 0003 SELECT 0x0000 MyDatabase2 4 TABLE_MPS Production
nSchedule 200 0
```



Precautions for Correct Use

Do not delete the latest log file (DB_ExecutionLog.log) and the log control file (DB_Execution-Log.fjc) from the SD Memory Card. If they are deleted, the log files are not saved correctly, for example, the Execution Log data are lost.

6-3 Debug Log

This section describes the "Debug Log" used for debugging the DB Connection Service.

6-3-1 Overview

You can check which SQL statement is executed, parameters of each SQL statement, and execution results with the Debug Log.

You can record this log by clicking the **Start** Button for **Debug Log** in the **Online Settings** Tab Page of Sysmac Studio. You can also record a specified log as Debug Log by executing a DB_PutLog (Record Operation Log) instruction.

This log is saved as Debug Log files on the SD Memory Card mounted in the CPU Unit. When no SD Memory Card is mounted in the CPU Unit, you cannot record the Debug Log.

You can check the contents of this log in the **Debug Log** Tab Page of the Operation Log Window in Sysmac Studio.

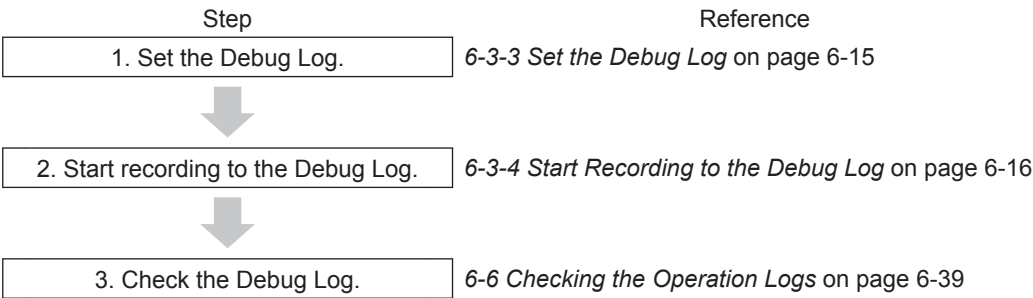


Additional Information

The Debug Log is used to check the parameters and execution results of the SQL statements executed using the DB Connection Instructions. When the Spool data is resent, it is not recorded to the Debug Log. To check the time and execution results of SQL statements resent from the Spool memory, check the Execution Log record with the same serial ID. To check the parameters of the SQL statements in that case, check the log record at the time when the applicable SQL statement is spooled in the Debug Log.

6-3-2 Application Procedure

Use the Debug Log according to the following procedure.



6-3-3 Set the Debug Log

Double-click **DB Connection Service Settings** under **Configurations and Setup - Host Connection Settings - DB Connection** in the Multiview Explorer. Then, set the following in the Service Setting.

Item	Description	Values
Number of files	Set the maximum number of files of the Debug Log.	1 to 100 files (Default: 1)

Item	Description	Values
File size	Set the maximum file size. When the maximum file size is exceeded or when the number of records exceeds 65,536 records in a file, a new file is created.	1 to 100 MB (Default: 10 MB)
When the log is full	Set the action to be taken when the Debug Log has reached the maximum number of files.	- Stop logging (Default) - Continue logging (Delete the oldest file)
Delete the log at recording start	Set whether to delete the Debug Log contained in the SD Memory Card when recording is started.	- Delete (Default) - Do not delete

You can record a specified log as Debug Log using a DB_PutLog (Record Operation Log) instruction. The logs recorded by a DB_PutLog (Record Operation Log) instruction are called "user-specified log". To record the user-specified log, set Log Type to "Debug Log" and specify the log code, log name, and log message in a DB_PutLog (Record Operation Log) instruction and execute the instruction. Refer to *Section 7 DB Connection Instructions* on page 7-1 for details of the DB_PutLog (Record Operation Log) instruction.

6-3-4 Start Recording to the Debug Log

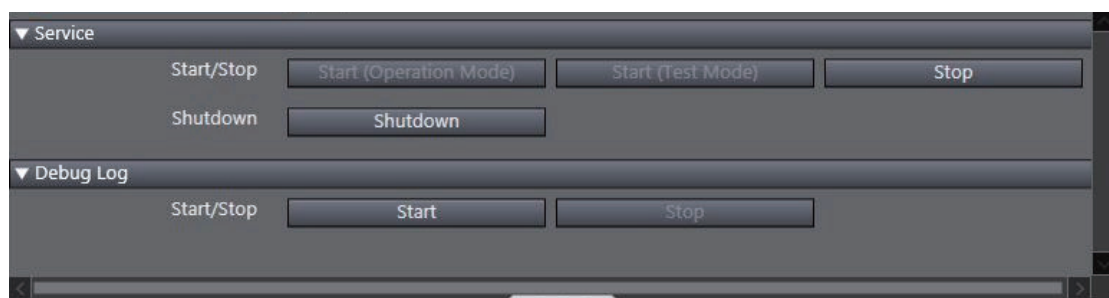
You can start recording to the Debug Log by the following methods.

- Online operation from Sysmac Studio
- Executing a DB_ControlService (Control DB Connection Service) instruction.

Start by Online Operation from Sysmac Studio

- 1 Right-click **DB Connection Service Settings** under **Configurations and Setup - Host Connection Settings - DB Connection** in the Multiview Explorer and select **Online Settings** from the menu.

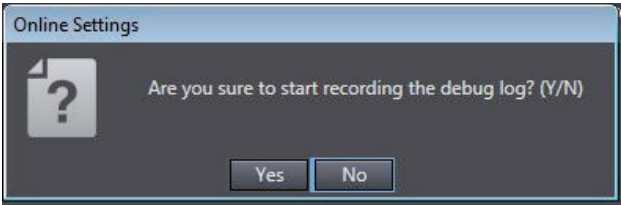
The following Online Settings Tab Page is displayed.



You can start and stop recording to the Debug Log by clicking the following buttons.

Category	Item	Button	Operation
Debug Log	Start/Stop	Start	Recording to the Debug Log is started.
		Stop	Recording to the Debug Log is stopped.

- 2 Click the **Start** Button.
A confirmation message is displayed.



3 Click the **Yes** Button.

Start by Executing a DB_ControlService Instruction

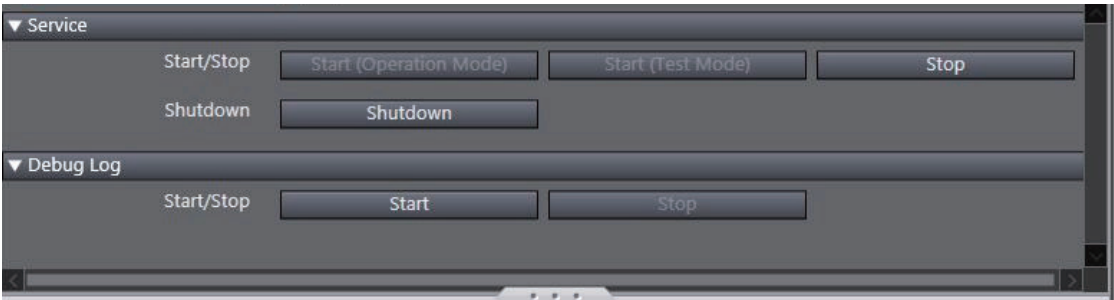
Specify Start recording to Debug Log in the Cmd input variable of the DB_ControlService (Control DB Connection Service) instruction and execute the instruction. Refer to *Section 7 DB Connection Instructions* on page 7-1 for details of the instruction.

6-3-5 Stopping Recording to Debug Log

- You can stop recording to the Debug Log by the following methods.
- Online operation from Sysmac Studio
 - Executing a DB_ControlService (Control DB Connection Service) instruction.
 - Automatically stopped when a specified condition is met

Stop by Online Operation from Sysmac Studio

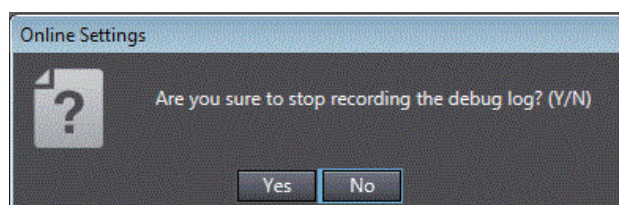
1 Right-click **DB Connection Service Settings** under **Configurations and Setup - Host Connection Settings - DB Connection** in the Multiview Explorer and select **Online Settings** from the menu.
The following Online Settings Tab Page is displayed.



You can start and stop recording to the Debug Log by clicking the following buttons.

Category	Item	Button	Operation
Debug Log	Start/Stop	Start	Recording to the Debug Log is started.
		Stop	Recording to the Debug Log is stopped.

2 Click the **Stop** Button.
A confirmation message is displayed.



- 3** Click the **Yes** Button.

Stop by Executing a DB_ControlService Instruction

Specify Finish recording to Debug Log in the Cmd input variable of the DB_ControlService (Control DB Connection Service) instruction and execute the instruction. Refer to *Section 7 DB Connection Instructions* on page 7-1 for details of the instruction.

Automatically Stopped when a Condition is Met

The recording to Debug Log is automatically stopped in the following conditions.

- When the SD Memory Card power supply switch is pressed
- When the Synchronization (download) operation is executed on Sysmac Studio
- When the Clear All Memory operation is executed
- When the Restore operation of the SD Memory Card backup function or Sysmac Studio Controller backup function is executed

6-3-6 Checking the Debug Log

Refer to 6-6 *Checking the Operation Logs* on page 6-39 for how to check the Debug Log.

6-3-7 Debug Log File Specifications

This section describes the specifications of Debug Log files.

- Each Debug Log file is composed of multiple records.
- The maximum size of each Debug Log file is set in Sysmac Studio.
- The size of a single record is up to 58 KB.
- The following table shows the file name and type.

File name	File type
DB_DebugLog.log	Latest log file of the log
DB_DebugLog_[year_month_date_hours_minutes_seconds_milliseconds].log ^{*1} Example: DB_DebugLog_20120724220915040.log	Previous log files
DB_DebugLog.fjc	Log control file

^{*1.} The system time of the CPU Unit is used for the time information included in the file name.

- The files are stored in the following directory (of the SD Memory Card).
 - a) Log files:
/packages/DB_Connection/DebugLog/
 - b) Log control file:
/packages/DB_Connection/System/

- The record format is shown below.

Each record is expressed in one line and composed of multiple parameters. The parameters are separated from each other by a tab.

```
[Serial number]<tab>[Date]<tab>[Time]<tab>[Millisecond]<tab>[Category]<tab>[Log code]<tab>[Log name]<tab>[Result]<tab>[DB Connection name]<tab>[Serial ID]<tab>[Details]<CR><LF>
```

Parameter	Size	Description
Serial number	1 to 5 bytes	0 to 65535 When exceeding 65535, this value returns to 0. The serial number is given across multiple files. (Even if a new file is created, the serial number is not reset to 0.)
Date	10 bytes (Fixed)	Displays year, month, and date when the log was recorded.* ¹ YYYY-MM-DD Example: 2012-07-23
Time	8 bytes (Fixed)	Displays hours, minutes, and seconds when the log was recorded.* ¹ hh:mm:ss Example: 15:33:45
Millisecond	3 bytes (Fixed)	Displays 3-digit decimal integer (000 to 999) that shows millisecond of the time when the log was recorded.* ¹ Example: 10 ms: 010 623 ms: 623
Category	16 bytes max. (Variable)	Displays the category. Refer to <i>Category</i> on page 6-20 for details.
Log code	4 bytes (Fixed)	Displays a 4-digit decimal code that is a unique identification code in the category. Refer to <i>Log Code</i> on page 6-20 for details.
Log name	32 bytes max. (Variable)	Displays a name that shows the contents of the log. Refer to <i>Log Name</i> on page 6-21 for details.
Result	6 bytes (Fixed)	Displays a 4-digit hexadecimal code that shows the execution result. (e.g., 0x1234) 0x0000: Succeeded Other than 0x0000: Failed (Same code as ErrorID of DB Connection Instruction)
DB Connection name	16 bytes max. (Variable)	Displays a DB Connection name (single-byte alphanumeric characters) * When the category is DB Connection Service or User-specified Log, nothing is displayed.
Serial ID	10 bytes max. (Variable)	Displays the ID code assigned at the execution of record processing and stored procedure instructions. (Displays the same ID as the serial ID displayed for the "SQL" category records in the Execution Log) Decimal code consisting of 10 digits max. Possible range: 0 to 2147483647 When this value exceeds 2147483647 or when the power supply to the CPU Unit is turned ON, the value returns to 0. * When the category is "DB Connection Service", "DB Connection", or "User-specified Log", nothing is displayed.

Parameter	Size	Description
Details	Variable	Displays the details of the Debug Log. The contents differ according to the category. In the Details parameter, information items are separated from each other by a tab. Refer to <i>Information Item Details</i> on page 6-22 for details of each information item. [Category string: SQL (Category: SQL)] - For INSERT/UPDATE/SELECT/DELETE [Table name]<tab>[DB Map Variable name]<tab>[SQL statement] - For BATCHINSERT Refer to <i>Details for BATCHINSERT</i> on page 6-22. [Category string: PROCEDURE (Category: PROCEDURE)] Refer to <i>Details of Stored Procedure</i> on page 6-24. [Category string: SQL_RESULT (Category: SQL execution result)] - For INSERT/UPDATE/SELECT/DELETE/BATCHINSERT [Table name]<tab>[DB Map Variable name]<tab>[DB response time]<tab>[DB error code]<tab>[Error message] [Category string: PROCEDURE_RESULT (Category: PROCEDURE execution result)] - For EXECUTE [Procedure name]<tab>[Procedure handle]<tab>[Procedure return value]<tab>[DB response time]<tab>[DB error code]<tab>[Error message] [Category string: USER (Category: User-specified Log)] [Log Message]
Tab separation	10 bytes in total	
CR+LF	2 bytes	

*1. The date and time information follows the time zone set when the power supply to the Controller is turned ON. After you change the time zone, cycle the power supply.

● Category

Category	Category string
DB Connection	DB_CONNECTION
SQL	SQL
Stored Procedure	PROCEDURE
SQL Execution Result	SQL_RESULT
Stored procedure execution result	PROCEDURE_RESULT
User-specified Log	USER

● Log Code

Category	Code (decimal)	Operation	Log recording timing
DB Connection	0001	DB Connection Established	When the establishment processing of a DB Connection is completed (succeeded/failed) after the establishment is commanded from the applicable instruction.

Category	Code (decimal)	Operation	Log recording timing
SQL	0001	INSERT	- Before the DB Connection Service sends an SQL statement after a DB_Insert (Insert DB Record) instruction is executed - When an SQL statement is stored in the Spool memory
	0002	UPDATE	- Before the DB Connection Service sends an SQL statement after a DB_Update (Update DB Record) instruction is executed - When an SQL statement is stored in the Spool memory
	0003	SELECT	Before the DB Connection Service sends an SQL statement after a DB_Select (Retrieve DB Record) instruction is executed.
	0004	DELETE	Before the DB Connection Service sends an SQL statement after a DB_Delete (Delete DB Record) instruction is executed.
	0005	BATCHINSERT	Before the DB Connection Service sends an SQL statement after the DB_BatchInsert instruction is executed.
PROCEDURE	0001	ATTACH	Before executing the DB_AttachProcedure instruction
	0002	EXECUTE	Before the DB Connection Service sends an SQL statement after the DB_ExecuteProcedure is executed.
	0003	DETACH	Before executing the DB_DetachProcedure instruction
SQL Execution Result	0001	INSERT	When a response (succeeded/failed) is returned to the INSERT issued from DB Connection Service to DB.
	0002	UPDATE	When a response (succeeded/failed) is returned to the UPDATE issued from DB Connection Service to DB.
	0003	SELECT	When a response (succeeded/failed) is returned to the SELECT issued from DB Connection Service to DB.
	0004	DELETE	When a response (succeeded/failed) is returned to the DELETE issued from DB Connection Service to DB.
	0005	BATCHINSERT	When a response (succeeded/failed) is returned to the BATCHINSERT issued from DB Connection Service to DB.
PROCEDURE Execution Result	0001	EXECUTE	When a response (succeeded/failed) is returned to the PROCEDURE issued from DB Connection Service to DB.
User-specified Log	0000 to 9999 (specified by the user)	DB_PutLog Instruction Executed	When a DB_PutLog (Record Operation Log) instruction is executed

● Log Name

Category	Operation	Log name
DB Connection	DB Connection Established	Connect
SQL	INSERT	INSERT
	UPDATE	UPDATE
	SELECT	SELECT
	DELETE	DELETE
	BATCHINSERT	BATCHINSERT
PROCEDURE	ATTACH	ATTACH
	EXECUTE	EXECUTE
	DETACH	DETACH

Category	Operation	Log name
SQL Execution Result	INSERT	INSERT
	UPDATE	UPDATE
	SELECT	SELECT
	DELETE	DELETE
	BATCHINSERT	BATCHINSERT
PROCEDURE Execution Result	EXECUTE	EXECUTE
User-specified Log	DB_PutLog Instruction Executed	Text string specified in the <i>LogName</i> input variable of the DB_PutLog instruction.

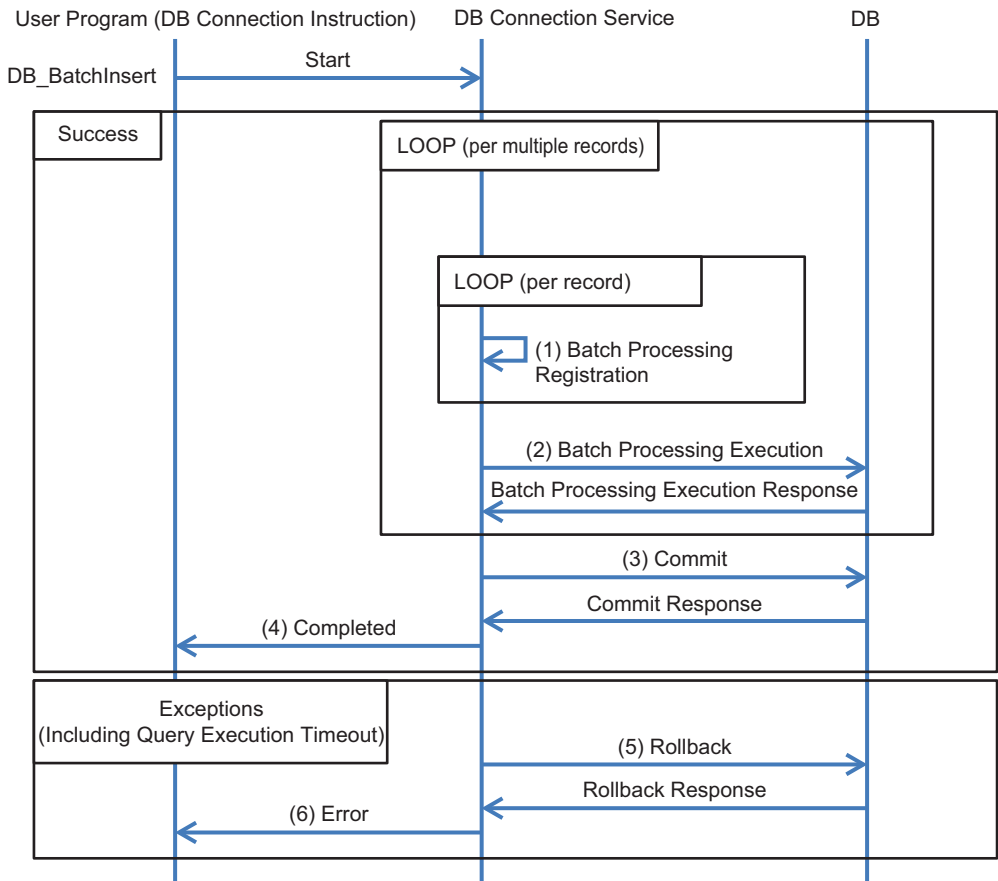
● Information Item Details

Information	Description
Table name, DB Map Variable name	Displays Table name and DB Map Variable name.
DB Map Variable name	Variable name specified in the <i>MapVar</i> input variable (The POU instance name is not displayed. Nothing is displayed for DELETE.)
SQL statement	Displays the SQL statement.
DB response time	An integer value in milliseconds is displayed.
DB error code	Error code that is specific to DB vendor of the device to connect. When a network error has occurred, 0 is displayed for DB error code in some cases. When 0 is displayed, check its SQL status.
Error message	Displays an error message.
DB log message	Displays the text string specified in the <i>LogMsg</i> input variable of the DB_PutLog instruction. (128 bytes max.)

Details for BATCHINSERT

The details for BATCHINSERT are described below.

The format of details for BATCHINSERT varies by the timing (1) through (6) shown in the figure below.



The format of details on the timing (1) through (6) is specified below.

Output timing		Category	Log name	Details ^{*1}
(1)	Batch Processing Registration	SQL	BATCHINSERT	[Table name]<tab>[DB Map Variable name]<tab>[Batch Insert processing name:APPEND]<tab>[Element number]<tab>[SQL statement]
(2)	Batch Processing Execution			[Table name]<tab>[DB Map Variable name]<tab>[Batch Insert processing name:REQUEST]<tab>[Insert count]<tab>[Processing array range]
(3)	Commit			[Table name]<tab>[DB Map Variable name]<tab>[Batch Insert processing name:COMMIT]
(4)	Done	SQL_RESULT	BATCHINSERT	[Table name]<tab>[DB Map Variable name]<tab>[DB response time]<tab><tab>
(5)	Rollback	SQL	BATCHINSERT	[Table name]<tab>[DB Map Variable name]<tab>[Batch insert processing name:ROLLBACK]
(6)	Error	SQL_RESULT	BATCHINSERT	[Table name]<tab>[DB Map Variable name]<tab>[DB response time]<tab>[DB error code]<tab>[Error message]

^{*1}. Refer to *Information Item Details (for BATCHINSERT)* on page 6-23 for details of each item.

● Information Item Details (for BATCHINSERT)

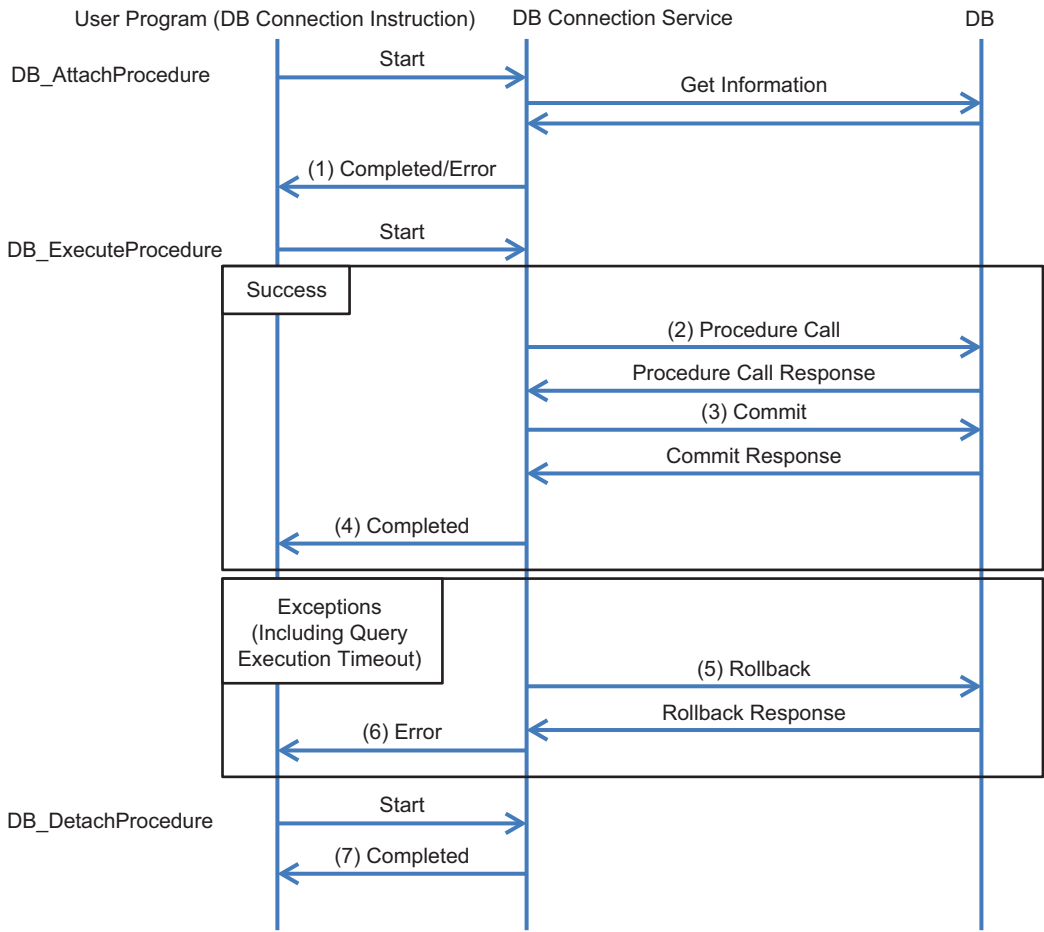
Item	Description
Table name, DB Map Variable name	Displays Table name and DB Map Variable name.

Item	Description
Batch Insert processing name	It shows the internal processing of BATCHINSERT. The processing varies by the timing (1), (2), (3), (5) shown in the above figure. (1): APPEND: Batch Processing Registration (2): REQUEST: Batch Processing Execution (3): COMMIT: Commit (5): ROLLBACK: Rollback
Element number	Displays the element number of the array being processed.
SQL statement	Outputs the SQL statement.
Insert count	The value of the <i>InsertCnt</i> input variable is displayed. If the number of array elements in the DB Map Variable is equal to or less than <i>InsertCnt</i> , or if <i>InsertCnt</i> is equal to 0, the number of array elements in the DB Map Variable is displayed. (1 to 5 bytes)
Processing array range	It displays the difference between the start and end numbers of the array elements that were requested to be inserted in the database. Example: For the array elements [0..299], it shows as 0..99, 100..199, 200..299.
DB response time	An integer value in milliseconds is displayed.
DB error code	Error code that is specific to DB vendor of the device to connect. When a network error has occurred, 0 is displayed for DB error code in some cases. When 0 is displayed, check its SQL status.
Error message	Displays an error message.

Details of Stored Procedure

The details of the stored procedure are described below.

The format of the stored procedure details varies by the timing (1) through (7) shown in the figure below.



The format of details on the timing (1) through (7) is specified below.

Output timing	Category	Log name	Details*1
(1) Done/Error	PROCEDURE	ATTACH	Normally completed: [Procedure name]<tab>[Procedure handle]<tab>[Attach error code]<tab>[Procedure IF]<tab>[IN argument map variable name]<tab>[OUT argument map variable name]<tab>[INOUT argument map variable name]<tab>[Return value map variable name]<tab>[Result set map variable name] Ended with error: [Procedure name]<tab><tab>[Attach error code]<tab>[Procedure IF]<tab>[IN argument map variable name]<tab>[OUT argument map variable name]<tab>[INOUT argument map variable name]<tab>[Return value map variable name]<tab>[Result set map variable name]<tab>[Attach error location]<tab>[Attach error member name]
(2) Procedure Call	PROCEDURE	EXECUTE	[Procedure name]<tab>[Procedure handle]<tab>[Procedure process name: CALL]<tab>[Procedure call statement]
(3) Commit			[Procedure name]<tab>[Procedure handle]<tab>[Procedure process name: COMMIT]
(4) Done	PROCEDURE_RESULT	EXECUTE	[Procedure name]<tab>[Procedure handle]<tab>[Return value]<tab>[DB response time]<tab><tab>
(5) Rollback	PROCEDURE	EXECUTE	[Procedure name]<tab>[Procedure handle]<tab>[Procedure process name: ROLLBACK]
(6) Error	PROCEDURE_RESULT	EXECUTE	[Procedure name]<tab>[Procedure handle]<tab>[DB response time]<tab>[DB error code]<tab>[Error message]

Output timing		Category	Log name	Details*1
(7)	Done	PROCEDURE	DETACH	[Procedure name]<tab>[Procedure handle]

*1. Refer to *Information Item Details (Stored Procedure)* on page 6-26 for details of each item.

● Information Item Details (Stored Procedure)

Item	Description
Procedure Name	A maximum of 60 bytes from the beginning are displayed. When multi-byte characters are used, the characters are displayed up to the delimiter.
Procedure Handle	Displays the value being output to the <i>ProcHandle</i> output variable for the DB_AttachProcedure instruction.
Attach error code	Refer to <i>Details of [Attach error code], [Attach error location], and [Attach error member name]</i> on page 6-27.
Procedure IF	Displays the procedure IF retrieved from the database. It is output in the following format: "return data type_procedure name (argument direction_argument 1 data type_argument 1 name,_argument 2 data type_argument 2 name)" *_ is a single-byte space. For the argument direction, IN/OUT/INOUT is output. All the data types output are those for the databases.
IN argument map variable name	Displays the variable name specified for the <i>ArgIn</i> input variable.
OUT argument map variable name	Displays the variable name specified for the <i>ArgOut</i> input variable.
INOUT argument map variable name	Displays the variable name specified for the <i>ArgInOut</i> input variable.
Return value map variable name	Displays the variable name specified for the <i>ReturnVal</i> input variable.
Result set map variable name	Displays the variable name specified for the <i>ResultSet</i> input variable.
Attach error location	Refer to <i>Details of [Attach error code], [Attach error location], and [Attach error member name]</i> on page 6-27.
Attach error member name	Refer to <i>Details of [Attach error code], [Attach error location], and [Attach error member name]</i> on page 6-27.
Procedure process name	Displays the internal process of the stored procedure. The processing varies by the timing (2), (3), (5) shown in the above figure. (2): CALL: Procedure Call (3): COMMIT: Commit (5): ROLLBACK: Rollback
Procedure call statement	It is displayed in the format corresponding to the database type. The value is output for the IN argument and OUT argument of a procedure. For the OUT argument of a procedure, only the OUT argument name of the procedure is output. Example: A stored procedure named StoredProc01 having two arguments consisting of an INT-type IN argument param1 (the value is 10) and an INT-type OUT argument param2 • SQLServer: EXEC StoredProc01 10, @param2 OUT; • Oracle: StoredProc01(10, :param2); • MySQL: CALL StoredProc01(10, @param2); • PostgreSQL: SELECT StoredProc01(10);
Return value	Displays the return value.
DB response time	An integer value in milliseconds is displayed.

Item	Description
DB error code	Error code that is specific to DB vendor of the device to connect. When a network error has occurred, 0 is displayed for DB error code in some cases. When 0 is displayed, check its SQL status.
Error message	Displays an error message.

● **Details of [Attach error code], [Attach error location], and [Attach error member name]**

Attach error code	Attach error location	Attach error member name	Description
Success	---	---	Succeeded
ExistOnlyOneSide	ArgIn	---	Although the stored procedure contains an IN argument, _DBC_Unused is set to ArgIn of the controller.
	ArgOut	---	Although the stored procedure contains an OUT argument, _DBC_Unused is set to ArgOut of the controller.
	ArgInOut	---	Although the stored procedure contains an INOUT argument, _DBC_Unused is set to ArgInOut of the controller.
	ReturnVal	---	Although the stored procedure does not contain a return value, a value other than _DBC_Unused is set to ReturnVal of the controller.
TypeNotMatch	ArgIn	Member name	The IN argument type of the stored procedure does not match the ArgIn member variable type on the controller.
	ArgOut	Member name	The OUT argument type of the stored procedure does not match the ArgOut member variable type on the controller.
	ArgInOut	Member name	The INOUT argument type of the stored procedure does not match the ArgInOut member variable type on the controller.
	ReturnVal	---	The return value type of the stored procedure does not match the ReturnVal type on the controller.
CountNotMatch	ArgIn	---	The number of IN arguments of the stored procedure does not match the number of ArgIn member variables on the controller.
	ArgOut	---	The number of OUT arguments of the stored procedure does not match the number of ArgOut member variables on the controller.
	ArgInOut	---	The number of INOUT arguments of the stored procedure does not match the number of ArgInOut member variables on the controller.
NameNotMatch	ArgIn	Member name	The ArgIn member variables on the controller contain a name that does not exist in the IN arguments of the stored procedure.
	ArgOut	Member name	The ArgOut member variables on the controller contain a name that does not exist in the OUT arguments of the stored procedure.
	ArgInOut	Member name	The ArgInOut member variables on the controller contain a name that does not exist in the INOUT arguments of the stored procedure.
DBMSSpecific	---	---	The OUT arguments of the stored procedure contain two or more cursor-type data. (Oracle and PostgreSQL only)

Log File Example

1	2012-07-24	09:00:00	150	SQL	0001	INSERT 0x0000	MyDatabase1	45
	TABLE_Production		Production	INSERT				INTO
	TABLE_Production("Column1") VALUES('1000')							
2	2012-07-24	09:00:00	200	SQL_RESULT	0001	INSERT 0x300B		
	MyDatabase1	46	17072	ORA-17072: Inserted value too large for co				
	lumn							



Precautions for Correct Use

Do not delete the latest log file (DB_DebugLog.log) and the log control file (DB_DebugLog.fjc) from the SD Memory Card. If they are deleted, the log files are not saved correctly, for example, the Debug Log data are lost.

6-4 SQL Execution Failure Log

This section describes the "SQL Execution Failure Log" used to trace the execution failures of the DB Connection Service due to a DB-caused factor.

6-4-1 Overview

You can check the SQL statements and error information when transmission of an SQL statement failed due to a problem*1 of the DB itself.

- *1. For example,
- a) Because the column names of the table have been changed, they do not match the column names of an SQL statement sent from the DB Connection Service.
 - b) A value to insert is outside the valid range of the data type of the column.

You can record this log by setting "SQL execution failure log" to "Record" in the DB Connection Service Setting of Sysmac Studio.

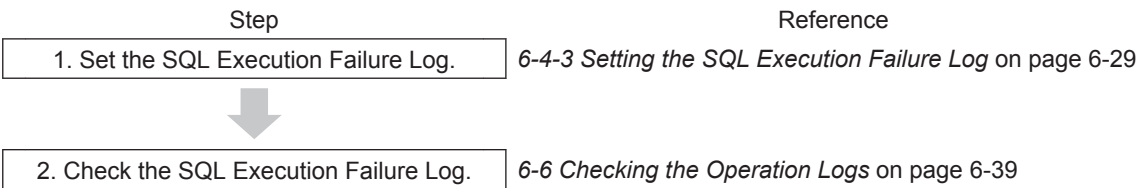
This log is saved as SQL Execution Failure Log files on the SD Memory Card mounted in the CPU Unit.

When no SD Memory Card is mounted in the CPU Unit, you cannot record the SQL Execution Failure Log.

You can check the contents of this log in the **SQL Execution Failure Log** Tab Page of the Operation Log Window in Sysmac Studio.

6-4-2 Application Procedure

Use the SQL Execution Failure Log according to the following procedure.



6-4-3 Setting the SQL Execution Failure Log

Double-click **DB Connection Service Settings** under **Configurations and Setup - Host Connection Settings - DB Connection** in the Multiview Explorer. Then, set the following in the Service Setting.

Item	Description	Values
SQL execution failure log	Set whether to record the SQL Execution Failure Log.	- Record - Do not record (Default)
Number of files	Set the maximum number of files of the SQL Execution Failure Log. When the maximum number of files is reached, the oldest file is deleted and a new file is created.	2 to 100 files (Default: 50)
File size	Set the maximum file size. When the maximum file size is exceeded or when the number of records exceeds 65,536 records in a file, a new file is created.	1 to 100 MB (Default: 10 MB)

6-4-4 Checking the SQL Execution Failure Log

Refer to 6-6 *Checking the Operation Logs* on page 6-39 for how to check the SQL Execution Failure Log.

6-4-5 SQL Execution Failure Log File Specifications

This section describes the specifications of SQL Execution Failure Log files.

- Each SQL Execution Failure Log file is composed of multiple records.
- Each record is expressed in one line.
- The maximum size of each SQL Execution Failure Log file is set on Sysmac Studio.
- The size of a single record is up to 58 KB.
- The following table shows the file name and type.

File name	File type
DB_SQLFailedLog.log	Latest log file of the log
DB_SQLFailedLog_[year_month_date_hours_minutes_seconds_milliseconds].log ^{*1} Example: DB_SQLFailedLog_20120724220915040.log	Previous log files
DB_SQLFailedLog.fjc	Log control file

^{*1} The system time of the CPU Unit is used for the time information included in the file name.

- The files are stored in the following directory (of the SD Memory Card).

a) Log files:

/packages/DB_Connection/SQLFailedLog/

b) Log control file:

/packages/DB_Connection/System/

- The following is the format of records.

Each record is expressed in one line and composed of multiple parameters. The parameters are separated from each other by a tab.

```
[Serial number]<tab>[Date]<tab>[Time]<tab>[Millisecond]<tab>[Category]<tab>[Log code]<tab>[Log  
name]<tab>[Result]<tab>[DB Connection name]<tab>[Serial ID]<tab>[Details]<CR><LF>
```

Parameter	Size	Description
Serial number	1 to 5 bytes	0 to 65535 When exceeding 65535, this value returns to 0. The serial number is given across multiple files. (Even if a new file is created, the serial number is not reset to 0.)
Date	10 bytes (Fixed)	Displays year, month, and date when the log was recorded. ^{*1} YYYY-MM-DD Example: 2012-07-23
Time	8 bytes (Fixed)	Displays hours, minutes, and seconds when the log was recorded. ^{*1} hh:mm:ss Example: 15:33:45
Millisecond	3 bytes (Fixed)	Displays 3-digit decimal integer (000 to 999) that shows millisecond of the time when the log was recorded. ^{*1} Example: 10 ms: 010 623 ms: 623
Category	16 bytes max. (Variable)	Displays the category. Refer to <i>Category</i> on page 6-31 for details.

Parameter	Size	Description
Log code	4 bytes (Fixed)	Displays a 4-digit decimal code that is a unique identification code in the category. Refer to <i>Log Code</i> on page 6-32 for details.
Log name	32 bytes max. (Variable)	Displays a name that shows the contents of the log. Refer to <i>Log Name</i> on page 6-32 for details.
Result	6 bytes (Fixed)	Displays a 4-digit hexadecimal code that shows the execution result. (e.g., 0x1234) 0x0000: Succeeded Other than 0x0000: Failed (Same code as ErrorID of DB Connection Instruction)
DB Connection name	16 bytes max. (Variable)	Displays a DB Connection name (single-byte alphanumeric characters)
Serial ID	10 bytes max. (Variable)	Displays the ID code assigned at the execution of record processing and stored procedure instructions. (The same ID as Serial ID displayed in the "SQL" or "SQL Resend" record of Execution Log is displayed.)
Details	Variable	Displays the details of the SQL Execution Failure Log. The contents differ according to the category. In the Details parameter, information items are separated from each other by a tab. Refer to <i>Information Item Details</i> on page 6-33 for details. [Category string: SQL_FAIL (Category: SQL execution failure)] - For INSERT/UPDATE/SELECT/DELETE [Table name]<tab>[DB Map Variable name]<tab>[DB error code]<tab>[Error message]<tab>[SQL statement] - For BATCHINSERT Refer to <i>Details for BATCHINSERT</i> on page 6-33. [Category string: PROCEDURE_FAIL (Category: PROCEDURE execution failure)] Refer to <i>Details of Stored Procedure</i> on page 6-34. [Category string: SPOOL (Category: Spooled)] [Table name]<tab>[DB Map Variable name]<tab>[SQL statement] [Category string: STATUS_ERROR (Category: Status Error)] - For INSERT/UPDATE/SELECT/DELETE [Table name]<tab>[DB Map Variable name]<tab>[SQL statement] - For BATCHINSERT Refer to <i>Details for BATCHINSERT</i> on page 6-33. - For EXECUTE Refer to <i>Details of Stored Procedure</i> on page 6-34.
Tab separation	10 bytes in total	
CR+LF	2 bytes	

*1. The date and time information follows the time zone set when the power supply to the Controller is turned ON. After you change the time zone, cycle the power supply.

● Category

Category	Category string
SQL Execution Failed	SQL_FAIL
Stored Procedure Execution Failure	PROCEDURE_FAIL

Category	Category string
Spooled	SPOOL
Status Error	STATUS_ERROR

● Log Code

Category	Code (decimal)	Operation	Log recording timing
SQL Execution Failed	0001	INSERT	When execution of an SQL statement issued from DB Connection Service to DB failed due to a DB-caused factor.
	0002	UPDATE	
	0003	SELECT	
	0004	DELETE	
	0005	BATCHINSERT	
PROCEDURE execution failure	0001	EXECUTE	When execution of an SQL statement issued from DB Connection Service to DB failed due to a DB-caused factor.
Spooled	0001	INSERT	When an SQL statement is stored in the Spool memory because a failure occurred in information exchange between DB Connection Service and DB.
	0002	UPDATE	
Status Error	0001	INSERT	- When the DB Connection Service detected an error and could not send an SQL statement. - When a failure occurred in information exchange between DB Connection Service and DB (when spooling is disabled) - When an SQL statement cannot be stored in the Spool memory because the Spool capacity is insufficient as a failure occurred in information exchange between DB Connection Service and DB
	0002	UPDATE	
	0003	SELECT	- When the DB Connection Service detected an error and could not send an SQL statement. - When a failure occurred in information exchange between DB Connection Service and DB. - When an SQL statement cannot be executed because one or more SQL statements are stored in the Spool memory.
	0004	DELETE	
	0005	BATCHINSERT	When the DB Connection Service detected an error and could not send an SQL statement.
	0010	PROCEDURE	- When a failure occurred in information exchange between DB Connection Service and DB. - When an SQL statement cannot be executed because one or more SQL statements are stored in the Spool memory.

● Log Name

Category	Operation	Log name
DB Connection Service	DB Connection Service Started	Start
	DB Connection Service Stopped	Stop
	Shutdown DB Connection Service	Shutdown
DB Connection	DB Connection Established	Connect
	DB Connection Closed	Close
	DB Connection Disconnected	Disconnect
	DB Connection Reestablished	Reconnect

Category	Operation	Log name
SQL	INSERT	INSERT
	UPDATE	UPDATE
	SELECT	SELECT
	DELETE	DELETE
	BATCHINSERT	BATCHINSERT
PROCEDURE	EXECUTE	EXECUTE
	DETACH	DETACH
	ATTACH	ATTACH
SQL Resend	INSERT	INSERT
	UPDATE	UPDATE
User-specified Log	DB_PutLog Instruction Executed	Text string specified in the <i>LogName</i> input variable of the DB_PutLog instruction.

● Information Item Details

Information	Description
Table name, DB Map Variable name	Displays Table name and DB Map Variable name.
DB Map Variable name	Variable name specified in the <i>MapVar</i> input variable (The POU instance name is not displayed. Nothing is displayed for DELETE.)
SQL statement	Displays the SQL statement.
DB error code	Error code that is specific to DB vendor of the device to connect. When a network error has occurred, 0 is displayed for DB error code in some cases. When 0 is displayed, check its SQL status.
Error message	Displays an error message.

Details for BATCHINSERT

The details for BATCHINSERT are described below.

The SQL execution failure log is output when an error occurs. The details being output when an error occurs contain the overall error information in the first row and the information corresponding to each element number for the subsequent rows.

Output timing	Category	Details ^{*1}
Error	SQL_FAIL	First row: [Table name]<tab>[DB Map Variable name]<tab>[Log record type:IN-FO]<tab>[Insert count]<tab>[DB error code]<tab>[Error message] Second row: [Table name]<tab>[DB Map Variable name]<tab>[Log record type:SQL]<tab>[Element number]<tab>[SQL statement] : Element number + First row: [Table name]<tab>[DB Map Variable name]<tab>[Log record type:SQL]<tab>[Element number]<tab>[SQL statement]
	STATUS_ERROR	First row: [Table name]<tab>[DB Map Variable name]<tab>[Log record type:IN-FO]<tab>[Insert count] Second row: [Table name]<tab>[DB Map Variable name]<tab>[Log record type:SQL]<tab>[Element number]<tab>[SQL statement] : Element number + First row: [Table name]<tab>[DB Map Variable name]<tab>[Log record type:SQL]<tab>[Element number]<tab>[SQL statement]

*1. Refer to *Information Item Details (for BATCHINSERT)* on page 6-34 for details of each item.

● Information Item Details (for BATCHINSERT)

Item	Description
Table name, DB Map Variable name	Displays Table name and DB Map Variable name.
Log record type	Displays the log record type. • INFO: Displays the number of inserted records, DB error code, and error message • SQL: Displays the SQL statement of each record
Insert count	The value of the <i>InsertCnt</i> input variable is displayed. (1 to 5 bytes)
DB error code	Error code that is specific to DB vendor of the device to connect. When a network error has occurred, 0 is displayed for DB error code in some cases. When 0 is displayed, check its SQL status.
Error message	Displays an error message.
Element number	Displays the element number of the array being processed.
SQL statement	Displays the SQL statement.

Details of Stored Procedure

The details of the stored procedure are described below.

The SQL execution failure log is output when an error occurs.

Output timing	Category	Details*1
Error	PROCEDURE_FAIL	[Procedure name]<tab>[Procedure handle]<tab>[DB error code]<tab>[Error message]<tab>[Procedure call statement]
	STATUS_ERROR	[Procedure name]<tab>[Procedure handle]<tab>[Procedure call statement]

*1. Refer to *Information Item Details (Stored Procedure)* on page 6-34 for details of each item.

● Information Item Details (Stored Procedure)

Item	Description
Procedure Name	A maximum of 60 bytes from the beginning are displayed. When multi-byte characters are used, the characters are displayed up to the delimiter.
Procedure Handle	Displays the value being output to the <i>ProcHandle</i> output variable for the DB_AttachProcedure instruction.
DB error code	Error code that is specific to DB vendor of the device to connect. When a network error has occurred, 0 is displayed for DB error code in some cases. When 0 is displayed, check its SQL status.
Error message	Displays an error message.
Procedure call statement	It is displayed in the format corresponding to the database type. The value is output for the IN argument and OUT argument of a procedure. For the OUT argument of a procedure, only the OUT argument name of the procedure is output. Example: A stored procedure named StoredProc01 having two arguments consisting of an INT-type IN argument param1 (the value is 10) and an INT-type OUT argument param2 • SQLServer: EXEC StoredProc01 10, @param2 OUT; • Oracle: StoredProc01(10, :param2); • MySQL: CALL StoredProc01(10, @param2); • PostgreSQL: SELECT StoredProc01(10);

Log File Example

1	2012-07-24	09:00:00	200	SQL_FAIL	0001	INSERT	0x300B	
	MyDatabase1	0	17072	ORA-17072: Inserted value too large for column				
	INSERT INTO TABLE_Production(Column1)VALUES('1000')							
2	2012-07-24	09:01:13	550	SPOOL	0001	INSERT	0x3012	MyDatabase1
	15	INSERT INTO TABLE_Production(Column2)VALUES('200')						
3	2012-07-24	09:01:14	050	SPOOL	0001	INSERT	0x3014	MyDatabase1
	18	INSERT INTO TABLE_Production(Column2)VALUES('300')						
4	2012-07-24	09:01:14	550	STATUS_ERROR	0001	INSERT	0x300C	
	MyDatabase1	19	INSERT INTO TABLE_Production(Column2) VALUES('400')					



Precautions for Correct Use

Do not delete the latest log file (DB_SQLFailedLog.log) and the log control file (DB_SQLFailedLog.fjc) from the SD Memory Card. If they are deleted, the log files are not saved correctly, for example, the SQL Execution Failure Log data are lost.

6-5 SD Memory Card Operations

In the DB Connection Service, the SD Memory Card mounted in the CPU Unit is used for the Operation Log function.

The Execution Log files, Debug Log files, and SQL Execution Failure Log files are stored in the SD Memory Card.

This section describes how to save the log files on the SD Memory Card and precautions for replacing the SD Memory Card.

Refer to the *NJ/NX-series CPU Unit Software User's Manual (W501)* for details of the SD Memory Card functions.

6-5-1 Saving Operation Log Files on SD Memory Card

Each Operation Log file is stored in the SD Memory Card in the following conditions.

Operation Logs	Operation to use the function	Conditions for saving log files on SD Memory Card
Execution Log	Set Execution log to Record in the DB Connection Service Settings of Sysmac Studio.	Constantly saved while the DB Connection Service is running.*1
Debug Log	Right-click DB Connection Service Settings in the Multiview Explorer on Sysmac Studio and select Online Settings from the menu. Then, click the Start Button for Debug Log in the Online Settings Tab Page. Or Execute a DB_ControlService (Control DB Connection Service) instruction to start recording to the Debug Log.	Constantly saved while the Debug Log is recorded.
SQL execution failure log	Set SQL execution failure log to Record in the DB Connection Service Settings of Sysmac Studio.	Saved when transmission of an SQL statement failed due to a DB-caused factor.*2

*1. If the power supply to the CPU Unit is turned ON while no SD Memory Card is mounted in the CPU Unit, an "Execution Log Save Failed Error" is registered into the event log when the Execution Log is saved. Recording to the Execution Log is started when an SD Memory Card is inserted into the CPU Unit.

*2. If the power supply to the CPU Unit is turned ON while no SD Memory Card is mounted in the CPU Unit, an "SQL Execution Failure Log Save Failed Error" is registered into the event log when the SQL Execution Failure Log is saved. Recording to the SQL Execution Failure Log is started when an SD Memory Card is inserted into the CPU Unit.

6-5-2 Directory Used for DB Connection Service

The DB Connection Service uses the directory under "packages/DB_Connection" in the SD Memory Card.

packages/DB_Connection/System	: Contains log control files.
packages/DB_Connection/ExecutionLog	: Contains Execution Log files.
packages/DB_Connection/DebugLog	: Contains Debug Log files.
packages/DB_Connection/SQLFailedLog	: Contains SQL Execution Failure Log files.

6-5-3 Operation Log Operations in Replacing the SD Memory Card

This section describes operations of each Operation Log when the SD Memory Card is replaced while the DB Connection Service is running.

Operation Log function	SD Memory Card Replacing Status		
	When the SD Memory Card power supply switch is pressed	When no SD Memory Card is mounted	When an SD Memory Card is inserted
Execution Log	Continued If Execution Log is contained in the internal buffer of the CPU Unit, it is recorded into the SD Memory Card.	Temporarily recorded into the internal buffer of the CPU Unit.	The log that is temporarily recorded in the internal buffer is automatically recorded to the SD Memory Card.
Debug Log	Stopped. If Debug Log is contained in the internal buffer of the CPU Unit, it is recorded into the SD Memory Card.	Debug Log is not recorded.	Recording to the Debug Log is still stopped. Recording is started by an online operation from Sysmac Studio or by executing a DB_ControlService (Control DB Connection Service) instruction.
SQL Execution Failure Log	Stopped. If SQL Execution Failure Log is contained in the internal buffer of the CPU Unit, it is recorded into the SD Memory Card.	SQL Execution Failure Log is not recorded.	Recording to the SQL Execution Failure Log is automatically started.



Precautions for Correct Use

Please note the following for replacing the SD Memory Card.

- Use a formatted SD Memory Card when replacing the SD Memory Card.
- When you replace the SD Memory Card while recording the Execution Log, press the SD Memory Card power supply switch and insert a new SD Memory Card within five minutes after the SD PWR indicator is turned OFF.
If it takes more than five minutes, Execution Log recorded in the internal buffer may be lost.
If the internal buffer space becomes full before inserting the SD Memory Card, an "Execution Log Save Failed Error" is registered into the event log.

6-5-4 Guidelines for SD Memory Card Replacement Time

If you replace the SD Memory Card while the DB Connection Service is running, replace the SD Memory Card within the following time. The guidelines for SD memory card replacement time depends on the CPU Unit model and the execution interval of the DB Connection instruction.

CPU Unit model	Execution Interval of the DB Connection Instructions		
	50 ms	100 ms	500 ms
NJ501-□□20	30 s	60 s	300 s (5 min.)
NJ101-□□20			
NX701-□□20	300 s (5 min.)	600 s (10 min.)	3,000 s (50 min.)
NX102-□□20	30 s	60 s	300 s (5 min.)



Precautions for Correct Use

When replacing the SD Memory Card, observe the followings:

- Use a formatted SD Memory Card when replacing the SD Memory Card.
 - When you replace the SD Memory Card while recording the Execution Log, press the SD Memory Card power supply switch and insert a new SD Memory Card within the above guideline for replacement time after the SD PWR indicator is turned OFF.
If the replacement time is exceeded the guideline, Execution Log recorded in the internal buffer may be lost.
If the internal buffer space becomes full before inserting the SD Memory Card, an "Execution Log Save Failed Error" is registered into the event log.
 - If you exceed the guidelines for the SD Memory Card replacement time, stop the equipment temporarily or select **Do not record** of the **Execution Log** in the DB Connection Service Settings. Make sure that the Execution Log is not recorded before replacing the SD Memory Card. Refer to *2-2-1 DB Connection Service Settings* on page 2-5 for details.
-

6-5-5 Replacement Timing of SD Memory Card

Replace the SD Memory Card in the following cases.

- The "SD Memory Card Life Exceeded" Event occurred.
- The system-defined variable `_Card1Deteriorated` (SD Memory Card Life Warning Flag) became TRUE.

6-6 Checking the Operation Logs

This section describes how to check the Operation Logs stored on the SD Memory Card mounted in the CPU Unit.

6-6-1 How to Check the Operation Logs

You can use the following methods to check the Operation Logs (i.e., Execution Log, Debug Log, and SQL Execution Failure Log).

- Checking the log on the Operation Log Window in Sysmac Studio
- Checking the log with the SD Memory Card
- Checking the log by transferring data using FTP client software



Precautions for Correct Use

Each Operation Log file is encoded by the UTF-8 character code.

6-6-2 Checking the Log on the Operation Log Window in Sysmac Studio

You can check the Operation Logs (i.e., Execution Log, Debug Log, and SQL Execution Failure Log) stored in the SD Memory Card on the Operation Log Window in Sysmac Studio while online with the CPU Unit.

- 1 Right-click **DB Connection** under **Configurations and Setup - Host Connection Settings** in the Multiview Explorer and select **Show Operation Logs** from the menu while online with the CPU Unit.
- 2 The Execution Log, Debug Log, and SQL Execution Failure Log are displayed in the different tab pages.

Click the **Execution Log** Tab, **Debug Log** Tab, or **SQL Execution Failure Log** Tab.

The following information is displayed.

Entry	Date/Time	Category	Log Code	Log Name	Result	Connection Name	Serial ID
00000	1/24/2013 19:43:15.794	DB_CONNECTION	0001	Connect	0x0000	DB_Connect_2	
00001	1/24/2013 19:44:20.850	DB_CONNECTION	0001	Connect	0x0000	DB_Connect_2	
00002	1/24/2013 19:44:30.507	SQL	0001	INSERT	0x0000	DB_Connect_2	0000000000
00003	1/24/2013 19:44:31.430	SQL	0001	INSERT	0x0000	DB_Connect_2	0000000001
00004	1/24/2013 19:44:32.430	SQL	0001	INSERT	0x0000	DB_Connect_2	0000000002
00005	1/24/2013 19:44:33.430	SQL	0001	INSERT	0x0000	DB_Connect_2	0000000003
00006	1/24/2013 19:44:34.430	SQL	0001	INSERT	0x0000	DB_Connect_2	0000000004

Details 12514

Upload

Clear

List view

Detailed information

Buttons

- List view

Item	Description
Entry	Displays a serial number.
Date/Time	Displays a date and time.
Category	Displays a category.
Log Code	Displays a log code.

Item	Description
Log Name	Displays a log name.
Result	Displays results.
Connection Name	Displays a DB Connection name.
Serial ID	Displays a serial ID.

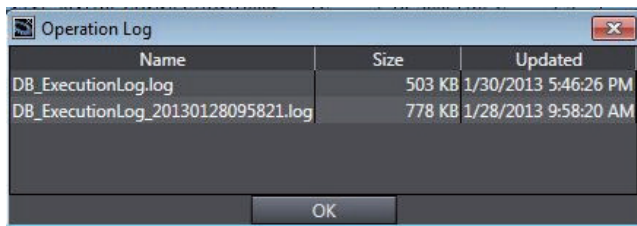
- Detailed information

The **Details** parameter of the log is displayed.

- Buttons

Upload Button:

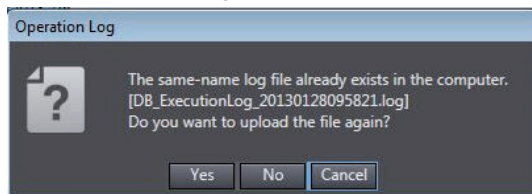
The log files are uploaded from the Controller. A list of log files is displayed in the following **Operation Log** Dialog Box.



Select a log file to display and click the **OK** Button. The log file is uploaded.

- Execution Log** Tab Page: Execution Log is uploaded from the Controller.
- Debug Log** Tab Page: Debug Log is uploaded from the Controller.
- SQL Execution Failure Log** Tab Page: SQL Execution Failure Log is uploaded from the Controller.

Note 1. If the same-name log file exists in the computer, the following message is displayed.



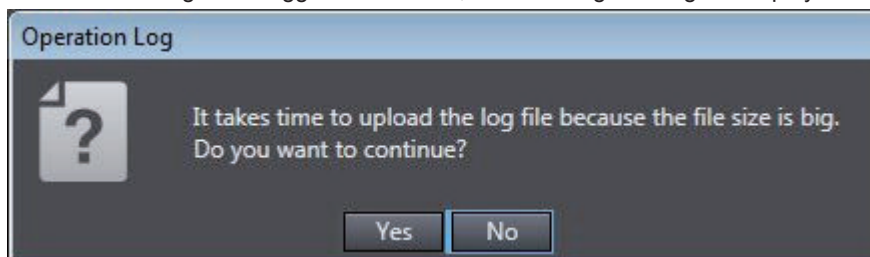
Click a button.

Yes: The specified file is uploaded from the Controller and displayed.

No: The specified file is not uploaded from the Controller and the contents of the file that already exists in the computer are displayed.

Cancel: The file list is displayed again.

Note 2. If the selected log file is bigger than 10 MB, the following message is displayed.



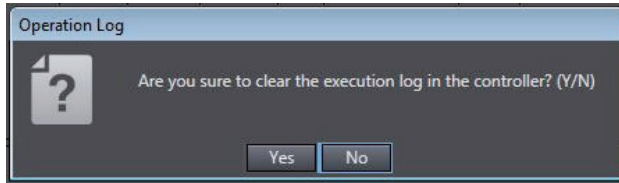
Click a button.

Yes: The specified file is uploaded from the Controller and displayed.

No: The file list is displayed again.

Clear Button:

The selected Operation Log is cleared in the Controller. A confirmation message is displayed.



When you click the **Yes** Button, the selected log is cleared.

- a) **Execution Log** Tab Page: Execution Log is cleared in the Controller.
- b) **Debug Log** Tab Page: Debug Log is cleared in the Controller.
- c) **SQL Execution Failure Log** Tab Page: SQL Execution Failure Log is cleared in the Controller.

6-6-3 Checking the Log with the SD Memory Card

Remove the SD Memory Card from the CPU Unit and insert it into a computer. Then, check the contents of the logs on Microsoft Excel or a text editor.

6-6-4 Checking the Log by Transfer using FTP Client Software

You can transfer the log files using the FTP Server function via the Ethernet network and check the contents on Microsoft Excel or a text editor.

Use the following procedure.

You use the FTP Server function of the built-in EtherNet/IP port.

- 1** Double-click **Built-in EtherNet/IP Port Settings** under **Configurations and Setup** - Controller Setup in the Multiview Explorer and set **FTP server** to **Use** in the **FTP Settings**.
- 2** Log into the CPU Unit using the FTP client software.
- 3** Transfer Operation Log files.
You can transfer more than one log file by using a wildcard in the Mget command.
Example: mget DB_ExecutionLog_*.log
- 4** Disconnect the FTP client software from the CPU Unit.
- 5** Open the transferred Operation Log files on Microsoft Excel or a text editor to check the contents.

DB Connection Instructions

DB Connection Instructions and Variables.....	7-2
DB_Connect (Establish DB Connection).....	7-6
DB_Close (Close DB Connection)	7-10
DB_CreateMapping (Create DB Map)	7-13
DB_Insert (Insert DB Record).....	7-17
DB_Update (Update DB Record).....	7-21
DB_Select (Retrieve DB Record)	7-39
DB_Delete (Delete DB Record).....	7-45
DB_ControlService (Control DB Connection Service).....	7-60
DB_GetServiceStatus (Get DB Connection Service Status)	7-66
DB_GetConnectionStatus (Get DB Connection Status)	7-71
DB_ControlSpool (Resend/Clear Spool Data)	7-77
DB_PutLog (Record Operation Log).....	7-84
DB_Shutdown (Shutdown DB Connection Service)	7-90
DB_BatchInsert (DB Records Batch Insert).....	7-94
DB_AttachProcedure (Generate DB Stored Procedure Handle)	7-107
DB_ExecuteProcedure (Execute DB Stored Procedure)	7-112
DB_DetachProcedure (Release DB Stored Procedure Handle)	7-124

DB Connection Instructions and Variables

DB Connection Instruction Set

This section gives a list of DB Connection Instructions.

Instruction	Name	Supported DB Connection Service versions	Page
DB_Connect	Establish DB Connection	1.00 or higher	page 7-6
DB_Close	Close DB Connection	1.00 or higher	page 7-10
DB_CreateMapping	Create DB Map	1.00 or higher	page 7-13
DB_Insert	Insert DB Record	1.00 or higher	page 7-17
DB_Update	Update DB Record	1.00 or higher	page 7-21
DB_Select	Retrieve DB Record	1.00 or higher	page 7-39
DB_Delete	Delete DB Record	1.00 or higher	page 7-45
DB_ControlService	Control DB Connection Service	1.00 or higher	page 7-60
DB_GetServiceStatus	Get DB Connection Service Status	1.00 or higher	page 7-66
DB_GetConnectionStatus	Get DB Connection Status	1.00 or higher	page 7-71
DB_ControlSpool	Resend/Clear Spool Data	1.00 or higher	page 7-77
DB_PutLog	Record Operation Log	1.00 or higher	page 7-84
DB_Shutdown	Shutdown DB Connection Service	1.00 or higher	page 7-90
DB_BatchInsert	DB Records Batch Insert	2.00 or higher	page 7-94
DB_AttachProcedure	Generate DB Stored Procedure Handle	2.00 or higher	page 7-107
DB_ExecuteProcedure	Execute DB Stored Procedure	2.00 or higher	page 7-112
DB_DetachProcedure	Release DB Stored Procedure Handle	2.00 or higher	page 7-124

Variables Used in the DB Connection Instructions

This section describes the details of the variables used in the DB Connection Instructions.

Common Input and Output Variables Used in the DB Connection Instructions

● DBConnection

Input variable	Meaning	Data type	Description
DBConnection	DB Connection	DWORD	DB Connection output from a DB_Connect instruction. The instructions are executed for a specified DB Connection.

● ServiceStatus

Output variable	Meaning	Data type	Description
Member			
ServiceStatus	DB Connection Service Status	_sDBC_SERVICE_STATUS	Structure to show the status of the DB Connection Service.
Status	Service Status	_eDBC_STATUS	Enumeration data type to show the service status _DBC_STATUS_IDLE(0): Idle _DBC_STATUS_RUNNING(1): Running in Operation Mode _DBC_STATUS_TEST(2): Running in Test Mode
DebugLog	Debug Log Flag	BOOL	TRUE while the Debug Log is recorded. FALSE while recording to the Debug Log is stopped.
Operating-Time	Operating Time	TIME	Time elapsed since the service was started.
ExecCnt	Number of Normal Executions	DINT	Total number of times in all connections when an SQL statement was normally executed.
FailedCnt	Number of Error Executions	DINT	Total number of times in all connections when an SQL statement execution failed.
SpoolDataCnt	Number of Spool Data	DINT	Number of SQL statements stored in the Spool memory in all connections.

● ConnectionStatus

Output variable	Meaning	Data type	Description
Member			
ConnectionStatus	DB Connection Status	_sDBC_CONNECTION_STATUS	Structure to show the status of a DB Connection.
Status	Connection Status	_eDBC_CONNECTION_STATUS	Enumeration data type to show the status of a DB Connection _DBC_CONNECTION_STATUS _CLOSED(0): Closed _DBC_CONNECTION_STATUS _CONNECTED(1) : Connected _DBC_CONNECTION_STATUS _DBC_CONNECTION_STATUS_DISCONNECTED(2): Disconnected (Disconnected due to a network failure while the DB is connected.)
Connected-Time	Connected Time	TIME	Total time when the DB is connected.
Disconnected-Time	Disconnected Time	TIME	Total time when the DB is disconnected due to an error.
ExecCnt	Number of Normal Executions	DINT	Number of times when an SQL statement was executed normally in the DB Connection.
FailedCnt	Number of Error Executions	DINT	Number of times when an SQL statement execution failed in the DB Connection.
DBRespTime	DB Response Time	TIME	Time since an SQL statement is sent from the CPU Unit until the SQL execution result is returned from the CPU Unit when an SQL statement is executed. This is stored only when a normal response is returned from the DB. If an instruction execution timeout occurred, the DB Response Time is not stored when the instruction execution is completed (i.e. when the Error output variable changes from FALSE to TRUE). (The previous DB Response Time is held.) The new DB Response Time is stored when a normal response is returned from the DB after the instruction execution timeout.
SpoolDataCnt	Number of Spool Data	INT	Number of SQL statements stored in the Spool memory for the DB Connection.
SpoolUsageRate	Spool usage in percentage	SINT	Use rate of the Spool memory for the DB Connection. The unit is percentage (%).
ErrorDateTime	Disconnection Date/Time	DATE_AND_TIME	Date and time the last time the connection was disconnected due to an error.
SQLSTATE	SQL status	STRING(8)	Error code* ² defined in SQL Standards (ISO/IEC9075) for disconnection* ¹
ErrorCode	Error Code	DINT	Error code* ² for disconnection* ¹ , which is specific to DB vendor
ErrorMsg	Error Message	STRING(128)	Error message* ² for disconnection* ¹ , which is specific to DB vendor

*1. When a network failure or an SQL Execution Error occurred

- *2. The value may differ by unit version of the CPU Unit. The value of connection error to SQL Server was changed in the unit version 1.08 of the CPU Units.

● SendStatus

Output variable	Meaning	Data type	Description
SendStatus	Send Status	_eDBC_SEND_STATUS	Enumeration data type that shows transmission status of the SQL statement to DB _DBC_SEND_INIT(0): Initial status _DBC_SEND_UNSENT(1): SQL statement unsent _DBC_SEND_SENDING(2): Sending SQL statement _DBC_SEND_SPOOLED(3): SQL statement spooled _DBC_SEND_COMPLETE(4): SQL statement transmission completed

Common Variables Used in NJ/NX-series Instructions

Input Variable	Meaning	Data type	Description
Execute	Execute	BOOL	The instruction is executed when Execute changes to TRUE.

Output variable	Meaning	Data type	Description
Done	Done	BOOL	Shows whether the instruction is normally completed. TRUE: Normally completed FALSE: Terminated due to an error, being executed or execution conditions not satisfied
Busy	Executing	BOOL	Shows whether the instruction is being executed. TRUE: Being executed FALSE: Not being executed
Error	Error	BOOL	Shows whether the instruction is terminated due to an error. TRUE: Terminated due to an error FALSE: Terminated due to an error, being executed or execution conditions not satisfied
ErrorID	Error Code	WORD	Contains the error code when the instruction is terminated due to an error. WORD#16#0 indicates normal execution.

System-defined Variables Related to DB Connection Service

Variables	Meaning	Data type	Description
_DBC_Status	DB Connection Service Status	_sDBC_STATUS	System-defined variable that shows the status of the DB Connection Service.
_DBC_Unused	DB Connection Input Variable Omitted	BOOL	The system-defined variable used for omitting the input variable for the DB_AttachProcedure instruction if the stored procedure's argument, return value, or result set does not exist

Refer to 3-5-4 *System-defined Variables* on page 3-26 for details of the system-defined variables.

DB_Connect (Establish DB Connection)

The DB_Connect instruction connects to a specified DB.

Instruction	Name	FB/FUN	Graphic expression	ST expression
DB_Connect	Establish DB Connection	FB		DB_Connect_instance(Execute, DBConnectionName, Done, Busy, Error, ErrorID, DBConnection);

Note The DB_Connect_instance is an instance of DB_Connect instruction, which is declared as a variable.

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.
DBConnectionName	DB Connection name	STRING	17 bytes max. (including the final NULL character)	---	''	Specify a DB Connection name set on Sysmac Studio.

Output Variable

Name	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.
Error	Error	BOOL	TRUE or FALSE	---	TRUE when the instruction is terminated due to an error.
ErrorID	Error Code	WORD	16#0000 to 16#FFFF	---	Contains the error code when an error occurs.
DBConnection	DB Connection	DWORD	16#00000000 to 16#FFFFFFFF	---	Outputs a DB Connection. Specify this DB Connection in DB_CreateMapping, DB_Insert, DB_Update, DB_Select, DB_Delete, and DB_Close instructions.

Related System-defined Variables

Name	Meaning	Data type	Description
_EIP_EtnOnlineSta	Online	BOOL	Status of the communications function of the built-in EtherNet/IP port. TRUE: Can be used. FALSE: Cannot be used.

Related Error Codes

Error code	Meaning	Description
0406 hex	Illegal Data Position Specified	When the <i>DBConnectionName</i> input variable is a text string consisting of NULL characters (16#00) only.
0410 hex	Text String Format Error	A space character is included in the text string specified for the <i>DBConnectionName</i> input variable. When the <i>DBConnectionName</i> input variable does not end in NULL.
041D hex	Too Many Instructions Executed at the Same Time	More than 32 DB Connection Instructions were executed at the same time.
3000 hex	DB Connection Service not Started	The instruction was executed when the DB Connection Service was not running.
3002 hex	DB Connection Service Shutdown or Shutting Down	The instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
3003 hex	Invalid DB Connection Name	When the DB Connection name specified in the <i>DBConnectionName</i> input variable is not set in any DB Connection Settings.
3004 hex	DB Connection Rejected	The DB set in the DB Connection Settings rejected the connection.
3005 hex	DB Connection Failed	The DB Connection Service cannot communicate with the DB due to a network failure or other factors. The address set in the DB Connection Settings is wrong.
3006 hex	DB Connection Already Established	When a same-name DB Connection is already established.
3007 hex	Too Many DB Connections	When the maximum number of connections that can be established at the same time is exceeded.
3008 hex	Invalid DB Connection	The instruction was executed for the same connection at the same time.
3013 hex	DB Connection Service Error Stop	The instruction was executed while the DB Connection Service was stopped due to an error.
3015 hex	DB Connection Service Initializing	The instruction was executed while the initialization processing of the DB Connection Service was in progress.

Function

This instruction is used to connect to the DB specified in the *DBConnectionName* input variable.

The *DB Connection name* is set in the DB Connection Settings on Sysmac Studio.

When this instruction is normally completed (i.e. when the *Done* output variable changes to TRUE), a DB Connection is established and a value is output to the *DBConnection* output variable. This value is used to specify a DB Connection in some instructions described below.

Precautions for Correct Use

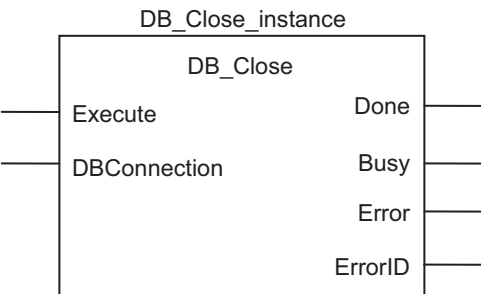
- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual* (Cat. No. W502) for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- This instruction cannot be used on an event task. A compiling error will occur.
- This instruction can be used only for the built-in EtherNet/IP port of an NJ/NX-series CPU Unit. It is impossible to connect to a DB via an EtherNet/IP Unit connected to an NJ/NX-series CPU Unit.
- The DB Connection created by this instruction is closed in the following cases.
 - a) When a DB_Close or DB_Shutdown instruction is executed.
 - b) When the operating mode of the Controller is changed from RUN mode to PROGRAM mode.
 - c) When the DB Connection Service is stopped.
- Refer to 1-2-1 *DB Connection Service Specifications* on page 1-5 for the number of DB Connections that can be established at the same time.
- When the DB Connection Service was started in Test Mode, this instruction is completed normally without connecting to the DB actually.
- When a same-name DB Connection is already established, the already-established DB Connection is output to the *DBConnection* output variable.
- An error occurs for this instruction in the following cases. *Error* will be TRUE.
 - a) When the instruction was executed when the DB Connection Service was not running.
 - b) When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
 - c) When the instruction was executed while the DB Connection Service was stopped due to an error.
 - d) When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
 - e) When the DB Connection name specified in the *DBConnectionName* input variable is not set in any DB Connection Settings.
 - f) When the *DBConnectionName* input variable is a text string consisting of NULL characters (16#00) only.
 - g) A space character is included in the text string specified for the *DBConnectionName* input variable.
 - h) When the *DBConnectionName* input variable does not end in NULL.
 - i) When the connection could not be established because the address set in the DB Connection Settings was wrong.
 - j) When the DB set in the DB Connection Settings rejected the connection.
 - k) When the DB Connection Service cannot communicate with the DB due to a network failure or other causes.
 - l) When the instruction was executed for the same connection at the same time.
 - m) When a same-name DB Connection is already established.
 - n) When the maximum number of connections that can be established at the same time is exceeded.
 - o) When more than 32 DB Connection Instructions were executed at the same time.

Sample Programming

Refer to *Sample Programming* on page 7-25 for the sample programming that is provided for the DB_Update instruction.

DB_Close (Close DB Connection)

The DB_Close instruction closes the connection with the DB established by a DB_Connect (Establish DB Connection) instruction.

Instruction	Name	FB/FUN	Graphic expression	ST expression
DB_Close	Close DB Connection	FB		DB_Close_instance (Execute, DBConnection, Done, Busy, Error, ErrorID);

Note The DB_Close_instance is an instance of DB_Close instruction, which is declared as a variable.

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.
DBConnection	DB Connection	DWORD	16#00000000 to 16#FFFFFFFF	---	16#00000000	Specify the DB connection established by a DB_Connect instruction.

Output Variable

Name	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.
Error	Error	BOOL	TRUE or FALSE	---	TRUE when the instruction is terminated due to an error.
ErrorID	Error Code	WORD	16#0000 to 16#FFFF	---	Contains the error code when an error occurs.

Related System-defined Variables

Name	Meaning	Data type	Description
_EIP_EtnOnlineSta	Online	BOOL	Status of the communications function of the built-in EtherNet/IP port. TRUE: Can be used. FALSE: Cannot be used.

Related Error Codes

Error code	Name	Meaning
041D hex	Too Many Instructions Executed at the Same Time	More than 32 DB Connection Instructions were executed at the same time.
3000 hex	DB Connection Service not Started	The instruction was executed when the DB Connection Service was not running.
3002 hex	DB Connection Service Shut-down or Shutting Down	The instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
3008 hex	Invalid DB Connection	When the value of the <i>DBConnection</i> input variable is invalid or the specified DB Connection is already closed.
3013 hex	DB Connection Service Error Stop	The instruction was executed while the DB Connection Service was stopped due to an error.
3015 hex	DB Connection Service Initializing	The instruction was executed while the initialization processing of the DB Connection Service was in progress.

Function

This instruction is used to close the DB Connection specified in the *DBConnection* input variable.

Precautions for Correct Use

- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual* (Cat. No. W502) for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- This instruction cannot be used on an event task. A compiling error will occur.
- When the DB Connection Service was started in Test Mode, this instruction is completed normally without connecting to the DB actually.
- An error occurs for this instruction in the following cases. *Error* will be TRUE.
 - a) When the instruction was executed when the DB Connection Service was not running.
 - b) When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
 - c) When the instruction was executed while the DB Connection Service was stopped due to an error.
 - d) When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
 - e) When the value of the *DBConnection* input variable is invalid or the specified DB Connection is already closed.
 - f) When more than 32 DB Connection Instructions were executed at the same time.

Sample Programming

Refer to *Sample Programming* on page 7-25 for the sample programming that is provided for the DB_Update instruction.

DB_CreateMapping (Create DB Map)

The DB_CreateMapping instruction creates a mapping from a DB Map Variable to a table of a DB.

Instruction	Name	FB/FUN	Graphic expression	ST expression
DB_CreateMapping	Create DB Map	FB		DB_CreateMapping_instance (Execute, DBConnection, TableName, MapVar, SQLType, Done, Busy, Error, ErrorID);

Note The DB_CreateMapping_instance is an instance of DB_CreateMapping instruction, which is declared as a variable.

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.
DBConnection	DB Connection	DWORD	16#00000000 to 16#FFFFFFFF	---	16#00000000	Specify the DB connection established by a DB_Connect instruction.
TableName	Table Name	STRING	Depends on the data type.*1	---	''	Specify a table name in the DB.
MapVar	DB Map Variable	Structure, Structure array (entire array)	Depends on the data type.	---	---	Specify a structure variable defined for accessing the DB.

Name	Meaning	Data type	Valid range	Unit	Default	Description
SQLType	SQL type	_eDBC _SQLTYPE	_DBC_SQLTYPE _INSERT(1): INSERT _DBC_SQLTYPE _UPDATE(2): UPDATE _DBC_SQLTYPE _SELECT(3): SELECT _DBC_SQLTYPE_BATCH- INSERT(4): BatchInsert	---	0	Specify a type of record processing for the variable to map.

- *1. When the database is case sensitive, specify the table name as shown below.
 When connecting to MySQL, enclose the table name in single-byte backquotes.
 Example: `TableName1`
 When connecting to other databases, enclose the table name in single-byte double quotes.
 Example: "TableName1"

Output Variable

Name	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.
Error	Error	BOOL	TRUE or FALSE	---	TRUE when the instruction is terminated due to an error.
ErrorID	Error Code	WORD	16#0000 to 16#FFFF	---	Contains the error code when an error occurs.

Related System-defined Variables

Name	Meaning	Data type	Description
_EIP_EtnOnlineSta	Online	BOOL	Status of the communications function of the built-in EtherNet/IP port. TRUE: Can be used. FALSE: Cannot be used.

Related Error Codes

Error code	Meaning	Description
0400 hex	Input Value Out of Range	A value that is not defined as an enumerator was specified in the <i>SQLType</i> input variable.
0406 hex	Illegal Data Position Specified	The <i>TableName</i> input variable is a text string consisting of NULL characters (16#00) only.
0410 hex	Text String Format Error	A space character is included in the text string specified for the <i>TableName</i> input variable.
041B hex	Data Capacity Exceeded	The upper limit of DB Map Variables for a single DB Connection is exceeded.
041D hex	Too Many Instructions Executed at the Same Time	More than 32 DB Connection Instructions were executed at the same time.

Error code	Meaning	Description
3000 hex	DB Connection Service not Started	The instruction was executed when the DB Connection Service was not running.
3002 hex	DB Connection Service Shutdown or Shutting Down	The instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
3008 hex	Invalid DB Connection	When the value of the <i>DBConnection</i> input variable is invalid or the specified DB Connection is already closed.
3009 hex	Invalid DB Map Variable	The data type of the variable specified in the <i>MapVar</i> input variable is not a structure. A derivative data type is included as a member of the structure variable specified in the <i>MapVar</i> input variable. The DB Map Variable specified in the <i>MapVar</i> input variable is a structure array though INSERT or UPDATE is specified for the SQL Type. When a variable other than a structure array was specified in the <i>MapVar</i> input variable for BATCHINSERT. When a variable that is not one-dimensional array was specified in the <i>MapVar</i> input variable for BATCHINSERT.
300B hex	SQL Execution Error	The executed SQL statement resulted in an error in the DB.
3011 hex	DB Connection Disconnected Error Status	The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
3013 hex	DB Connection Service Error Stop	The instruction was executed while the DB Connection Service was stopped due to an error.
3015 hex	DB Connection Service Initializing	The instruction was executed while the initialization processing of the DB Connection Service was in progress.
3019 hex	Instruction Executed for Unsupported Database Type	The instruction was executed for a database type that is not supported by this instruction.

Function

This instruction is used to map the table specified in the *TableName* input variable with a DB Map Variable specified in the *MapVar* input variable.

You need to execute this instruction before executing a DB_Insert, DB_Update, DB_Select, or DB_BatchInsert instruction.

Specify the type of SQL command for the variable to map in the *SQLType* input variable. For example, specify *_DBC_SQLTYPE_INSERT* to insert the values of the *DB Map Variable* to the table using a DB_Insert instruction.

Precautions for Correct Use

- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual* (Cat. No. W502) for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- This instruction cannot be used on an event task. A compiling error will occur.
- When the DB Connection Service was started in Test Mode, this instruction is completed normally without connecting to the DB actually.

- Refer to *1-2-1 DB Connection Service Specifications* on page 1-5 for the number of DB Map Variables for which you can create a mapping. However, even if the number of DB Map Variables has not reached the upper limit, an instruction error (Data Capacity Exceeded) will occur when any of the following condition is met.
 - a) When the total number of members of structures used as data type of DB Map Variables in all DB Connections exceeds 10,000 members.
- An error occurs for this instruction in the following cases. *Error* will be TRUE.
 - a) The instruction was executed when the DB Connection Service was not running.
 - b) The instruction was executed while the initialization processing of the DB Connection Service was in progress.
 - c) The instruction was executed while the DB Connection Service was stopped due to an error.
 - d) The instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
 - e) When the value of the *DBConnection* input variable is invalid or the specified DB Connection is already closed.
 - f) The *TableName* input variable is a text string consisting of NULL characters (16#00) only.
 - g) A space character is included in the text string specified for the *TableName* input variable.
 - h) When the data type of the variable specified in the *MapVar* input variable is not a structure.
 - i) A derivative data type is included as a member of the structure variable specified in the *MapVar* input variable.
 - j) The DB Map Variable specified in *MapVar* for INSERT and UPDATE is a structure array variable.
 - k) A value that is not defined as an enumerator was specified in the *SQLType* input variable.
 - l) The executed SQL statement resulted in an error in the DB.
 - m) The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
 - n) The maximum number of DB Map Variables for which a mapping can be created is exceeded.
 - o) More than 32 DB Connection Instructions were executed at the same time.
 - p) The DB Map Variable specified for BATCHINSERT is a structure variable

Sample Programming

Refer to *Sample Programming* on page 7-25 for the sample programming that is provided for the DB_Update instruction.

DB_Insert (Insert DB Record)

The DB_Insert instruction inserts values of a DB Map Variable to a table of the connected DB as a record.

Instruction	Name	FB/FUN	Graphic expression	ST expression
DB_Insert	Insert DB Record	FB		DB_Insert_instance (Execute, DBConnection, MapVar, TimeOut, Done, Busy, Error, ErrorID, SendStatus);

Note The DB_Insert_instance is an instance of DB_Insert instruction, which is declared as a variable.

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.
DBConnection	DB Connection	DWORD	16#00000000 to 16#FFFFFFF	---	16#00000000	Specify the DB connection established by a DB_Connect instruction.
MapVar	DB Map Variable	Structure	Depends on the data type.	---	---	Specify the DB Map Variable mapped by a DB_CreateMapping instruction.
TimeOut	Instruction Execution Timeout	TIME	T#0s, T#0.05s to T#180s	---	T#0s	Specify the time to detect the instruction execution timeout. When T#0s is specified, timeout is not monitored.

Output Variable

Name	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.
Error	Error	BOOL	TRUE or FALSE	---	TRUE when the instruction is terminated due to an error.

Name	Meaning	Data type	Valid range	Unit	Description
ErrorID	Error Code	WORD	16#0000 to 16#FFFF	---	Contains the error code when an error occurs.
SendStatus	Send Status	_eDBC_SEND_STATUS	Depends on the data type.	---	Outputs the progress of transmission of the SQL statement.

Related System-defined Variables

Name	Meaning	Data type	Description
_EIP_EtnOnlineSta	Online	BOOL	Status of the communications function of the built-in EtherNet/IP port. TRUE: Can be used. FALSE: Cannot be used.

Related Error Codes

Error code	Meaning	Description
0400 hex	Input Value Out of Range	The value of the <i>TimeOut</i> input variable is outside the valid range.
041D hex	Too Many Instructions Executed at the Same Time	More than 32 DB Connection Instructions were executed at the same time.
3000 hex	DB Connection Service not Started	The instruction was executed when the DB Connection Service was not running.
3002 hex	DB Connection Service Shutdown or Shutting Down	The instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
3008 hex	Invalid DB Connection	When the value of the <i>DBConnection</i> input variable is invalid or the specified DB Connection is already closed.
300A hex	DB Map Variable Unregistered	The variable specified in the <i>MapVar</i> input variable has not been mapped by a <i>DB_CreateMapping</i> instruction.
300B hex	SQL Execution Error	The executed SQL statement resulted in an error in the DB. The combination of data types is not listed in the table of data type correspondence between NJ/NX-series Controllers and database and the data type cannot be converted.
300C hex	Spool Capacity Exceeded	The SQL statement cannot be stored in the Spool memory because its capacity is exceeded.
3011 hex	DB Connection Disconnected Error Status	The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
3012 hex	DB Connection Instruction Execution Timeout	The instruction was not completed within the time specified for instruction execution timeout.
3013 hex	DB Connection Service Error Stop	The instruction was executed while the DB Connection Service was stopped due to an error.
3014 hex	Data Already Spooled	The SQL statement was spooled because one or more SQL statements are already stored in the Spool memory.
3015 hex	DB Connection Service Initializing	The instruction was executed while the initialization processing of the DB Connection Service was in progress.

Error code	Meaning	Description
3016 hex	DB in Process	The instruction was executed before completion of the DB's processing for the DB Connection Instruction Execution Timeout that occurred for the previous DB_Insert, DB_Update, DB_Select, or DB_Delete instruction.

Function

This instruction is used to insert the values of the DB Map Variable specified in the *MapVar* input variable to the table mapped by a DB_CreateMapping instruction as a record.

When the Spool function is enabled and the DB records cannot be updated due to a network failure or other causes, the SQL statement is stored in the Spool memory. In these cases, *_DBC_SEND_SPOOLED* is set in the *SendStatus* output variable and the instruction is terminated due to an error (DB Connection Disconnected Error Status).

When the Spool function is enabled and the DB records cannot be updated to the DB within the instruction execution timeout specified in the *TimeOut* input variable, the SQL statement is stored in the Spool memory. In these cases, *_DBC_SEND_SPOOLED* is set in the *SendStatus* output variable and the instruction is terminated due to an error (DB Connection Instruction Execution Timeout).

When the Spool function is enabled, the SQL statement is stored in the Spool memory if one or more SQL statements are already stored in the Spool memory. In these cases, *_DBC_SEND_SPOOLED* is set in the *SendStatus* output variable and the instruction is terminated due to an error (Data Already Spooled).

If an instruction error (SQL Execution Error) occurs when the Spool function is enabled, the transmitted SQL statement itself can be the cause of the SQL Execution Error. Therefore, the SQL statement is not stored in the Spool memory because the SQL Execution Error may occur again when the SQL statement is resent.

When the Spool capacity for each DB Connection is exceeded by spooling the SQL statement, this instruction is terminated due to an error (Spool Capacity Exceeded).

Precautions for Correct Use

- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual* (Cat. No. W502) for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- This instruction cannot be used on an event task. A compiling error will occur.
- If the values cannot be registered to the DB, for example, because the SQL statement is invalid, this instruction is terminated due to an error without storing the SQL statement into the Spool memory.
- When the DB Connection Service was started in Test Mode, this instruction is completed normally without executing the INSERT operation for the DB actually.
- When the error code is 300B hex (SQL Execution Error), you can get the detailed information of the SQL Execution Error by executing a DB_GetConnectionStatus instruction.

- The measurement error of instruction execution timeout is +50 ms for a 100-column record when the percentage of task execution time is 50% as a guide. However, the measurement error varies according to the percentage of task execution time and the number of columns.
- When two or more DB Connection Instructions are executed for a DB Connection at the same time, the DB Connection Service executes the instructions one by one. The measurement of instruction execution timeout for the second and later instructions is started when the instruction is executed by the DB Connection Service, not when the *Execute* input variable is changed to TRUE. Therefore, the time from when the *Execute* input variable is changed to TRUE to when the timeout occurs for the instruction is longer than the time set for the instruction execution timeout.
- If a value of a DB Map Variable is changed before the DB Connection Instruction is actually executed, the new value may be used when the DB Connection Instruction is executed. When changing a value of a DB Map Variable, write the user program so that the value is changed after confirming completion of the DB Connection Instruction.
- An error occurs for this instruction in the following cases. *Error* will be TRUE.
 - a) When the instruction was executed when the DB Connection Service was not running.
 - b) When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
 - c) When the instruction was executed while the DB Connection Service was stopped due to an error.
 - d) When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
 - e) When the value of the *DBConnection* input variable is invalid or the specified DB Connection is already closed.
 - f) The variable specified in the *MapVar* input variable has not been mapped by a *DB_CreateMap* instruction.
 - g) The value of the *TimeOut* input variable is outside the valid range.
 - h) The executed SQL statement resulted in an error in the DB.
 - i) The combination of data types is not listed in the table of data type correspondence between NJ/NX-series Controllers and database and the data type cannot be converted.
 - j) The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
 - k) When one or more SQL statements are already stored in the Spool memory.
 - l) The SQL statement cannot be stored in the Spool memory because its capacity is exceeded.
 - m) The instruction was not completed within the time specified for instruction execution timeout.
 - n) When the instruction was executed before completion of the DB's processing for the DB Connection Instruction Execution Timeout that occurred for the previous *DB_Insert*, *DB_Update*, *DB_Select*, or *DB_Delete* instruction.
 - o) When more than 32 DB Connection Instructions were executed at the same time.

Sample Programming

Refer to *Sample Programming* on page 7-25 for the sample programming that is provided for the *DB_Update* instruction.

DB_Update (Update DB Record)

The DB_Update (Update DB Record) instruction updates the values of a record of a table with the values of a DB Map Variable.

Instruction	Meaning	FB/FUN	Graphic expression	ST expression
DB_Update	Update DB Record	FB		DB_Update_instance (Execute, DBConnection, MapVar, Where, TimeOut, Done, Busy, Error, ErrorID, RecCnt, SendStatus);

Note The DB_Update_instance is an instance of DB_Update instruction, which is declared as a variable.

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.
DBConnection	DB Connection	DWORD	16#00000000 to 16#FFFFFFFF	---	16#00000000	Specify the DB connection established by a DB_Connect instruction.
MapVar	DB Map Variable	Structure	Depends on the data type.	---	---	Specify the DB Map Variable mapped by a DB_CreateMapping instruction.
Where	Retrieval Conditions	STRING	1986 bytes max. (including the final NULL character)*1	---	"	Specify a text string that expresses retrieval conditions (WHERE clause). ('WHERE' is not included.)
TimeOut	Instruction Execution Timeout	TIME	T#0s, T#0.05s to T#180s	---	T#0s	Specify the time to detect the instruction execution timeout. When T#0s is specified, timeout is not monitored.

- *1. When the database is case sensitive, specify the table name as shown below.
 When connecting to MySQL, enclose the table name in single-byte backquotes.
 Example: `ColumnA`
 When connecting to other databases, enclose the table name in single-byte double quotes.
 Example: "ColumnA"

Output Variable

Output variable	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.
Error	Error	BOOL	TRUE or FALSE	---	TRUE when the instruction is terminated due to an error.
ErrorID	Error Code	WORD	16#0000 to 16#FFFF	---	Contains the error code when an error occurs.
RecCnt	Number of Records	DINT	0 to 2147483647	---	Contains the number of records that were updated.
SendStatus	Send Status	_eDBC_SEND_STATUS	Depends on the data type.	---	Outputs the progress of transmission of the SQL statement.

Related System-defined Variables

Name	Meaning	Data type	Description
_EIP_EtnOnlineSta	Online	BOOL	Status of the communications function of the built-in EtherNet/IP port. TRUE: Can be used. FALSE: Cannot be used.

Related Error Codes

Error code	Meaning	Description
0400 hex	Input Value Out of Range	The value of the <i>TimeOut</i> input variable is outside the valid range.
041D hex	Too Many Instructions Executed at the Same Time	More than 32 DB Connection Instructions were executed at the same time.
3000 hex	DB Connection Service not Started	The instruction was executed when the DB Connection Service was not running.
3002 hex	DB Connection Service Shutdown or Shutting Down	The instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
3008 hex	Invalid DB Connection	When the value of the <i>DBConnection</i> input variable is invalid or the specified DB Connection is already closed.
300A hex	DB Map Variable Unregistered	The variable specified in the <i>MapVar</i> input variable has not been mapped by a <i>DB_CreateMapping</i> instruction.
300B hex	SQL Execution Error	The executed SQL statement resulted in an error in the DB. The combination of data types is not listed in the table of data type correspondence between NJ/NX-series Controllers and database and the data type cannot be converted.
300C hex	Spool Capacity Exceeded	The SQL statement cannot be stored in the Spool memory because its capacity is exceeded.
300E hex	Invalid Retrieval Conditions	The <i>Where</i> input variable is a text string consisting of NULL characters (16#00) only.

Error code	Meaning	Description
3011 hex	DB Connection Disconnected Error Status	The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
3012 hex	DB Connection Instruction Execution Timeout	The instruction was not completed within the time specified for instruction execution timeout.
3013 hex	DB Connection Service Error Stop	The instruction was executed while the DB Connection Service was stopped due to an error.
3014 hex	Data Already Spooled	The SQL statement was spooled because one or more SQL statements are already stored in the Spool memory.
3015 hex	DB Connection Service Initializing	The instruction was executed while the initialization processing of the DB Connection Service was in progress.
3016 hex	DB in Process	The instruction was executed before completion of the DB's processing for the DB Connection Instruction Execution Timeout that occurred for the previous DB_Insert, DB_Update, DB_Select, or DB_Delete instruction.

Function

This instruction is used to update the values of the records retrieved from the table mapped by a DB_CreateMapping instruction according to the retrieval conditions specified in the *Where* input variable (WHERE clause) with the values of a DB Map Variable specified in the *MapVar* input variable.

The records to update are retrieved according to the retrieval conditions specified in the *Where* input variable (WHERE clause). The *Where* input variable is expressed as a text string.

The text string in the *Where* input variable cannot consist of NULL characters (16#00) only. In that case, the instruction is terminated due to an error.

When using single quotes in the WHERE clause, use the escape character (\$').

Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for the escape character.

Refer to the manual of the database for the format of the WHERE clause.

Specify the retrieval conditions by the following values in the *Where* input variable.

Example 1: Update the values of the records where the value of a specific column is equal to or greater than the specified value.

Update the values of records where the value of *ColumnA* (unsigned integer) is 1234 or greater.

"ColumnA" >= 1234'

SQL statement to create: UPDATE TableProduct SET "ColumnA" =<Value>, "ColumnB" =<Value>

Where "ColumnA" >= 1234

Example 2: Update the values of the records where the value of a specific column starts with the specified text string.

Update the values of records where the value of *ColumnB* (text string) starts with 'ABC'.

"ColumnB" LIKE '\$ABC%\$'

SQL statement to create: UPDATE TableProduct SET "ColumnA" =<value>, "ColumnB" =<value>

Where "ColumnB" LIKE 'ABC%'

Example 3: Update the values of the records where the value of a specific column is equal to or greater than the value of the specified variable.

Update the values of records where the value of *ColumnA* (unsigned integer) is equal to or greater than the specified variable.

Specified value: `UINTVar := 1234;`

Input parameter in the *Where* clause: `WhereCond_Update := CONCAT("$ColumnA$ " >= ' ,
UINT_TO_STRING(UINTVar));`

SQL statement to create: `UPDATE TableProduct SET "ColumnA" =<Value>, "ColumnB" =<Value>
Where "ColumnA" >= 1234`

When the Spool function is enabled and the DB records cannot be updated due to a network failure or other causes, the SQL statement is stored in the Spool memory. In these cases, `_DBC_SEND_SPOOLED` is set in the *SendStatus* output variable and the instruction is terminated due to an error.

When the Spool function is enabled and the DB records cannot be updated within the instruction execution timeout specified in the *Timeout* input variable, the SQL statement is stored in the Spool memory. In these cases, `_DBC_SEND_SPOOLED` is set in the *SendStatus* output variable and the instruction is terminated due to an error (DB Connection Instruction Execution Timeout).

If an instruction error (SQL Execution Error) occurs when the Spool function is enabled, the transmitted SQL statement itself can be the cause of the SQL Execution Error, for example, due to a retrieval condition setting error. Therefore, the SQL statement is not stored in the Spool memory because the SQL Execution Error may occur again when the SQL statement is resent.

When the Spool capacity for each DB Connection is exceeded by spooling the SQL statement, this instruction is terminated due to an error (Spool Capacity Exceeded).

Precautions for Correct Use

- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual* (Cat. No. W502) for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- This instruction cannot be used on an event task. A compiling error will occur.
- This instruction cannot be executed without specifying the retrieval conditions.
- If the values cannot be registered to the DB, for example, because the SQL statement is invalid, this instruction is terminated due to an error without storing the SQL statement into the Spool memory.
- When the DB Connection Service was started in Test Mode, this instruction is completed normally without executing the UPDATE operation for the DB actually.
- When the error code is 300B hex (SQL Execution Error), you can get the detailed information of the SQL Execution Error by executing a `DB_GetConnectionStatus` instruction.
- The measurement error of instruction execution timeout is +50 ms for a 100-column record when the percentage of task execution time is 50% as a guide. However, the measurement error varies according to the percentage of task execution time and the number of columns.
- When two or more DB Connection Instructions are executed for a DB Connection at the same time, the DB Connection Service executes the instructions one by one. The measurement of instruction execution timeout for the second and later instructions is started when the instruction is executed by the DB Connection Service, not when the *Execute* input variable is changed to TRUE. Therefore,

the time from when the *Execute* input variable is changed to TRUE to when the timeout occurs for the instruction is longer than the time set for the instruction execution timeout.

- If a value of a DB Map Variable is changed before the DB Connection Instruction is actually executed, the new value may be used when the DB Connection Instruction is executed. When changing a value of a DB Map Variable, write the user program so that the value is changed after confirming completion of the DB Connection Instruction.
- An error occurs for this instruction in the following cases. *Error* will be TRUE.
 - a) When the instruction was executed when the DB Connection Service was not running.
 - b) When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
 - c) When the instruction was executed while the DB Connection Service was stopped due to an error.
 - d) When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
 - e) When the value of the *DBConnection* input variable is invalid or the specified DB Connection is already closed.
 - f) The variable specified in the *MapVar* input variable has not been mapped by a DB_CreateMapping instruction.
 - g) The *Where* input variable is a text string consisting of NULL characters (16#00) only.
 - h) The value of the *TimeOut* input variable is outside the valid range.
 - i) The executed SQL statement resulted in an error in the DB.
 - j) The combination of data types is not listed in the table of data type correspondence between NJ/NX-series Controllers and database and the data type cannot be converted.
 - k) The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
 - l) The SQL statement cannot be stored in the Spool memory because its capacity is exceeded.
 - m) When one or more SQL statements are already stored in the Spool memory.
 - n) The instruction was not completed within the time specified for instruction execution timeout.
 - o) When the instruction was executed before completion of the DB's processing for the DB Connection Instruction Execution Timeout that occurred for the previous DB_Insert, DB_Update, DB_Select, or DB_Delete instruction.
 - p) When more than 32 DB Connection Instructions were executed at the same time.

Sample Programming

This section gives sample programming for the following operations.

- Insert production data into a specified DB when the trigger variable changes to TRUE.
- Update production data in a specified DB when the trigger variable changes to TRUE.

DB Connection Settings and Data Type Definition

The minimum settings necessary for the sample programming are shown below.

● DB Connection Settings

DB Connection name: MyDatabase1

● Structure Data Type Definition

Name		Data type
PRODUCTION_INSERT		STRUCT
	Name	STRING[256]
	LotNo	STRING[32]
	Status	STRING[8]
	ProductionDate	DATE

Name		Data type
PRODUCTION_UPDATE		STRUCT
	Status	STRING[8]
	FinishTime	DATE_AND_TIME

Ladder Diagram

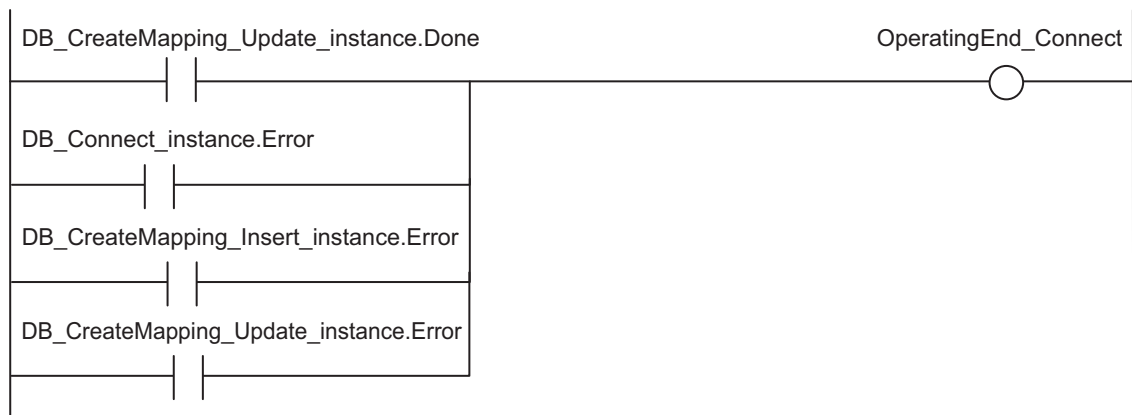
● Main Variables

Name	Data type	Default	Comment
_DBC_Status	_sDBC_STATUS	---	System-defined variable that shows the status of the DB Connection Service
DB_Connect_instance	DB_Connect	---	Instance of DB_Connect instruction
MyDB1	DWORD	---	This variable is assigned to the DBConnection output variable from DB_Connect_instance.
Trigger_Connect	BOOL	FALSE	Variable used as a trigger for establishing a DB Connection
RS_Connect_instance	RS	---	Instance of RS instruction
Operating_Connect	BOOL	FALSE	The DB_Connect instruction is executed when this variable is TRUE.
OperatingEnd_Connect	BOOL	FALSE	This variable changes to TRUE when the DB_Connect instruction is completed.
DB_CreateMapping_Insert_instance	DB_CreateMapping	---	Instance of DB_CreateMapping instruction
MapVar_Insert	PRODUCTION_INSERT		This variable is assigned to the MapVar input variable to DB_CreateMapping_Insert_instance.
DB_Insert_instance	DB_Insert	---	Instance of DB_Insert instruction
Name	STRING[256]	'WORK001'	Production information: Product name
LotNo	UINT	1234	Production information: Lot number
Trigger_Insert	BOOL	FALSE	Variable used as a trigger for inserting DB records
RS_Insert_instance	RS	---	Instance of RS instruction
Operating_Insert	BOOL	FALSE	The DB_Insert instruction is executed when this variable is TRUE.
OperatingEnd_Insert	BOOL	FALSE	This variable changes to TRUE when the DB_Insert instruction is completed.
DB_CreateMapping_Update_instance	DB_CreateMapping	---	Instance of DB_CreateMapping instruction
MapVar_Update	PRODUCTION_UPDATE		This variable is assigned to the MapVar input variable to DB_CreateMapping_Update_instance.

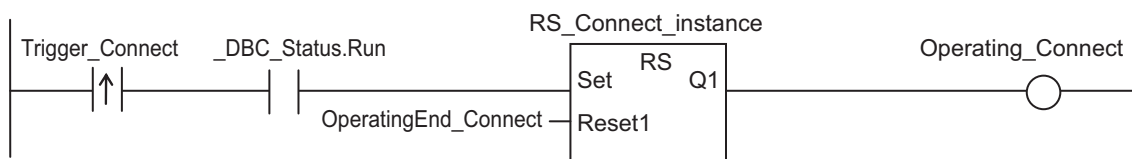
Name	Data type	Default	Comment
WhereCond	STRING[256]	---	This variable is assigned to the Where input variable to DB_CreateMapping_Update_instance.
DB_Update_instance	DB_Update	---	Instance of DB_Update instruction
Trigger_Update	BOOL	FALSE	Variable used as a trigger for updating DB records
RS_Update_instance	RS	---	Instance of RS instruction
Operating_Update	BOOL	FALSE	The DB_Update instruction is executed when this variable is TRUE.
OperatingEnd_Update	BOOL	FALSE	This variable changes to TRUE when the DB_Update instruction is completed.
DB_Close_instance	DB_Close	---	Instance of DB_Close instruction
Trigger_Close	BOOL	FALSE	Variable used as a trigger for closing the DB Connection
RS_Close_instance	RS	---	Instance of RS instruction
Operating_Close	BOOL	FALSE	The DB_Close instruction is executed when this variable is TRUE.
OperatingEnd_Close	BOOL	FALSE	This variable changes to TRUE when the DB_Close instruction is completed.

● Sample Programming

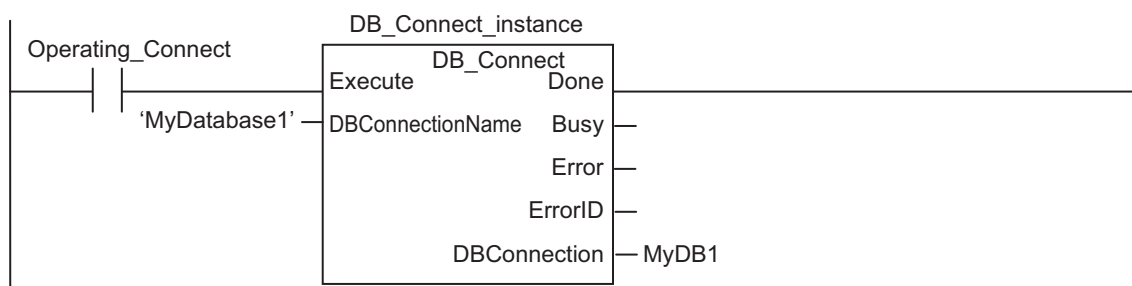
- Establish a DB Connection named *MyDatabase1* and map a table with a variable. Check the completion of DB_Connect and DB_CreateMapping instructions.



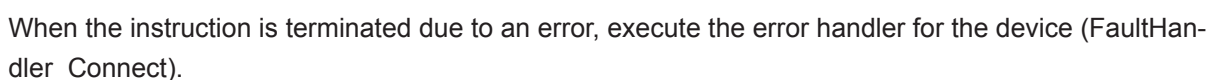
Accept the trigger for establishing the DB Connection.



Establish the DB Connection named *MyDatabase1*.



Map the variable MapVar_Insert to the table *Production* of the DB Connection *MyDB1* for the INSERT operation.



```

sequenceDiagram
    participant OC as Operating_Connect
    participant DCI as DB_Connect_instance
    participant FHC as FaultHandler_Connect

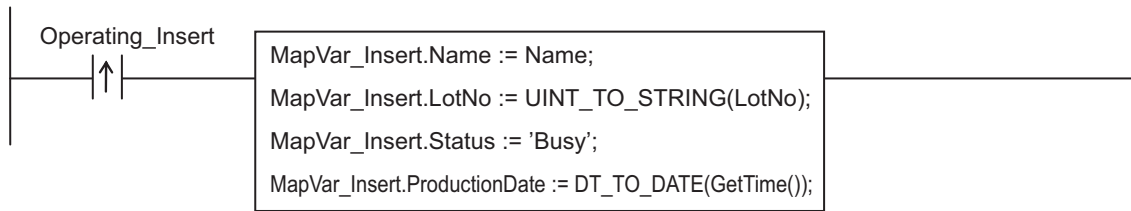
    OC->>DCI: Connect
    activate DCI
    DCI->>FHC: Connect
    activate FHC
    FHC-->>DCI: EN
    deactivate FHC
    DCI-->>OC: 
    deactivate DCI

    Note over DCI: DB_CreateMapping_Insert_instance
    DCI->>DCI: 
    activate DCI
    DCI-->>DCI: Error
    deactivate DCI

    Note over DCI: DB_CreateMapping_Update_instance
    DCI->>DCI: 
    activate DCI
    DCI-->>DCI: Error
    deactivate DCI
  
```

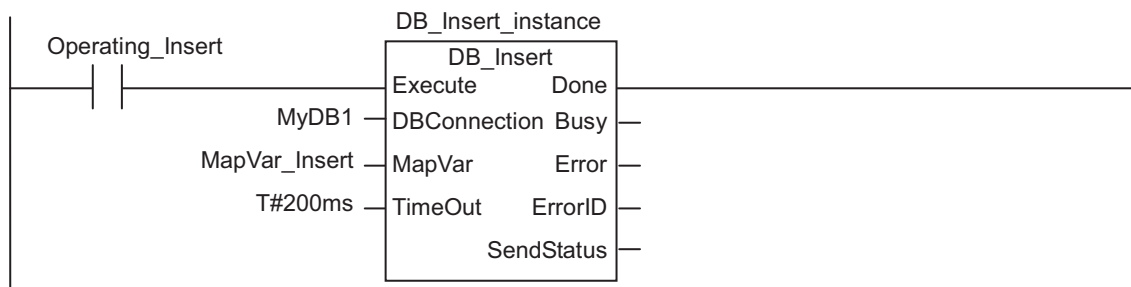
Check the completion of the DB_Insert instruction.





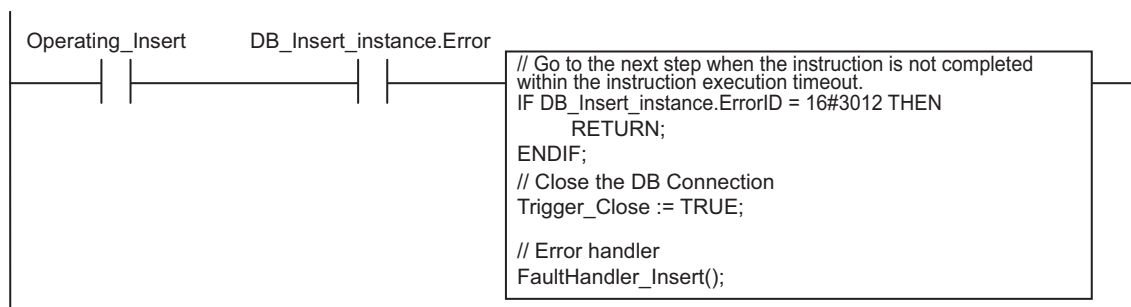
Insert production data to the DB Connection *MyDB1*.

Set the timeout for instruction execution to 200 ms.



When the instruction is terminated due to an error, execute the error handler for the device (FaultHandler_Insert).

Program the FaultHandler_Connect according to the device.

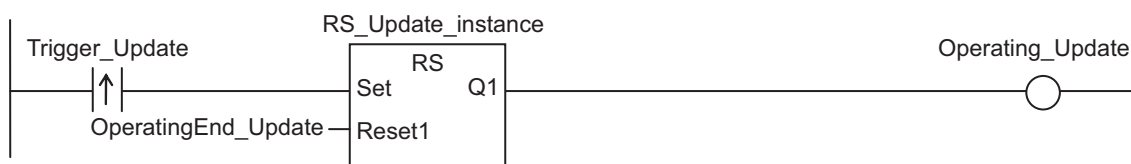


- Update the records in the DB Connection *MyDB1* when the variable *Trigger_Update* changes to TRUE.

Check the completion of the *DB_Update* instruction.

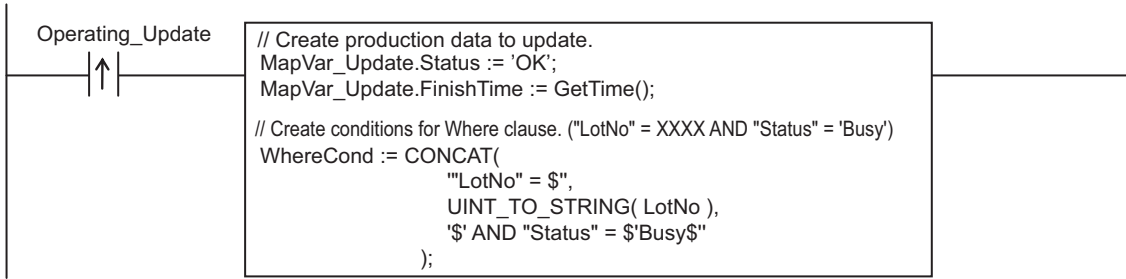


Accept the trigger for updating DB records.

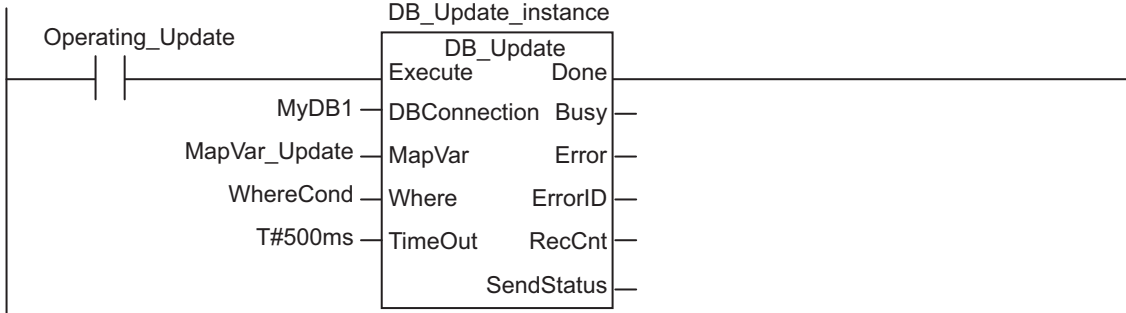


Create production data to update.

Create the conditions for Where clause.

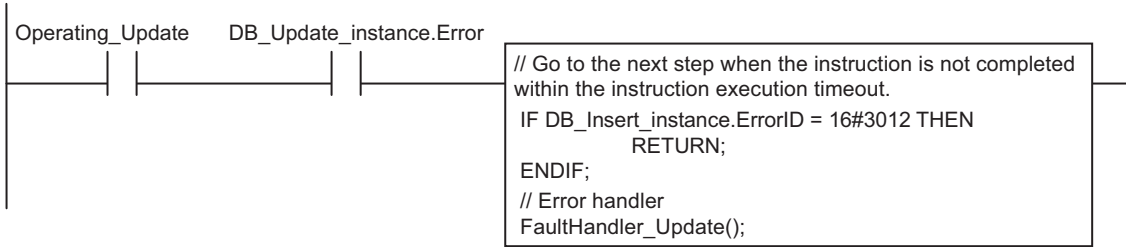


Update production data in the DB Connection *MyDB1*. Set the timeout for instruction execution to 500 ms.



When the instruction is terminated due to an error, execute the error handler for the device (FaultHandler_Update).

Program the FaultHandler_Update according to the device.



Close the DB Connection *MyDB1*.

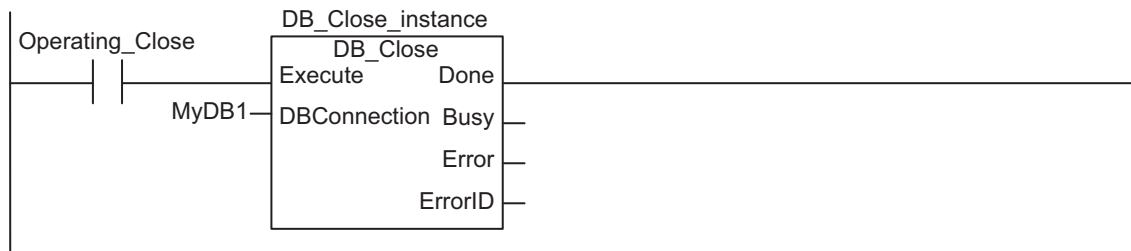
Check the completion of the DB_Close instruction.



Accept the trigger for closing the DB Connection.

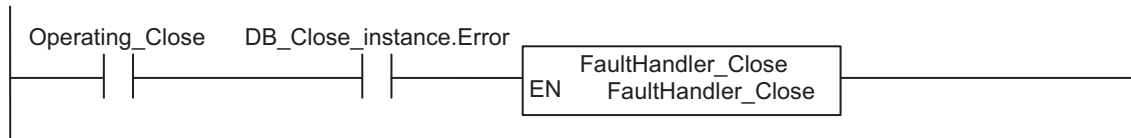


Close the DB Connection *MyDB1*.



When the instruction is terminated due to an error, execute the error handler for the device (FaultHandler_Close).

Program the FaultHandler_Connect according to the device.



Structured Text (ST)

● Main Variables

Name	Data type	Default	Comment
_DBC_Status	_sDBC_STATUS	---	System-defined variable that shows the status of the DB Connection Service
DB_Connect_instance	DB_Connect	---	Instance of DB_Connect instruction
MyDB1	DWORD	---	This variable is assigned to the DBConnection output variable from DB_Connect_instance.
Trigger_Connect	BOOL	FALSE	Variable used as a trigger for establishing a DB Connection
LastTrigger_Connect	BOOL	FALSE	Variable to retain the trigger status of the previous execution
Operating_Connect	BOOL	FALSE	The DB_Connect instruction is executed when this variable is TRUE.
OperatingStart_Connect	BOOL	FALSE	The start processing for establishing the DB Connection is executed when this variable is TRUE.
DB_CreateMapping_Insert_instance	DB_CreateMapping	---	Instance of DB_CreateMapping instruction
MapVar_Insert	PRODUCTION_INSERT		This variable is assigned to the MapVar input variable to DB_CreateMapping_Insert_instance.
DB_Insert_instance	DB_Insert	---	Instance of DB_Insert instruction
Name	STRING[256]	'WORK001'	Production information: Product name
LotNo	UINT	1234	Production information: Lot number
Trigger_Insert	BOOL	FALSE	Variable used as a trigger for inserting DB records
LastTrigger_Insert	BOOL	FALSE	Variable to retain the trigger status of the previous execution
Operating_Insert	BOOL	FALSE	The DB_Insert instruction is executed when this variable is TRUE.
OperatingStart_Insert	BOOL	FALSE	The start processing for inserting DB records is executed when this variable is TRUE.
DB_CreateMapping_Update_instance	DB_CreateMapping	---	Instance of DB_CreateMapping instruction
MapVar_Update	PRODUCTION_UPDATE	---	This variable is assigned to the MapVar input variable to DB_CreateMapping_Update_instance.
DB_Update_instance	DB_Update	---	Instance of DB_Update instruction
WhereCond	STRING[256]	---	This variable is assigned to the Where input variable to DB_CreateMapping_Update_instance.
Trigger_Update	BOOL	FALSE	Variable used as a trigger for updating DB records
LastTrigger_Update	BOOL	FALSE	Variable to retain the trigger status of the previous execution
Operating_Update	BOOL	FALSE	The DB_Update instruction is executed when this variable is TRUE.
OperatingStart_Update	BOOL	FALSE	The start processing for updating DB records is executed when this variable is TRUE.
DB_Close_instance	DB_Close	---	Instance of DB_Close instruction
Trigger_Close	BOOL	FALSE	Variable used as a trigger for closing the DB Connection
LastTrigger_Close	BOOL	FALSE	Variable to retain the trigger status of the previous execution
Operating_Close	BOOL	FALSE	The DB_Close instruction is executed when this variable is TRUE.
OperatingStart_Close	BOOL	FALSE	The start processing for closing the DB Connection is executed when this variable is TRUE.

Name	Data type	Default	Comment
Stage	INT	---	Variable that shows the status of the DB Connection

● Sample Programming

```
// - Establish a DB Connection named MyDatabase1 and map a table with a variable.

// Start the sequence when the variable Trigger_Connect changes to TRUE.
IF ( (Trigger_Connect=TRUE)
    AND (LastTrigger_Connect=FALSE)
    AND (_DBC_Status.Run=TRUE) ) THEN
    OperatingStart_Connect := TRUE;
    Operating_Connect := TRUE;
END_IF;
LastTrigger_Connect:=Trigger_Connect;

// Sequence start processing
IF (OperatingStart_Connect=TRUE) THEN
    // Initialize the instances of the applicable DB Connection Instructions.
    DB_Connect_instance( Execute:=FALSE );

    DB_CreateMapping_Insert_instance(
        Execute      := FALSE,
        MapVar       := MapVar_Insert,
        SQLType      := _DBC_SQLTYPE_INSERT
    );

    DB_CreateMapping_Update_instance(
        Execute      := FALSE,
        MapVar       := MapVar_Update,
        SQLType      := _DBC_SQLTYPE_UPDATE
    );

    Stage := INT#1;
    OperatingStart_Connect := FALSE;
END_IF;

// Establish the DB Connection named MyDatabase1
// Map the variable MapVar_Insert to the table Production of the DB Connection
MyDB1 for the INSERT operation.
// Map the variable MapVar_Update to the table Production of the DB Connection
MyDB1 for the UPDATE operation.
IF (Operating_Connect=TRUE) THEN
    CASE Stage OF
        1 : // Establish the DB Connection
            DB_Connect_instance(
                Execute          := TRUE,
                DBConnectionName := 'MyDatabase1',
```

```

        DBConnection      => MyDB1
    );

    IF (DB_Connect_instance.Done=TRUE) THEN
        Stage := INT#2; // Normal end
    END_IF;
    IF (DB_Connect_instance.Error=TRUE) THEN
        Stage := INT#99; // Error
    END_IF;

2 : // Map the DB table with the variable
    DB_CreateMapping_Insert_instance(
        Execute      := TRUE,
        DBConnection := MyDB1,
        TableName     := 'Production',
        MapVar        := MapVar_Insert,
        SQLType       := _DBC_SQLTYPE_INSERT
    );

    DB_CreateMapping_Update_instance(
        Execute      := TRUE,
        DBConnection := MyDB1,
        TableName     := 'Production',
        MapVar        := MapVar_Update,
        SQLType       := _DBC_SQLTYPE_UPDATE
    );

    IF ( (DB_CreateMapping_Insert_instance.Done=TRUE)
AND (DB_CreateMapping_Update_instance.Done=TRUE) ) THEN
        Operating_Connect:=FALSE; // Normal end
    END_IF;
    IF ( (DB_CreateMapping_Insert_instance.Error=TRUE)
OR (DB_CreateMapping_Update_instance.Error = TRUE) ) THEN
        Stage := INT#99; // Error
    END_IF;

99 :
    // Execute the error handler.
    // Program the error handler (FaultHandler_Connect) according to the device.

    FaultHandler_Connect();
    Operating_Connect := FALSE;
END_CASE;
END_IF;

// -----
// - Insert production data to DB Connection MyDB1 when the variable Trigger_

```

Insert changes to TRUE.

```
// Start the sequence when the variable Trigger_Insert changes to TRUE.
IF ( (Trigger_Insert=TRUE) AND (LastTrigger_Insert=FALSE) ) THEN
    OperatingStart_Insert := TRUE;
    Operating_Insert := TRUE;
END_IF;
LastTrigger_Insert := Trigger_Insert;

// Sequence start processing
IF (OperatingStart_Insert=TRUE) THEN
    // Initialize the instance of the applicable DB Connection Instruction.
    DB_Insert_instance( Execute:=FALSE, MapVar:=MapVar_Insert );

    // Create production data to insert.
    MapVar_Insert.Name           := Name;
    MapVar_Insert.LotNo          := UINT_TO_STRING(LotNo);
    MapVar_Insert.Status         := 'Busy';
    MapVar_Insert.ProductionDate := DT_TO_DATE(GetTime( ));

    OperatingStart_Insert := FALSE;
END_IF;

// Insert production data to the DB Connection MyDB1. Set the timeout for instruction execution to 200 ms.
IF (Operating_Insert=TRUE) THEN
    // Insert records
    DB_Insert_instance(
        Execute      := TRUE,
        DBConnection := MyDB1,
        MapVar       := MapVar_Insert,
        TimeOut      := T#200ms
    );

    IF (DB_Insert_instance.Done=TRUE) THEN
        Operating_Insert:=FALSE; // Normal end
    END_IF;
    IF (DB_Insert_instance.Error=TRUE) THEN
        // Go to the next step when the instruction is not completed within the instruction execution timeout
        IF (DB_Insert_instance.ErrorID = 16#3012) THEN
            Operating_Insert:=FALSE; // Normal end
        ELSE
            // Execute the error handler.
            // Program the error handler (FaultHandler_Insert) according to the device.
            FaultHandler_Insert();
        END_IF;
    END_IF;
END_IF;

```

```

        Operating_Insert := FALSE;
    END_IF;
END_IF;
END_IF;
// -----
// - Update the records in the DB Connection MyDB1 when the variable Trigger_Update changes to TRUE.

// Start the sequence when the variable Trigger_Update changes to TRUE.
IF ( (Trigger_Update=TRUE) AND (LastTrigger_Update=FALSE) ) THEN
    OperatingStart_Update := TRUE;
    Operating_Update := TRUE;
END_IF;
LastTrigger_Update := Trigger_Update;

// Sequence start processing
IF (OperatingStart_Update=TRUE) THEN
    // Initialize the instance of the applicable DB Connection Instruction.
    DB_Update_instance( Execute:=FALSE, MapVar:=MapVar_Update );

    // Create production data to update.
    MapVar_Update.Status := 'OK';
    MapVar_Update.FinishTime := GetTime();

    // Create the conditions for Where clause. ("LotNo" = XXXX AND "Status" = '
Busy')
    WhereCond := CONCAT(
        '"LotNo" = $',
        UINT_TO_STRING( LotNo ),
        '$' AND "Status" = '$Busy$'
    );

    OperatingStart_Update := FALSE;
END_IF;

// Update production data in the DB Connection MyDB1. Set the timeout for instruction execution to 200 ms.
IF (Operating_Update=TRUE) THEN
    // Update records
    DB_Update_instance(
        Execute      := TRUE,
        DBConnection := MyDB1,
        MapVar       := MapVar_Update,
        Where        := WhereCond,
        TimeOut      := T#200ms );

    IF (DB_Update_instance.Done=TRUE) THEN

```



```

        Operating_Update:=FALSE; // Normal end
    END_IF;
    IF (DB_Update_instance.Error=TRUE) THEN
        // Go to the next step when the instruction is not completed within the ins
        truction execution timeout.
        IF (DB_Update_instance.ErrorID = 16#3012) THEN
            Operating_Update:=FALSE; // Normal end
        ELSE
            // Execute the error handler.
            // Implement the error handler (FaultHandler_Update) according to the de
            vice.
            FaultHandler_Update();
            Operating_Update := FALSE;
        END_IF;
    END_IF;
END_IF;

// -----
// - Close the DB Connection "MyDB1".

// Start the sequence when the variable Trigger_Close changes to TRUE.
IF ( (Trigger_Close=TRUE) AND (LastTrigger_Close=FALSE) ) THEN
    OperatingStart_Close := TRUE;
    Operating_Close := TRUE;
END_IF;
LastTrigger_Close := Trigger_Close;

// Sequence start processing
IF (OperatingStart_Close=TRUE) THEN
    // Initialize the instance of the applicable DB Connection Instruction.
    DB_Close_instance( Execute:=FALSE );

    OperatingStart_Close := FALSE;
END_IF;

// Close the DB Connection "MyDB1".
IF (Operating_Close=TRUE) THEN
    // Close the DB Connection.
    DB_Close_instance( Execute:=TRUE, DBConnection:=MyDB1 );

    IF (DB_Close_instance.Done=TRUE) THEN
        Operating_Close := FALSE; // Normal end
    END_IF;
    IF (DB_Close_instance.Error=TRUE) THEN
        // Execute the error handler.
        // Program the error handler (FaultHandler_Close) according to the device.
        FaultHandler_Close();
    
```

```
        Operating_Close := FALSE;  
    END_IF;  
END_IF;
```

DB_Select (Retrieve DB Record)

The DB_Select instruction retrieves records from a table to a DB Map Variable.

Instruction	Name	FB/FUN	Graphic expression	ST expression
DB_Select	Retrieve DB Record	FB		DB_Select_instance (Execute, DBConnection, Where, Sort, TimeOut, MapVar, Done, Busy, Error, ErrorID, RecCnt, SelectedCnt);

Note The DB_Select_instance is an instance of DB_Select instruction, which is declared as a variable.

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.
DBConnection	DB Connection	DWORD	16#00000000 to 16#FFFFFFFF	---	16#00000000	Specify the DB connection established by a DB_Connect instruction.
Where	Retrieval Conditions	STRING	1986 bytes max. (including the final NULL character)*1	---	"	Specify a text string that expresses retrieval conditions (WHERE clause). ('WHERE' is not included.) In the DB Connection Service version 2.00 and higher, skipping the input for this variable does not cause the instruction to end abnormally.
Sort	Sort Conditions	STRING	1986 bytes max. (including the final NULL character)*1	---	"	Specify a text string that expresses sort conditions (ORDER BY clause). ('ORDER BY' is not included.)

Name	Meaning	Data type	Valid range	Unit	Default	Description
TimeOut	Instruction Execution Timeout	TIME	T#0s, T#0.05s to T#180s	---	T#0s	Specify the time to detect the instruction execution timeout. When T#0s is specified, timeout is not monitored.

- *1. When the database is case sensitive, specify the table name as shown below.
 When connecting to MySQL, enclose the table name in single-byte backquotes.
 Example: `ColumnA`
 When connecting to other databases, enclose the table name in single-byte double quotes.
 Example: "ColumnA"

In-out Variables

Name	Meaning	Data type	Valid range	Unit	Description
MapVar	DB Map Variable	Structure, Structure array (entire array)	Depends on the data type.	---	Specify the DB Map Variable mapped by a DB_CreateMapping instruction.

Output Variable

Name	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.
Error	Error	BOOL	TRUE or FALSE	---	TRUE when the instruction is terminated due to an error.
ErrorID	Error Code	WORD	16#0000 to 16#FFFF	---	Contains the error code when an error occurs.
RecCnt	Number of Records	DINT	0 to 65535	---	Contains the number of records that were retrieved to the DB Map Variable.
SelectedCnt	Number of Retrieved Records	DINT	0 to 2147483647	---	Total number of records retrieved according to the retrieval conditions.

Related System-defined Variables

Name	Meaning	Data type	Description
_EIP_EtnOnlineSta	Online	BOOL	Status of the communications function of the built-in EtherNet/IP port. TRUE: Can be used. FALSE: Cannot be used.

Related Error Codes

Error code	Meaning	Description
0400 hex	Input Value Out of Range	The value of the <i>TimeOut</i> input variable is outside the valid range.

Error code	Meaning	Description
041D hex	Too Many Instructions Executed at the Same Time	More than 32 DB Connection Instructions were executed at the same time.
3000 hex	DB Connection Service not Started	The instruction was executed when the DB Connection Service was not running.
3002 hex	DB Connection Service Shutdown or Shutting Down	The instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
3008 hex	Invalid DB Connection	When the value of the <i>DBConnection</i> input variable is invalid or the specified DB Connection is already closed.
300A hex	DB Map Variable Unregistered	The variable specified in the <i>MapVar</i> input variable has not been mapped by a <i>DB_CreateMapping</i> instruction.
300B hex	SQL Execution Error	The executed SQL statement resulted in an error in the DB. The retrieved record contains a column whose value is NULL. The combination of data types is not listed in the table of data type correspondence between NJ/NX-series Controllers and database and the data type cannot be converted.
300E hex	Invalid Retrieval Conditions	The <i>Where</i> input variable is a text string consisting of NULL characters (16#00) only.
3011 hex	DB Connection Disconnected Error Status	The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
3012 hex	DB Connection Instruction Execution Timeout	The instruction was not completed within the time specified for instruction execution timeout.
3013 hex	DB Connection Service Error Stop	The instruction was executed while the DB Connection Service was stopped due to an error.
3014 hex	Data Already Spooled	This instruction cannot be executed because one or more SQL statements are already stored in the Spool memory.
3015 hex	DB Connection Service Initializing	The instruction was executed while the initialization processing of the DB Connection Service was in progress.
3016 hex	DB in Process	The instruction was executed before completion of the DB's processing for the DB Connection Instruction Execution Timeout that occurred for the previous <i>DB_Insert</i> , <i>DB_Update</i> , <i>DB_Select</i> , or <i>DB_Delete</i> instruction.

Function

This instruction is used to retrieve records from a table mapped by a *DB_CreateMapping* instruction into the DB Map Variable specified in the *MapVar* in-out variable.

Define the DB Map Variable as an array when you want to retrieve more than one record.

The number of records retrieved to the DB Map Variable is output to the *RecCnt* output variable. The number of records retrieved according to the retrieval conditions is output to the *SelectedCnt* output variable.

The relationship between the number of array elements in the DB Map Variable and the number of records in the *RecCnt* and *SelectedCnt* output variables is described below.

[When the number of array elements of the DB Map Variable is equal to or smaller than ($\leq$) the number of retrieved records]

The records up to the maximum number of elements in the DB Map Variable are output.

For example, in the case where 30 records are retrieved for the DB Map Variable with 10 array elements, the records from MapVar[0] to MapVar[9] are retrieved. The value of *RecCnt* will be 10 and the value of *SelectedCnt* will be 30 in this case.

[When the number of array elements of the DB Map Variable is bigger than (>) the number of retrieved records]

The records up to the number of elements of the retrieved records are output. For the later elements, the records are not retrieved, but the previous values are retained.

For example, in the case where 3 records are retrieved for the DB Map Variable with 10 array elements, the records from MapVar[0] to MapVar[2] are retrieved. The values of MapVar[3] to MapVar[9] do not change. The value of *RecCnt* will be 3 and the value of *SelectedCnt* will be 3 in this case.

The records are retrieved according to the retrieval conditions specified in the *Where* input variable (WHERE clause). The *Where* input variable is expressed as a text string.

The text string in the *Where* input variable cannot consist of NULL characters (16#00) only. In that case, the instruction is terminated due to an error.

Specify the sort conditions in the *Sort* input variable (ORDER BY clause) to sort out the retrieved records. The *Sort* input variable is expressed as a text string.

When the sort conditions are specified, the records are contained in the DB Map Variable in the order specified by the sort conditions.

When the sort conditions are not specified, the output order to the DB Map Variable depends on the specifications of the DB type to connect.

When using single quotes in the WHERE and SORT clauses, use the escape character (\$').

Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for the escape character.

Refer to the manual of the database for the format of the WHERE and SORT clauses.

Specify the retrieval conditions by the following values in the *Where* input variable.

Example 1: Retrieve the values of the records where the value of a specific column is equal to or greater than the specified value.

Update the values of records where the value of *ColumnA* (unsigned integer) is 1234 or greater.

"ColumnA" >= 1234'

SQL statement to create: SELECT FROM TableProduct Where "ColumnA" = 1234

Example 2: Retrieve the records where the values of specific two columns are within the specified range.

Retrieve the records where the value of *ColumnA* (unsigned integer) is bigger than 1000 and the value of *ColumnB* (unsigned integer) is smaller than 2000.

"ColumnA" > 1000 AND "ColumnB" < 2000'

SQL statement to create: SELECT FROM TableProduct Where "ColumnA" > 1000 AND "ColumnB" < 2000

Example 3: Retrieve the values of the records where the value of a specific column is equal to or greater than the value of the specified variable.

Retrieve the values of records where the value of *ColumnA* (unsigned integer) is equal to or greater than the specified variable.

Specified value: UINTVar := 1234;

Input parameter in the *Where* clause: WhereCond_Select := CONCAT('\$"ColumnA\$" >= ',
UINT_TO_STRING(UINTVar));

SQL statement to create: SELECT FROM TableProduct Where "ColumnA" = 1234

Specify the sort conditions in the *Sort* input variable by the following values.

Example: Retrieve the records sorted by the values of two columns.

Sort the values of *ColumnA* in ascending order and values of *ColumnB* in descending order.

"ColumnA" ASC, "ColumnB" DESC'

SQL statement to create: SELECT FROM TableProduct ORDER BY "ColumnA" ASC, "ColumnB" DESC

Precautions for Correct Use

- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual* (Cat. No. W502) for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- This instruction cannot be used on an event task. A compiling error will occur.
- This instruction cannot be executed without specifying the retrieval conditions.
- When no record is retrieved as the execution result of this instruction, the values of the *RecCnt* and *SelectedCnt* output variables are both 0 and the instruction is normally completed.
- Even if the number of array elements of the DB Map Variable does not match the number of retrieved records as the execution result of this instruction, the instruction is also normally completed.
- When the DB Connection Service was started in Test Mode, this instruction is normally ended without executing the SELECT operation for the DB actually. No values are stored in the DB Map Variable specified in the *MapVar* in-out variable and 0 is output to both the *RecCnt* and *SelectedCnt* output variables.
- Even if the specified number of bytes in STRING data is shorter than the table data, this instruction is normally ended.

Example: When 12 characters are contained in a table column and data type of the corresponding member of the DB Map Variable is STRING[11], this instruction can retrieve only up to 11 characters, but will be normally ended.

- When the error code is 300B hex (SQL Execution Error), you can get the detailed information of the SQL Execution Error by executing a *DB_GetConnectionStatus* instruction.
- The measurement error of instruction execution timeout is +50 ms for a 100-column record when the percentage of task execution time is 50% as a guide. However, the measurement error varies according to the percentage of task execution time and the number of columns.
- When two or more DB Connection Instructions are executed for a DB Connection at the same time, the DB Connection Service executes the instructions one by one. The measurement of instruction execution timeout for the second and later instructions is started when the instruction is executed by the DB Connection Service, not when the *Execute* input variable is changed to TRUE. Therefore, the time from when the *Execute* input variable is changed to TRUE to when the timeout occurs for the instruction is longer than the time set for the instruction execution timeout.
- An error occurs for this instruction in the following cases. *Error* will be TRUE.
 - a) When the instruction was executed when the DB Connection Service was not running.
 - b) When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
 - c) When the instruction was executed while the DB Connection Service was stopped due to an error.
 - d) When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
 - e) When the value of the *DBConnection* input variable is invalid or the specified DB Connection is already closed.

- f) The value of the *TimeOut* input variable is outside the valid range.
- g) The variable specified in the *MapVar* input variable has not been mapped by a *DB_CreateMap*-ping instruction.
- h) The executed SQL statement resulted in an error in the DB.
- i) When the data types cannot be converted between NJ/NX-series Controllers and database
- j) The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
- k) When one or more SQL statements are already stored in the Spool memory.
- l) The instruction was not completed within the time specified for instruction execution timeout.
- m) When the instruction was executed before completion of the DB's processing for the DB Connection Instruction Execution Timeout that occurred for the previous *DB_Insert*, *DB_Update*, *DB_Select*, or *DB_Delete* instruction.
- n) When more than 32 DB Connection Instructions were executed at the same time.

Sample Programming

Refer to *Sample Programming* on page 7-48 for the sample programming that is provided for the *DB_Delete* instruction.

DB_Delete (Delete DB Record)

The DB_Delete instruction deletes the records that match the conditions from a specified table.

Instruction	Name	FB/FUN	Graphic expression	ST expression
DB_Delete	Delete DB Record	FB		DB_Delete_instance (Execute, DBConnection, TableName, Where, TimeOut, Done, Busy, Error, Error-ID, RecCnt);

Note The DB_Delete_instance is an instance of DB_Delete instruction, which is declared as a variable.

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.
DBCon- nection	DB Con- nection	DWORD	16#00000000 to 16#FFFFFFF	---	16#00000000	Specify the DB connection established by a DB_Connect instruction.
Table- Name	Table Name	STRING	Depends on the data type.	---	"	Specify a table name in the DB.
Where	Retrieval Conditions	STRING	1,986 bytes max. (including the final NULL character)*1	---	"	Specify a text string that expresses retrieval conditions (WHERE clause). ('WHERE' is not included.)
TimeOut	Instruction Execution Timeout	TIME	T#0s, T#0.05s to T#180s	---	T#0s	Specify the time to detect the instruction execution timeout. When T#0s is specified, timeout is not monitored.

- *1. When the database is case sensitive, specify the column name as shown below.
 When connecting to MySQL, enclose the table name in single-byte backquotes.
 Example: `ColumnA`
 When connecting to other databases, enclose the table name in single-byte double quotes.
 Example: "ColumnA"

Output Variable

Name	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.
Error	Error	BOOL	TRUE or FALSE	---	TRUE when the instruction is terminated due to an error.
ErrorID	Error Code	WORD	16#0000 to 16#FFFF	---	Contains the error code when an error occurs.
RecCnt	Number of Records	DINT	0 to 2147483647	---	Contains the number of records that were deleted.

Related System-defined Variables

Name	Meaning	Data type	Description
_EIP_EtnOnlineSta	Online	BOOL	Status of the communications function of the built-in EtherNet/IP port. TRUE: Can be used. FALSE: Cannot be used.

Related Error Codes

Error code	Meaning	Description
0400 hex	Input Value Out of Range	The value of the <i>TimeOut</i> input variable is outside the valid range.
0406 hex	Illegal Data Position Specified	The <i>TableName</i> input variable is a text string consisting of NULL characters (16#00) only.
0410 hex	Text String Format Error	A space character is included in the text string specified for the <i>TableName</i> input variable.
041D hex	Too Many Instructions Executed at the Same Time	More than 32 DB Connection Instructions were executed at the same time.
3000 hex	DB Connection Service not Started	The instruction was executed when the DB Connection Service was not running.
3002 hex	DB Connection Service Shutdown or Shutting Down	The instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
3008 hex	Invalid DB Connection	When the value of the <i>DBConnection</i> input variable is invalid or the specified DB Connection is already closed.
300B hex	SQL Execution Error	The executed SQL statement resulted in an error in the DB.
300E hex	Invalid Retrieval Conditions	The <i>Where</i> input variable is a text string consisting of NULL characters (16#00) only.
3011 hex	DB Connection Disconnected Error Status	The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
3012 hex	DB Connection Instruction Execution Timeout	The instruction was not completed within the time specified for instruction execution timeout.
3013 hex	DB Connection Service Error Stop	The instruction was executed while the DB Connection Service was stopped due to an error.
3014 hex	Data Already Spooled	This instruction cannot be executed because one or more SQL statements are already stored in the Spool memory.

Error code	Meaning	Description
3015 hex	DB Connection Service Initializing	The instruction was executed while the initialization processing of the DB Connection Service was in progress.
3016 hex	DB in Process	The instruction was executed before completion of the DB's processing for the DB Connection Instruction Execution Timeout that occurred for the previous DB_Insert, DB_Update, DB_Select, or DB_Delete instruction.

Function

This instruction is used to delete the records that match the conditions specified in the *Where* input variable from the table specified in the *TableName* input variable.

The records to delete are retrieved according to the retrieval conditions specified in the *Where* input variable (WHERE clause). The *Where* input variable is expressed as a text string.

The text string in the *Where* input variable cannot consist of NULL characters (16#00) only. In that case, the instruction is terminated due to an error.

When using single quotes in the WHERE clause, use the escape character (\$').

Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for the escape character.

Refer to the manual of the database for the format of the WHERE clause.

Specify the retrieval conditions in the *Where* input variable by the following values.

Example: Delete the records where either of the values of the specified two columns is equal to the specified value.

Delete the records where the value of *ColumnA* (unsigned integer) is equal to 1000 or the value of *ColumnB* (unsigned integer) is equal to 2000

"ColumnA" = 1000 OR "ColumnB" = 2000'

SQL statement to create: DELETE FROM TableProduct Where "ColumnA" = 1000 OR "ColumnB" = 2000

Precautions for Correct Use

- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual (Cat. No. W502)* for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- This instruction cannot be used on an event task. A compiling error will occur.
- This instruction cannot be executed without specifying the retrieval conditions.
- When the DB Connection Service was started in Test Mode, this instruction is normally ended without executing the DELETE operation for the DB actually.
- When the error code is 300B hex (SQL Execution Error), you can get the detailed information of the SQL Execution Error by executing a DB_GetConnectionStatus instruction.
- The measurement error of instruction execution timeout is +50 ms for a 100-column record when the percentage of task execution time is 50% as a guide. However, the measurement error varies according to the percentage of task execution time and the number of columns.
- When two or more DB Connection Instructions are executed for a DB Connection at the same time, the DB Connection Service executes the instructions one by one. The measurement of instruction

execution timeout for the second and later instructions is started when the instruction is executed by the DB Connection Service, not when the *Execute* input variable is changed to TRUE. Therefore, the time from when the *Execute* input variable is changed to TRUE to when the timeout occurs for the instruction is longer than the time set for the instruction execution timeout.

- An error occurs for this instruction in the following cases. *Error* will be TRUE.
 - a) When the instruction was executed when the DB Connection Service was not running.
 - b) When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
 - c) When the instruction was executed while the DB Connection Service was stopped due to an error.
 - d) When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
 - e) When the value of the *DBConnection* input variable is invalid or the specified DB Connection is already closed.
 - f) The *TableName* input variable is a text string consisting of NULL characters (16#00) only.
 - g) A space character is included in the text string specified for the *TableName* input variable.
 - h) The *Where* input variable is a text string consisting of NULL characters (16#00) only.
 - i) The value of the *TimeOut* input variable is outside the valid range.
 - j) The executed SQL statement resulted in an error in the DB.
 - k) The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
 - l) When one or more SQL statements are already stored in the Spool memory.
 - m) The instruction was not completed within the time specified for instruction execution timeout.
 - n) When the instruction was executed before completion of the DB's processing for the DB Connection Instruction Execution Timeout that occurred for the previous DB_Insert, DB_Update, DB_Select, or DB_Delete instruction.
 - o) When more than 32 DB Connection Instructions were executed at the same time.

Sample Programming

This section gives sample programming of the following operations for Oracle database.

- Retrieve production data for the specified lot number from a DB table when the trigger variable changes to TRUE.
- Delete the records other than the latest one if more than one record was retrieved.

DB Connection Settings and Data Type Definition

The minimum settings necessary for the sample programming are shown below.

● DB Connection Settings

DB Connection name: MyDatabase1

● Structure Data Type Definition

Name	Data type
PRODUCTION_SELECT	STRUCT

Name	STRING[256]
LotNo	STRING[32]
Status	STRING[8]
ProductionDate	DATE
FinishTime	DATE_AND_TIME

Ladder Diagram

● Main Variables

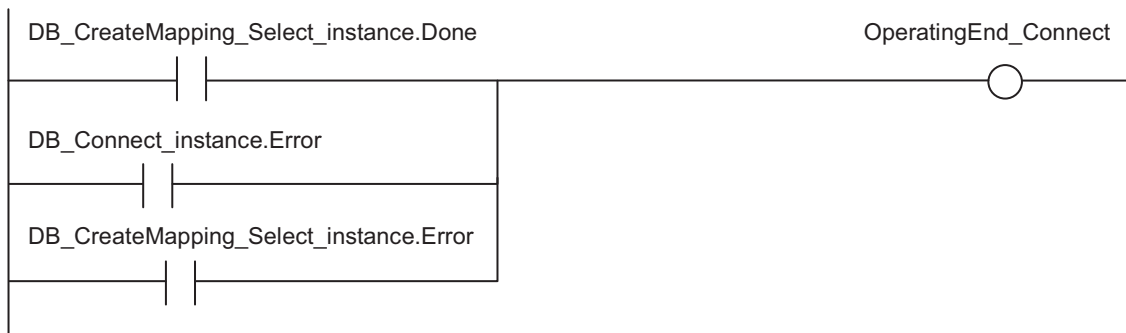
Name	Data type	Default	Comment
_DBC_Status	_sDBC_STATUS	---	System-defined variable that shows the status of the DB Connection Service
DB_Connect_instance	DB_Connect	---	Instance of DB_Connect instruction
MyDB1	DWORD	---	This variable is assigned to the DBConnection output variable from DB_Connect_instance.
LotNo	UINT	1234	Variable to specify the lot number for retrieving/deleting DB records
Trigger_Connect	BOOL	FALSE	Variable used as a trigger for establishing a DB Connection
RS_Connect_instance	RS	---	Instance of RS instruction
Operating_Connect	BOOL	FALSE	The DB_Connect instruction is executed when this variable is TRUE.
OperatingEnd_Connect	BOOL	FALSE	This variable changes to TRUE when the DB_Connect instruction is completed.
DB_CreateMapping_Select_instance	DB_CreateMapping	---	Instance of DB_CreateMapping instruction
MapVar_Select	ARRAY[0..9] OF PRODUCTION_SELECT		This variable is assigned to the MapVar input variable to DB_CreateMapping_Select_instance.
WhereCond_Select	STRING[256]	---	This variable is assigned to the Where input variable to DB_Select_instance.
SortCond_Select	STRING[256]	---	This variable is assigned to the Sort input variable to DB_Select_instance.
DB_Select_instance	DB_Select	---	Instance of DB_Select instruction
Trigger_Select	BOOL	FALSE	Variable used as a trigger for retrieving DB records
RS_Select_instance	RS	---	Instance of RS instruction
Operating_Select	BOOL	FALSE	The DB_Select instruction is executed when this variable is TRUE.
OperatingEnd_Select	BOOL	FALSE	This variable changes to TRUE when the DB_Select instruction is completed.
WhereCond_Delete	STRING[256]	---	This variable is assigned to the Where input variable to DB_Delete_instance.
Request_Delete	BOOL	FALSE	The DB_Delete instruction is executed when this variable is TRUE.
DB_Delete_instance	DB_Delete	---	Instance of DB_Delete instruction
RS_Delete_instance	RS	---	Instance of RS instruction
Operating_Delete	BOOL	FALSE	The DB_Delete instruction is executed when this variable is TRUE.

Name	Data type	Default	Comment
OperatingEnd_Delete	BOOL	FALSE	This variable changes to TRUE when the DB_Delete instruction is completed.
DB_Close_instance	DB_Close	---	Instance of DB_Close instruction
Trigger_Close	BOOL	FALSE	Variable used as a trigger for closing the DB Connection
RS_Close_instance	RS	---	Instance of RS instruction
Operating_Close	BOOL	FALSE	The DB_Close instruction is executed when this variable is TRUE.
OperatingEnd_Close	BOOL	FALSE	This variable changes to TRUE when the DB_Close instruction is completed.

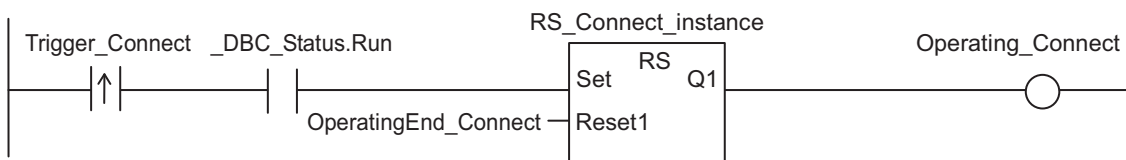
● Sample Programming

- Establish a DB Connection named *MyDatabase1* and map a table with a variable.

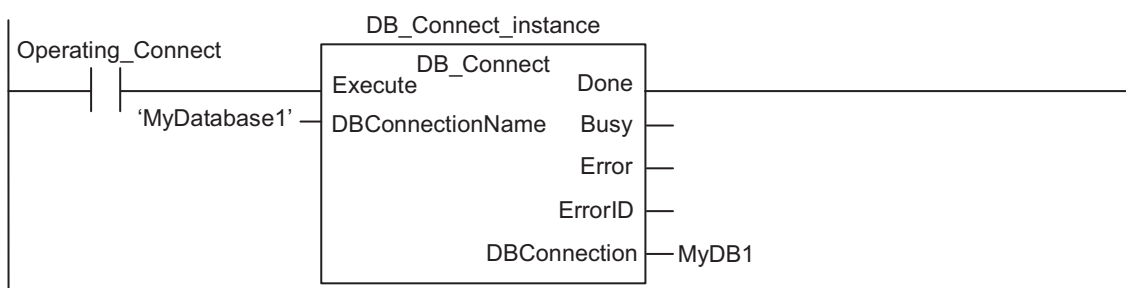
Check the completion of DB_Connect and DB_CreateMapping instructions.



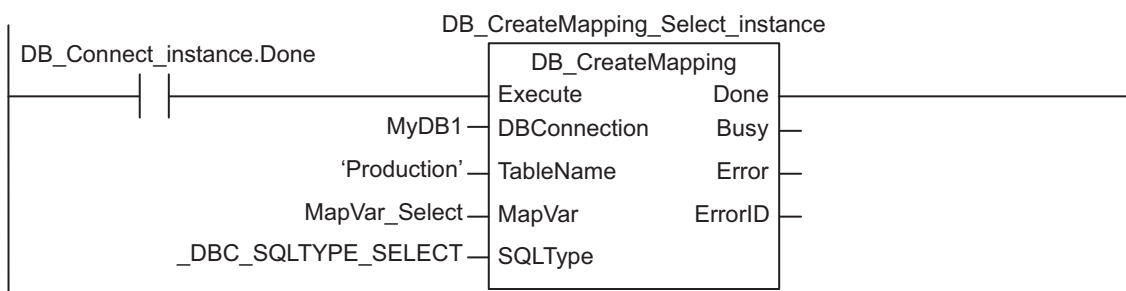
Accept the trigger for establishing the DB Connection.



Establish the DB Connection named *MyDataBase1*.

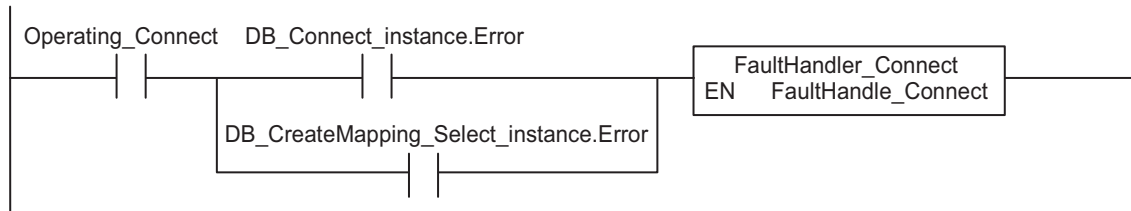


Map the variable `MapVar_Select` to the table *Production* of the DB Connection *MyDB1* for the SELECT operation.



When the instruction is terminated due to an error, execute the error handler for the device (FaultHandler_Connect).

Program the FaultHandler_Connect according to the device.

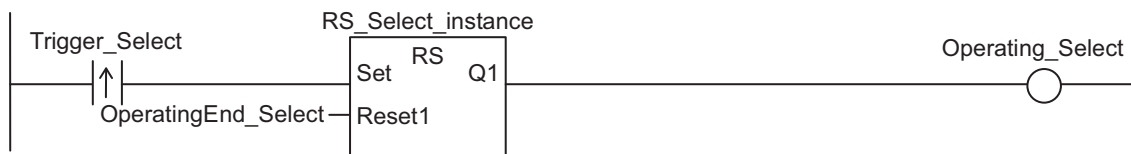


- Retrieve records for the specified lot number from the DB Connection *MyDB1* when the variable Trigger_Select changes to TRUE.

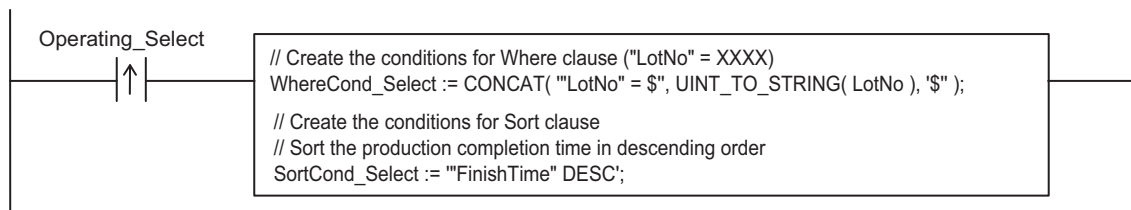
Check the completion of the DB_Select instruction.



Accept the trigger for retrieving DB records.

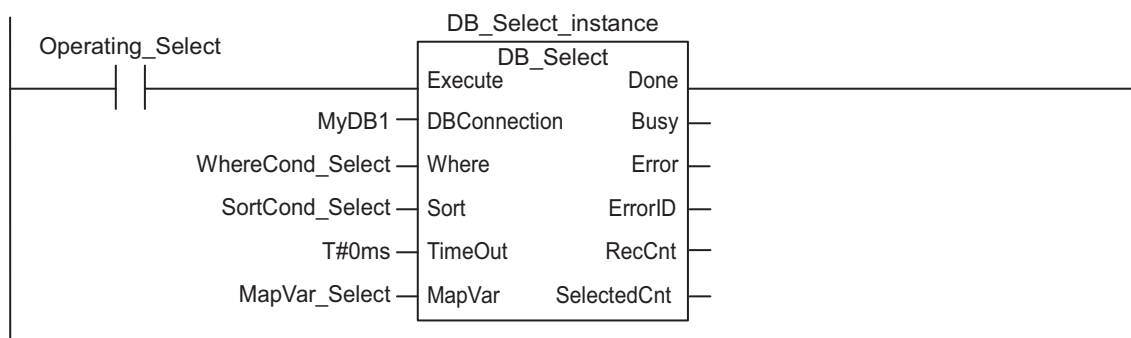


Create the conditions for the Where and Sort clauses.



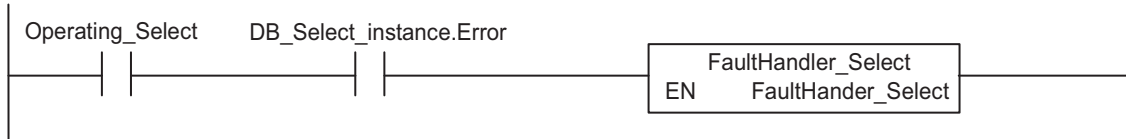
Retrieve the records from the DB Connection *MyDB1*.

Timeout is not monitored for the instruction execution.

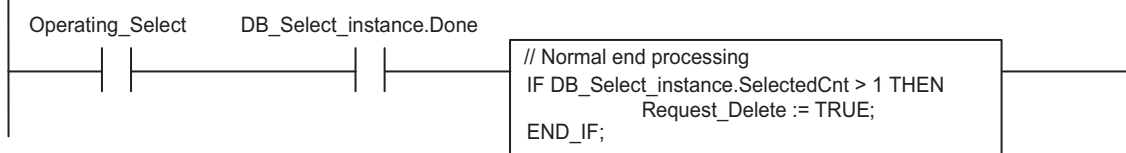


When the instruction is terminated due to an error, execute the error handler for the device (FaultHandler_Select).

Program the FaultHandler_Select according to the device.



If two or more records were retrieved, delete the records other than the latest one.

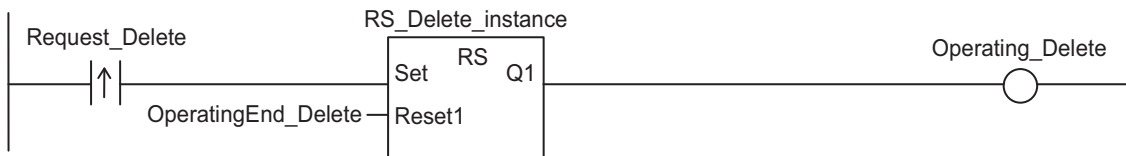


- Delete the records other than the latest one from the DB table

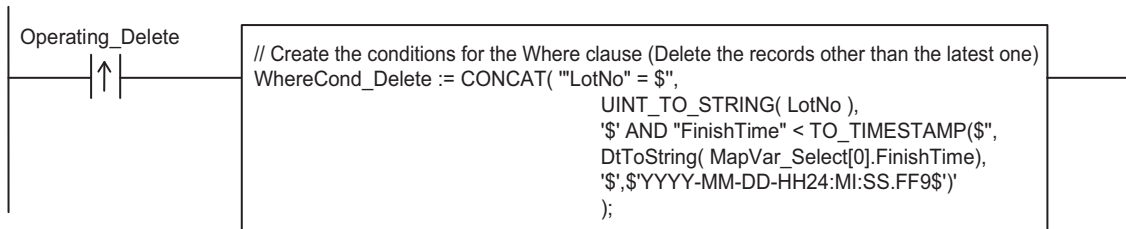
Check the completion of the DB_Delete instruction.



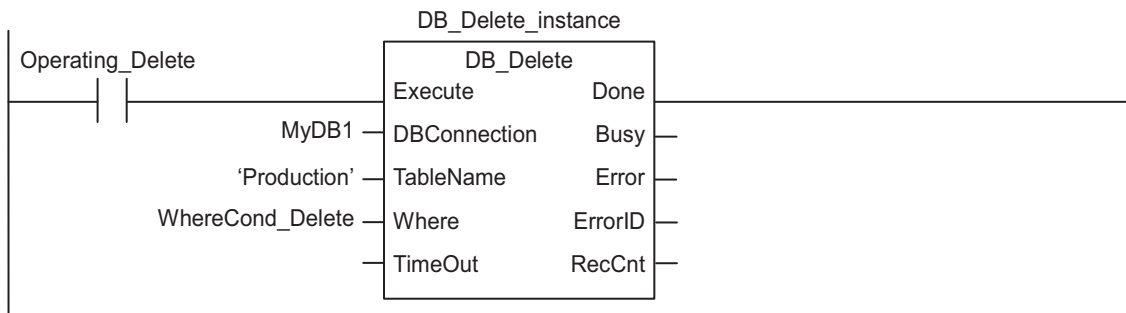
Accept the trigger for deleting DB records.



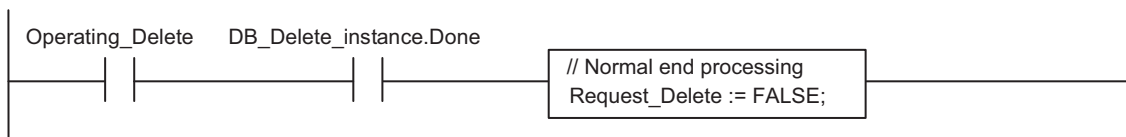
Create the conditions for Where clause.



Delete records from the table Production of the DB Connection *MyDB1*. Timeout is not monitored for the instruction execution.

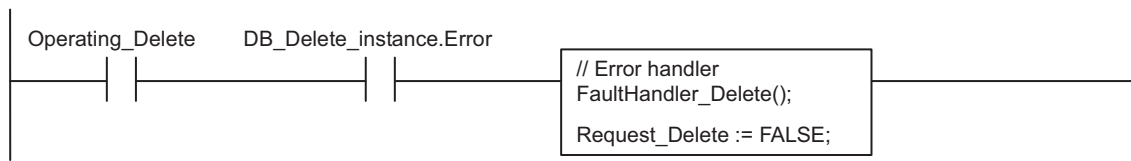


Execute the normal end processing.



When the instruction is terminated due to an error, execute the error handler for the device (FaultHandler_Delete).

Program the FaultHandler_Delete for the device.

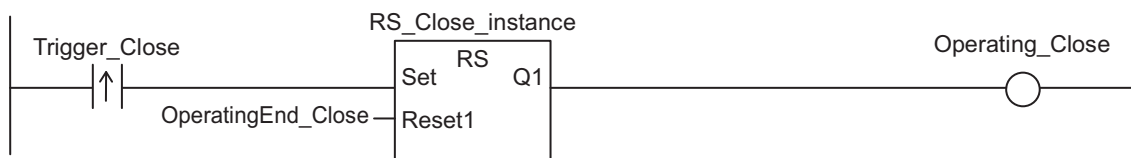


- Close the DB Connection *MyDB1*.

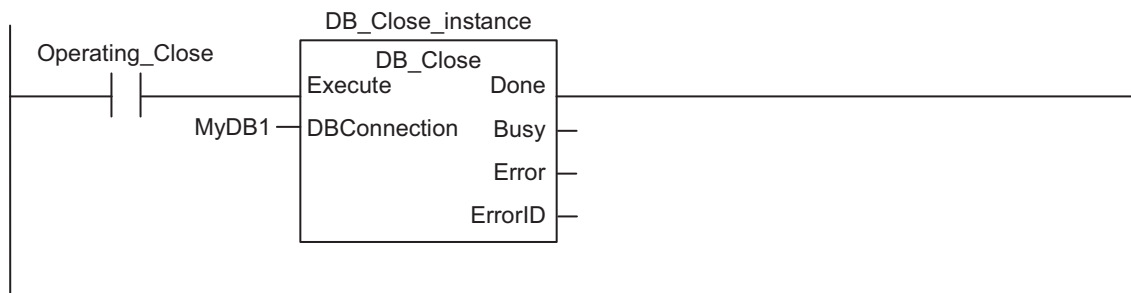
Check the completion of the DB_Close instruction.



Accept the trigger for closing the DB Connection.

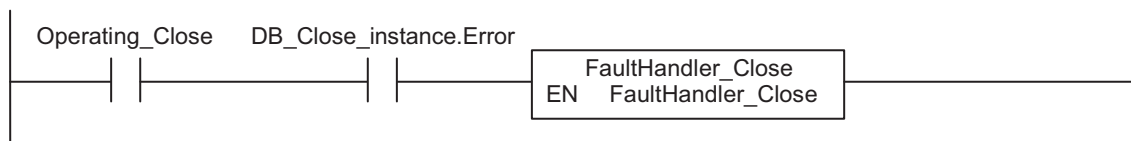


Close the DB Connection *MyDB1*.



When the instruction is terminated due to an error, execute the error handler for the device (FaultHandler_Close).

Program the FaultHandler_Connect according to the device.



Structured Text (ST)

● Main Variables

Name	Data type	Default	Comment
_DBC_Status	_sDBC_STATUS	---	System-defined variable that shows the status of the DB Connection Service
DB_Connect_instance	DB_Connect	---	Instance of DB_Connect instruction
MyDB1	DWORD	---	This variable is assigned to the DBConnection output variable from DB_Connect_instance.

Name	Data type	Default	Comment
LotNo	UINT	1234	Variable to specify the lot number for retrieving/deleting DB records
Trigger_Connect	BOOL	FALSE	Variable used as a trigger for establishing a DB Connection
LastTrigger_Connect	BOOL	FALSE	Variable to retain the trigger status of the previous execution
Operating_Connect	BOOL	FALSE	The DB_Connect instruction is executed when this variable is TRUE.
OperatingStart_Connect	BOOL	FALSE	The start processing for establishing the DB Connection is executed when this variable is TRUE.
DB_CreateMapping_Select_instance	DB_CreateMapping	---	Instance of DB_CreateMapping instruction
MapVar_Select	ARRAY[0..99] OF PRODUCTION_SELECT	---	This variable is assigned to the MapVar input variable to DB_CreateMapping_Select_instance.
DB_Select_instance	DB_Select	---	Instance of DB_Select instruction
Trigger_Select	BOOL	FALSE	Variable used as a trigger for retrieving DB records.
LastTrigger_Select	BOOL	FALSE	Variable to retain the trigger status of the previous execution
Operating_Select	BOOL	FALSE	The DB_Select instruction is executed when this variable is TRUE.
OperatingStart_Select	BOOL	FALSE	The start processing for retrieving DB records is executed when this variable is TRUE.
WhereCond_Select	STRING[256]	---	This variable is assigned to the Where input variable to DB_Select_instance.
SortCond_Select	STRING[256]	---	This variable is assigned to the Sort input variable to DB_Select_instance.
DB_Delete_instance	DB_Delete	---	Instance of DB_Delete instruction
WhereCond_Delete	STRING[256]	---	This variable is assigned to the Where input variable to DB_Delete_instance.
Request_Delete	BOOL	FALSE	The DB_Delete instruction is executed when this variable is TRUE.
LastRequest_Delete	BOOL	FALSE	Variable to retain the request status of the previous execution
Operating_Delete	BOOL	FALSE	The DB_Delete instruction is executed when this variable is TRUE.
OperatingStart_Delete	BOOL	FALSE	The start processing for deleting DB records is executed when this variable is TRUE.
DB_Close_instance	DB_Close	---	Instance of DB_Close instruction
Trigger_Close	BOOL	FALSE	Variable used as a trigger for closing the DB Connection
LastTrigger_Close	BOOL	FALSE	Variable to retain the trigger status of the previous execution
Operating_Close	BOOL	FALSE	The DB_Close instruction is executed when this variable is TRUE.
OperatingStart_Close	BOOL	FALSE	The start processing for closing the DB Connection is executed when this variable is TRUE.
Stage	INT	---	Variable that shows the status of the DB Connection

● Sample Programming

```
// -----
// - Establish a DB Connection named MyDatabase1 and map a table with a variable.
// Start the sequence when the variable Trigger_Connect changes to TRUE.
IF ( (Trigger_Connect=TRUE)
    AND (LastTrigger_Connect=FALSE)
    AND ( _DBC_Status.Run=TRUE) ) THEN
    OperatingStart_Connect := TRUE;
    Operating_Connect := TRUE;
END_IF;
LastTrigger_Connect:=Trigger_Connect;

// Sequence start processing
IF (OperatingStart_Connect=TRUE) THEN
    // Initialize the instances of the applicable DB Connection Instructions.
    DB_Connect_instance( Execute:=FALSE );
    DB_CreateMapping_Select_instance(
        Execute      := FALSE,
        MapVar       := MapVar_Select,
        SQLType      := _DBC_SQLTYPE_SELECT
    );

    Stage := 1;
    OperatingStart_Connect := FALSE;
END_IF;

// Establish the DB Connection named MyDatabase1.
// Map the variable MapVar_Select to the table Production of the DB Connection MyDB
1 for the SELECT operation.
IF (Operating_Connect=TRUE) THEN
    CASE Stage OF
    1 : // Establish the DB Connection
        DB_Connect_instance(
            Execute          := TRUE,
            DBConnectionName := 'MyDatabase1',
            DBConnection     => MyDB1
        );

        IF (DB_Connect_instance.Done=TRUE) THEN
            Stage := INT#2; // Normal end
        END_IF;
        IF (DB_Connect_instance.Error=TRUE) THEN
            Stage := INT#99; // Error
        END_IF;

    2 : // Map the DB table with the variable
        DB_CreateMapping_Select_instance(
```

```

        Execute          := TRUE,
        DBConnection     := MyDB1,
        TableName        := 'Production',
        MapVar           := MapVar_Select,
        SQLType          := _DBC_SQLTYPE_SELECT
    );

    IF (DB_CreateMapping_Select_instance.Done=TRUE) THEN
        Operating_Connect:=FALSE; // Normal end
    END_IF;
    IF (DB_CreateMapping_Select_instance.Error=TRUE) THEN
        Stage := INT#99; // Error
    END_IF;

99 :
    // Execute the error handler.
    // Program the error handler (FaultHandler_Connect) according to the device.
    FaultHandler_Connect();
    Operating_Connect := FALSE;
    END_CASE;
END_IF;

// -----
// - Retrieve the records for the specified lot number from the DB Connection MyDB1
.

// Start the sequence when the variable Trigger_Select changes to TRUE.
IF ( (Trigger_Select=TRUE) AND (LastTrigger_Select=FALSE) ) THEN
    OperatingStart_Select := TRUE;
    Operating_Select := TRUE;
END_IF;
LastTrigger_Select := Trigger_Select;

// Sequence start processing
IF (OperatingStart_Select=TRUE) THEN
    // Initialize the instance of the applicable DB Connection Instruction.
    DB_Select_instance( Execute:=FALSE, MapVar:=MapVar_Select );

    // Create the conditions for the Where clause ("LotNo" = XXXX).
    WhereCond_Select := CONCAT( 'LotNo' = '$', UINT_TO_STRING( LotNo ), '$' );

    // Create the conditions for the Sort clause.
    // Sort the production completion time in descending order.
    SortCond_Select := '"FinishTime" DESC';

    OperatingStart_Select := FALSE;
END_IF;

```

```

// Retrieve the records from the DB Connection MyDB1. Timeout is not monitored fo
r the instruction execution.
IF (Operating_Select=TRUE) THEN
    // Retrieve records.
    DB_Select_instance(
        Execute      := TRUE,
        DBConnection := MyDB1,
        Where        := WhereCond_Select,
        Sort         := SortCond_Select,
        MapVar       := MapVar_Select
    );

IF (DB_Select_instance.Done=TRUE) THEN
    // If two or more records were retrieved, delete the older records.
    IF (DB_Select_instance.SelectedCnt > 1) THEN
        Request_Delete := TRUE;
    END_IF;
    Operating_Select:=FALSE; // Normal end
END_IF;
IF (DB_Select_instance.Error=TRUE) THEN
    // Error handler.
    // Program the error handler (FaultHandler_Select) according to the device.
    FaultHandler_Select();

    Operating_Select := FALSE;
END_IF;
END_IF;

// -----
// - Delete the records other than the latest one from the DB table.

// Start the sequence when the variable Trigger_Delete changes to TRUE.
IF ( (Request_Delete=TRUE) AND (LastRequest_Delete=FALSE) ) THEN
    OperatingStart_Delete := TRUE;
    Operating_Delete := TRUE;
END_IF;
LastRequest_Delete := Request_Delete;

// Sequence start processing
IF (OperatingStart_Delete=TRUE) THEN
    // Initialize the instance of the applicable DB Connection Instruction.
    DB_Delete_instance( Execute:=FALSE );

    // Create the conditions for the Where clause (delete the records other than the
latest one).
    WhereCond_Delete := CONCAT( '"LotNo" = $',

```

```

        UINT_TO_STRING( LotNo ),
        '$' AND "FinishTime" < TO_TIMESTAMP('$',
        DtToString( MapVar_Select[0].FinishTime),
        '$', '$YYYY-MM-DD-HH24:MI:SS.FF9$') '
    );
    OperatingStart_Delete := FALSE;
END_IF;

// Delete records from the table Production of the DB Connection MyDB1. Timeout is
// not monitored for the instruction execution.
IF (Operating_Delete=TRUE) THEN
    // Delete the records.
    DB_Delete_instance(
        Execute      := TRUE,
        DBConnection := MyDB1,
        TableName    := 'Production',
        Where        := WhereCond_Delete
    );

    IF (DB_Delete_instance.Done=TRUE) THEN
        Operating_Delete :=FALSE; // Normal end
        Request_Delete :=FALSE;
    END_IF;
    IF (DB_Delete_instance.Error=TRUE) THEN
        // Execute the error handler.
        // Program the error handler (FaultHandler_Delete) for the device.
        FaultHandler_Update();

        Operating_Delete := FALSE;
        Request_Delete :=FALSE;
    END_IF;
END_IF;

// -----
// - Close the DB Connection MyDB1.

// Start the sequence when the variable Trigger_Close changes to TRUE.
IF ( (Trigger_Close=TRUE) AND (LastTrigger_Close=FALSE) ) THEN
    OperatingStart_Close := TRUE;
    Operating_Close := TRUE;
END_IF;
LastTrigger_Close := Trigger_Close;

// Sequence start processing
IF (OperatingStart_Close=TRUE) THEN
    // Initialize the instance of the applicable DB Connection Instruction.
    DB_Close_instance( Execute:=FALSE );

```

```

    OperatingStart_Close := FALSE;
END_IF;

// Close the DB Connection MyDB1.
IF (Operating_Close=TRUE) THEN
    // Close the DB Connection.
    DB_Close_instance( Execute:=TRUE, DBConnection:=MyDB1 );

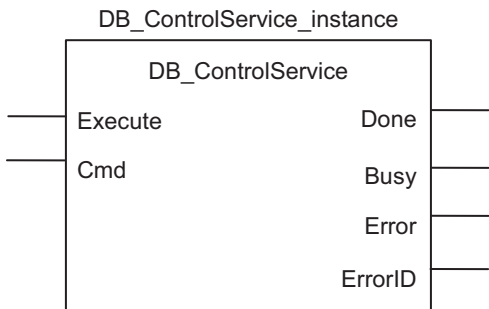
    IF (DB_Close_instance.Done=TRUE) THEN
        Operating_Close := FALSE; // Normal end
    END_IF;
    IF (DB_Close_instance.Error=TRUE) THEN
        // Error handler
        // Program the error handler (FaultHandler_Close) according to the device.
        FaultHandler_Close();

        Operating_Close := FALSE;
    END_IF;
END_IF;

```

DB_ControlService (Control DB Connection Service)

The DB_ControlService instruction starts/stops the DB Connection Service or starts/finishes recording to the Debug Log.

Instruction	Name	FB/FUN	Graphic expression	ST expression
DB_ControlService	Control DB Connection Service	FB		DB_ControlService_instance (Execute, Cmd, Done, Busy, Error, ErrorID);

Note The DB_ControlService_instance is an instance of DB_ControlService instruction, which is declared as a variable.

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.
Cmd	Command	_eDBC_CMD	_DBC_CMD_START(1): Start the service in Operation Mode _DBC_CMD_START_TEST(2): Start the service in Test Mode _DBC_CMD_STOP(3): Stop the service _DBC_CMD_DEBUGLOG_ON(4): Start recording to Debug Log _DBC_CMD_DEBUGLOG_OFF(5): Finish recording to Debug Log		0	Specify the command to execute

Output Variable

Name	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.

Name	Meaning	Data type	Valid range	Unit	Description
Error	Error	BOOL	TRUE or FALSE	---	TRUE when the instruction is terminated due to an error.
ErrorID	Error Code	WORD	16#0000 to 16#FFFF	---	Contains the error code when an error occurs.

Related System-defined Variables

System-defined variables	Name	Data type	Valid range	Unit	Description
_DBC_Status.Idle	DB Connection Service Idle Status	BOOL	TRUE or FALSE	---	TRUE when the operation status of the DB Connection Service is Idle. Otherwise, FALSE.
_DBC_Status.Run	DB Connection Service Running Status	BOOL	TRUE or FALSE	---	TRUE when the DB Connection Service is started in Operation Mode or Test Mode. FALSE when the DB Connection Service is stopped.
_DBC_Status.Test	DB Connection Service Test Mode Status	BOOL	TRUE or FALSE	---	TRUE when the DB Connection Service is started in Test Mode. FALSE when the DB Connection Service is stopped.
_DBC_Status.Shutdown	DB Connection Service Shutdown Status	BOOL	TRUE or FALSE	---	TRUE when the operation status of the DB Connection Service is shutdown. Otherwise, FALSE.

Related Error Codes

Error code	Meaning	Description
0400 hex	Input Value Out of Range	A value that is not defined as an enumerator was specified in the <i>Cmd</i> input variable.
041D hex	Too Many Instructions Executed at the Same Time	More than 32 DB Connection Instructions were executed at the same time.
1400 hex	SD Memory Card Access Failure	This instruction was executed with _DBC_CMD_DEBUGLOG_ON selected in the <i>Cmd</i> input variable when the SD Memory Card was not available
1401 hex	SD Memory Card Write-protected	This instruction was executed with _DBC_CMD_DEBUGLOG_ON selected in the <i>Cmd</i> input variable when the SD Memory Card was write-protected.
3001 hex	DB Connection Service Run Mode Change Failed	When this instruction was executed with _DBC_CMD_START_TEST selected in the <i>Cmd</i> input variable while the service was running in Operation Mode This instruction was executed with _DBC_CMD_START selected in the <i>Cmd</i> input variable while the service was running in Test Mode. Start of the DB Connection Service was commanded while the DB Connection Service was being stopped.
3002 hex	DB Connection Service Shutdown or Shutting Down	The instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
3013 hex	DB Connection Service Error Stop	The instruction was executed while the DB Connection Service was stopped due to an error.
3015 hex	DB Connection Service Initializing	The instruction was executed while the initialization processing of the DB Connection Service was in progress.

Function

This instruction is used to start and stop the DB Connection Service, and start and finish recording to the Debug Log.

When the DB can be connected, start the DB Connection Service in Operation Mode.

When there is no DB, for example, in the course of development, start the DB Connection Service in Test Mode. In this case, the following instructions are normally completed without accessing the DB nor executing the SQL statement actually: DB_Connect, DB_CreateMapping, DB_Insert, DB_Update, DB_Select and DB_Delete.

When the DB Connection Service is stopped, the established connections are all closed.

When recording to the debug log is started, the detailed log for each execution of DB Connection Instructions (such as transmitted SQL statements) is output to the Debug Log file in the SD Memory Card.

Precautions for Correct Use

- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual* (Cat. No. W502) for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- This instruction cannot be used on an event task. A compiling error will occur.
- When starting the DB Connection Service, confirm that the value of `_DBC_Status.Idle` is TRUE and then execute this instruction. If this instruction is executed while the DB Connection Service is being initialized, an error (DB Connection Connection Service Initializing) will occur.
- It is impossible to change the DB Connection Service from Operation Mode to Test Mode and vice versa while the DB Connection Service is running. Stop the service before changing the Run mode.
- The recording status of the Debug Log (i.e. whether or not to record the Debug Log) is held after the DB Connection Service is stopped and started again.
- Besides this instruction, recording to the Debug Log is stopped in the following cases.
 - a) When a DB_Shutdown instruction is executed
 - b) When the power supply to the CPU Unit is turned OFF
 - c) When the SD Memory Card is taken out
- An error occurs for this instruction in the following cases. *Error* will be TRUE.
 - a) When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
 - b) When the instruction was executed while the DB Connection Service was stopped due to an error.
 - c) When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
 - d) When this instruction was executed with `_DBC_CMD_START_TEST` selected in the *Cmd* input variable while the service was running in Operation Mode
 - e) This instruction was executed with `_DBC_CMD_START` selected in the *Cmd* input variable while the service was running in Test Mode.
 - f) Start of the DB Connection Service was commanded while the DB Connection Service was being stopped.

- g) When this instruction was executed with `_DBC_CMD_DEBUGLOG_ON` selected in the *Cmd* input variable when the SD Memory Card was not available or write-protected
- h) A value that is not defined as an enumerator was specified in the *Cmd* input variable.
- i) When more than 32 DB Connection Instructions were executed at the same time.

Sample Programming

This section gives sample programming for starting recording to the Debug Log when the trigger variable changes to TRUE and finishing the recording when another trigger variable changes to FALSE.

Ladder Diagram

● Main Variables

Name	Data type	Default	Comment
DB_ControlService_instance	DB_ControlService	---	Instance of DB_ControlService instruction
LogOn	BOOL	FALSE	Variable used as a trigger for controlling the Debug Log
Operating	BOOL	FALSE	The DB_ControlService instruction is executed when this variable is TRUE.
OperatingEnd	BOOL	FALSE	This variable changes to TRUE when the DB_ControlService instruction is completed.
RS_instance	RS	---	Instance of RS instruction
MyCmd	_eDBC_CMD	---	This variable is assigned to the <i>Cmd</i> input variable to DB_ControlService_instance.
ControlService_OK	BOOL	FALSE	This variable changes to TRUE when the DB_ControlService instruction is completed normally.

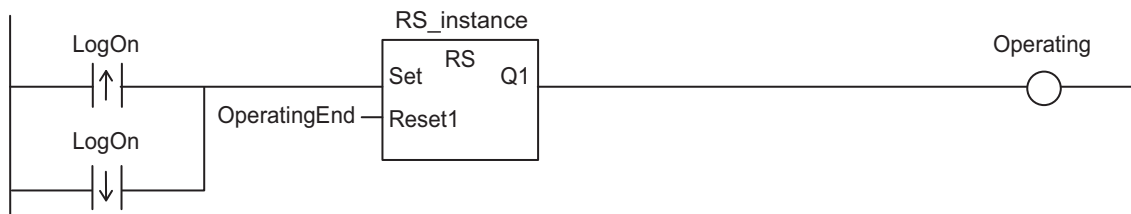
● Sample Programming

- Start recording to the Debug Log when the variable `LogOn` changes to TRUE. Finish the recording when the variable `LogOn` changes to FALSE.

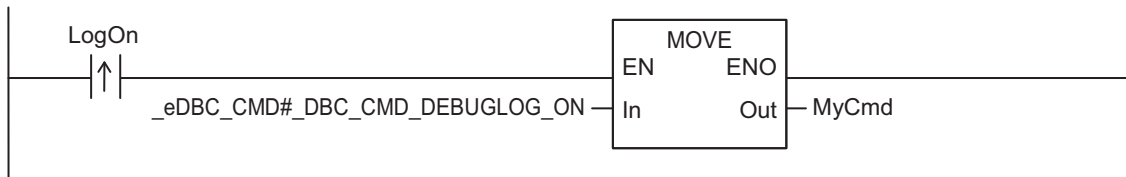
Check the completion of DB_ControlService instruction.



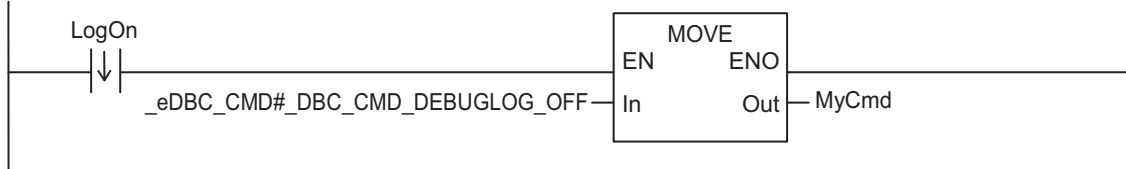
Accept the trigger for controlling the Debug Log.



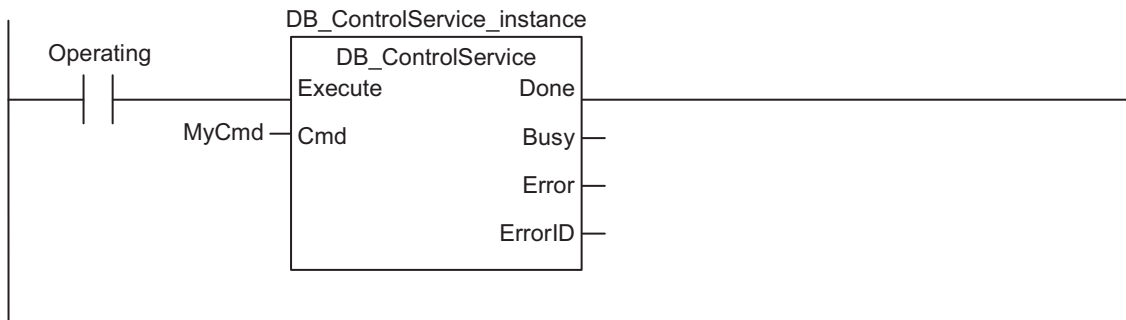
Start recording to the Debug Log.



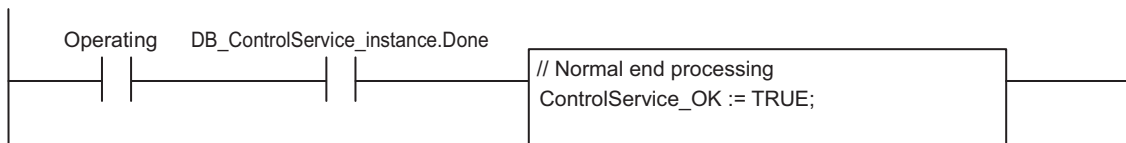
Finish recording to the Debug Log.



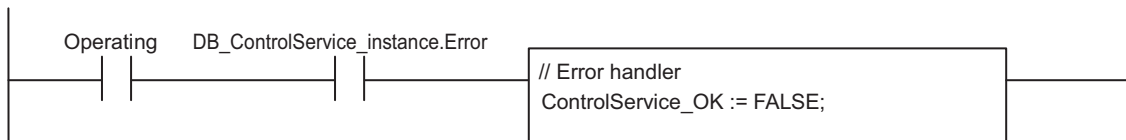
Command to start/finish recording to the Debug Log.



When the instruction is normally completed, change the variable ControlService_OK to TRUE.



When the instruction is terminated due to an error, change the variable ControlService_OK to FALSE.



Structured Text (ST)

● Main Variables

Name	Data type	Default	Comment
DB_ControlService_instance	DB_ControlService	---	Instance of DB_ControlService instruction
LogOn	BOOL	FALSE	Variable used as a trigger for controlling the Debug Log
LastTrigger	BOOL	FALSE	Variable to retain the trigger status of the previous execution
Operating	BOOL	FALSE	The DB_ControlService instruction is executed when this variable is TRUE.
OperatingStart	BOOL	FALSE	The initialization processing is executed when this variable is TRUE.
MyCmd	_eDBC_CMD	---	This variable is assigned to the Cmd input variable to DB_ControlService_instance.

● Sample Programming

```

(* -----
- Start recording to the Debug Log when the variable LogOn changes to TRUE.
  Finish the recording when the variable LogOn changes to FALSE.
----- *)

// Start the sequence when the variable LogOn changes to TRUE.
IF ( (LogOn=TRUE) AND (LastTrigger=FALSE) ) THEN
  OperatingStart := TRUE;
  Operating := TRUE;
  MyCmd := _DBC_CMD_DEBUGLOG_ON; // Start recording to the Debug Log.
ELSIF ( (LogOn=FALSE) AND (LastTrigger=TRUE) ) THEN
  OperatingStart := TRUE;
  Operating := TRUE;
  MyCmd := _DBC_CMD_DEBUGLOG_OFF; // Finish recording to the Debug Log.
END_IF;
LastTrigger := LogOn;

// Sequence start processing
IF (OperatingStart=TRUE) THEN
  // Initialize the instruction instance.
  DB_ControlService_instance( Execute:=FALSE );
  OperatingStart := FALSE;
END_IF;

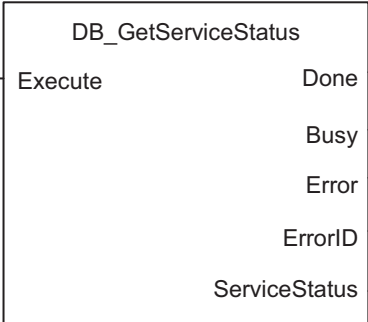
// Command to start or finish recording to the Debug Log.
IF (Operating=TRUE) THEN
  // Start or finish recording to the Debug Log.
  DB_ControlService_instance(
    Execute := TRUE,
    Cmd := MyCmd
  );

  IF (DB_ControlService_instance.Done=TRUE) THEN
    // Normal end processing
    Operating := FALSE;
  END_IF;
  IF (DB_ControlService_instance.Error=TRUE) THEN
    // Error handler.
    Operating := FALSE;
  END_IF;
END_IF;

```

DB_GetServiceStatus (Get DB Connection Service Status)

The DB_GetServiceStatus instruction gets the current status of the DB Connection Service.

Instruction	Name	FB/FUN	Graphic expression	ST expression
DB_GetServiceStatus	Get DB Connection Service Status	FB		DB_GetServiceStatus_instance (Execute, Done, Busy, Error, ErrorID, ServiceStatus);

Note The DB_GetServiceStatus_instance is an instance of DB_GetServiceStatus instruction, which is declared as a variable.

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.

Output Variable

Name	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.
Error	Error	BOOL	TRUE or FALSE	---	TRUE when the instruction is terminated due to an error.
ErrorID	Error Code	WORD	16#0000 to 16#FFFF	---	Contains the error code when an error occurs.
Service-Status	DB Connection Service Status	_sDBC_SERVICE_STATUS	Depends on the data type.		Shows the status of the DB Connection Service.

Related Error Codes

Error code	Meaning	Description
041D hex	Too Many Instructions Executed at the Same Time	More than 32 DB Connection Instructions were executed at the same time.
3002 hex	DB Connection Service Shut-down or Shutting Down	The instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
3013 hex	DB Connection Service Error Stop	The instruction was executed while the DB Connection Service was stopped due to an error.
3015 hex	DB Connection Service Initializing	The instruction was executed while the initialization processing of the DB Connection Service was in progress.

Function

This instruction is used to get the current status of the DB Connection Service. The current status is output to the *ServiceStatus* output variable.

Refer to the *ServiceStatus* on page 7-3 of *Common Input and Output Variables Used in the DB Connection Instructions* on page 7-2 for the status.

Precautions for Correct Use

- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual* (Cat. No. W502) for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- This instruction cannot be used on an event task. A compiling error will occur.
- An error occurs for this instruction in the following cases. *Error* will be TRUE.
 - a) When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
 - b) When the instruction was executed while the DB Connection Service was stopped due to an error.
 - c) When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
 - d) When more than 32 DB Connection Instructions were executed at the same time.

Sample Programming

This section gives sample programming for the following operations.

- Get the status of the DB Connection Service when the trigger variable changes to TRUE.
- Change the value of the Warning variable to TRUE if the number of error executions is 100 or greater.

Ladder Diagram

● Main Variables

Name	Data type	Default	Comment
DB_GetServiceStatus_instance	DB_GetServiceStatus	---	Instance of DB_GetServiceStatus instruction
Trigger	BOOL	FALSE	Variable used as a trigger for getting the status of the DB Connection Service
Operating	BOOL	FALSE	The DB_GetServiceStatus instruction is executed when this variable is TRUE.
OperatingEnd	BOOL	FALSE	This variable changes to TRUE when the DB_GetServiceStatus instruction is completed.
RS_instance	RS	---	Instance of RS instruction
MyStatus	_sDBC_SERVICE_STATUS	---	This variable is assigned to the ServiceStatus input variable to DB_GetServiceStatus_instance.
Warning	BOOL	FALSE	This variable changes to TRUE when the number of error executions is 100 or greater.
GetServiceStatus_OK	BOOL	FALSE	This variable changes to TRUE when the DB_GetServiceStatus instruction is completed normally.

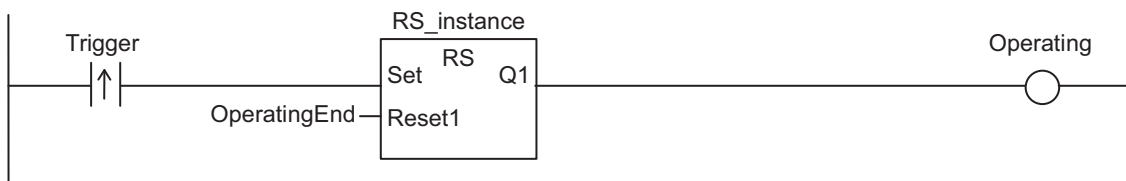
● Sample Programming

- Change the value of the variable Warning to TRUE when the number of error executions is 100 or greater.

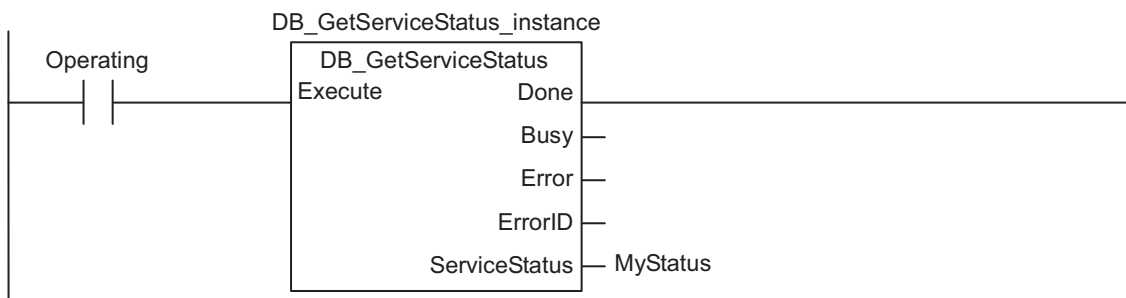
Check the completion of the DB_GetServiceStatus instruction.



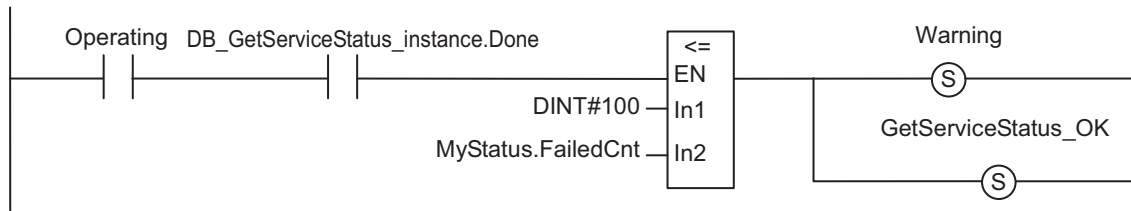
Accept the trigger.



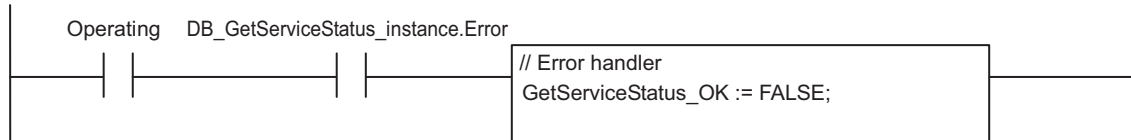
Get the status of the DB Connection Service.



When the instruction is normally completed, change the variable Warning to TRUE if the number of error executions is 100 or greater.



When the instruction is terminated due to an error, change the variable Warning to FALSE.



Structured Text (ST)

● Main Variables

Meaning	Data type	Default	Comment
DB_GetServiceStatus_instance	DB_GetServiceStatus	---	Instance of DB_GetServiceStatus instruction
Trigger	BOOL	FALSE	Variable used as a trigger for getting the status of the DB Connection Service
LastTrigger	BOOL	FALSE	Variable to retain the trigger status of the previous execution
Operating	BOOL	FALSE	The DB_GetServiceStatus instruction is executed when this variable is TRUE.
OperatingStart	BOOL	FALSE	The initialization processing is executed when this variable is TRUE.
MyStatus	_sDBC_SERVICE_STATUS	---	This variable is assigned to the ServiceStatus input variable to DB_GetServiceStatus_instance.
Warning	BOOL	FALSE	This variable changes to TRUE when the number of error executions is 100 or greater.

● Sample Programming

```
(* -----
- Change the value of the variable Warning to TRUE when the number of SQL execution failures in all connections is 100 or greater.
----- *)

// Start the sequence when the variable Trigger changes to TRUE.
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) ) THEN
    OperatingStart := TRUE;
    Operating := TRUE;
END_IF;
LastTrigger := Trigger;

// Sequence start processing
IF (OperatingStart=TRUE) THEN
    // Initialize the instruction instance.
```

```
DB_GetServiceStatus_instance( Execute:=FALSE );
OperatingStart := FALSE;
END_IF;

IF (Operating=TRUE) THEN
  // Get the status of the DB Connection Service.
  DB_GetServiceStatus_instance(
    Execute := TRUE,
    ServiceStatus => MyStatus
  );

  IF (DB_GetServiceStatus_instance.Done=TRUE) THEN
    // Normal end processing
    // Change the variable Warning to TRUE when the number of error executions is 1
00 or greater.
    IF (MyStatus.FailedCnt >= DINT#100) THEN
      Warning := TRUE;
    END_IF;
    Operating := FALSE;
  END_IF;
  IF (DB_GetServiceStatus_instance.Error=TRUE) THEN
    // Error handler
    Operating := FALSE;
  END_IF;
END_IF;
```

DB_GetConnectionStatus (Get DB Connection Status)

The DB_GetConnectionStatus instruction gets the status of a DB Connection.

Instruction	Name	FB/FUN	Graphic expression	ST expression
DB_GetConnectionStatus	Get DB Connection Status	FB		DB_GetConnectionStatus_instance (Execute, DBConnectionName, Done, Busy, Error, ErrorID, ConnectionStatus);

Note The DB_GetConnectionStatus_instance is an instance of DB_GetConnectionStatus instruction, which is declared as a variable.

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.
DBConnectionName	DB Connection Name	STRING	17 bytes max. (including the final NULL character)	---	''	Specify a DB Connection name set on Sysmac Studio.

Output Variable

Name	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.
Error	Error	BOOL	TRUE or FALSE	---	TRUE when the instruction is terminated due to an error.
ErrorID	Error Code	WORD	16#0000 to 16#FFFF	---	Contains the error code when an error occurs.
ConnectionStatus	Connection-Status	_sDBC_CONNECTION_STATUS	Depends on the data type.		Shows the status of the connection specified in the DBConnectionName input variable.

Related System-defined Variables

Name	Meaning	Data type	Description
_EIP_EtnOnlineSta	Online	BOOL	Status of the communications function of the built-in EtherNet/IP port. TRUE: Can be used. FALSE: Cannot be used.

Related Error Codes

Error code	Meaning	Description
0406 hex	Illegal Data Position Specified	When the <i>DBConnectionName</i> input variable is a text string consisting of NULL characters (16#00) only.
0410 hex	Text String Format Error	A space character is included in the text string specified for the <i>DBConnectionName</i> input variable. The <i>DBConnectionName</i> input variable does not end in NULL.
041D hex	Too Many Instructions Executed at the Same Time	More than 32 DB Connection Instructions were executed at the same time.
3000 hex	DB Connection Service not Started	The instruction was executed when the DB Connection Service was not running.
3002 hex	DB Connection Service Shutdown or Shutting Down	The instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
3003 hex	Invalid DB Connection Name	When the DB Connection name specified in the <i>DBConnectionName</i> input variable is not set in any DB Connection Settings.
3013 hex	DB Connection Service Error Stop	The instruction was executed while the DB Connection Service was stopped due to an error.
3015 hex	DB Connection Service Initializing	The instruction was executed while the initialization processing of the DB Connection Service was in progress.

Function

This instruction is used to get the status of the DB Connection specified in the *DBConnection* input variable. The current status is output to the *ConnectionStatus* output variable.

Refer to the *ConnectionStatus* on page 7-4 of *Common Input and Output Variables Used in the DB Connection Instructions* on page 7-2 for the status.

Refer to A-2-3 *How to Measure DB Response Time* on page A-23 for the measurement of the DB response time.

Precautions for Correct Use

- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual* (Cat. No. W502) for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- This instruction cannot be used on an event task. A compiling error will occur.
- If you execute this instruction before completion of a DB_Connect instruction and confirm that the connection status of the DB Connection is "Connected", an instruction error (Invalid DB Connection) may occur when you execute the next DB Connection Instruction. When you use the *DBConnection*

output variable from the DB_Connect instruction, confirm that the *Done* output variable of the DB_Connect instruction is TRUE or the value of the *DBConnection* output variable is not 16#00000000 before executing the DB Connection Instruction.

- An error occurs for this instruction in the following cases. *Error* will be TRUE.
 - a) When the instruction was executed when the DB Connection Service was not running.
 - b) When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
 - c) When the instruction was executed while the DB Connection Service was stopped due to an error.
 - d) When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
 - e) When the DB Connection name specified in the *DBConnectionName* input variable is not set in any DB Connection Settings.
 - f) When the *DBConnectionName* input variable is a text string consisting of NULL characters (16#00) only.
 - g) A space character is included in the text string specified for the *DBConnectionName* input variable.
 - h) The *DBConnectionName* input variable does not end in NULL.
 - i) When more than 32 DB Connection Instructions were executed at the same time.

Sample Programming

This section gives sample programming for the following operations.

- Get the status of the DB Connection when the trigger variable changes to TRUE.
- Change the value of the Warning variable to TRUE when the spool usage has exceeded 80%.

Ladder Diagram

● Main Variables

Meaning	Data type	Default	Comment
DB_GetConnection-Status_instance	DB_GetConnectionStatus	---	Instance of DB_GetConnectionStatus instruction
Trigger	BOOL	FALSE	Variable used as a trigger for getting the status of the DB Connection
Operating	BOOL	FALSE	The DB_GetConnectionStatus instruction is executed when this variable is TRUE.
OperatingEnd	BOOL	FALSE	This variable changes to TRUE when the DB_GetConnectionStatus instruction is completed.
RS_instance	RS	---	Instance of RS instruction
MyStatus	_sDBC_CONNECTION_STATUS	---	This variable is assigned to the ConnectionStatus output variable from DB_GetConnectionStatus_instance.
Warning	BOOL	FALSE	This variable changes to TRUE when the Spool usage has exceeded 80%.
GetConnectionStatus_OK	BOOL	FALSE	This variable changes to TRUE when the DB_GetConnectionStatus instruction is completed normally.

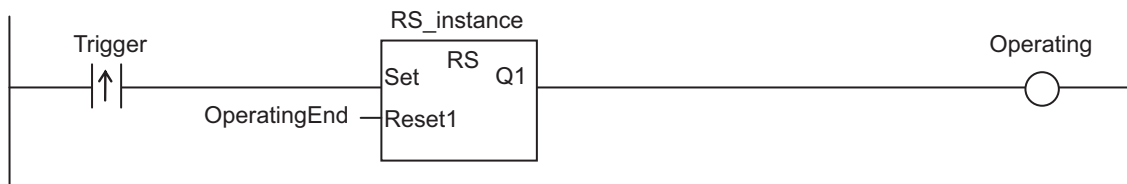
● Sample Programming

Change the variable Warning to TRUE when the Spool usage of the DB Connection named *MyDatabase1* has exceeded 80%.

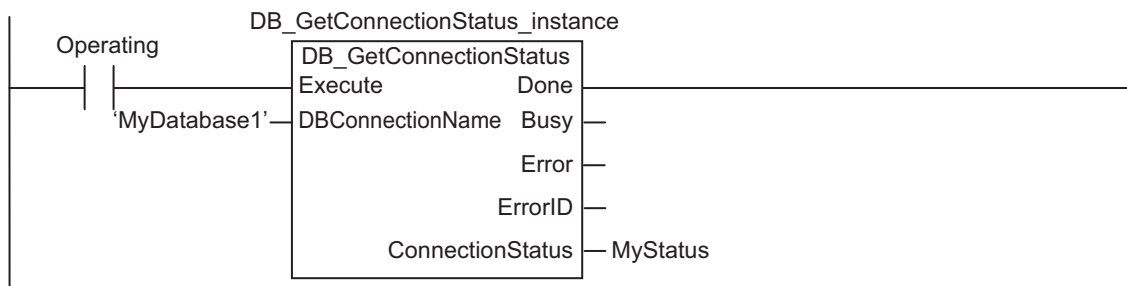
Check the completion of the DB_GetConnectionStatus instruction.



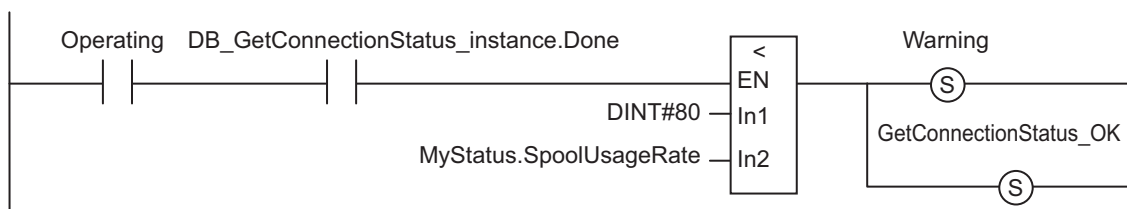
Accept the trigger.



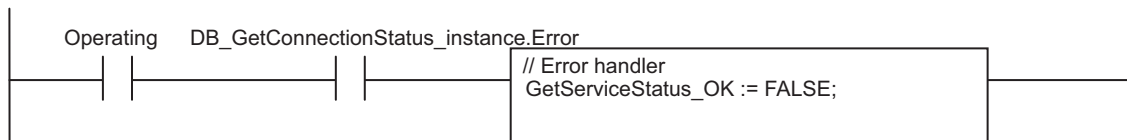
Get the status of the DB Connection.



When the instruction is normally completed, change the value of the variable Warning to TRUE if the Spool usage has exceeded 80%.



When the instruction is terminated due to an error, change the variable Warning to FALSE.



Structured Text (ST)

● Main Variables

Meaning	Data type	Default	Comment
DB_GetConnec- tionStatus _instance	DB_GetConnection- Status	---	Instance of DB_GetConnectionStatus instruction

Meaning	Data type	Default	Comment
Trigger	BOOL	FALSE	Variable used as a trigger for getting the status of the DB Connection
LastTrigger	BOOL	FALSE	Variable to retain the trigger status of the previous execution
Operating	BOOL	FALSE	The DB_GetConnectionStatus instruction is executed when this variable is TRUE.
OperatingStart	BOOL	FALSE	The initialization processing is executed when this variable is TRUE.
MyStatus	_sDBC_CONNECTION_STATUS	---	This variable is assigned to the ConnectionStatus output variable from DB_GetConnectionStatus_instance.
Warning	BOOL	FALSE	This variable changes to TRUE when the Spool usage has exceeded 80%.

● Sample Programming

```

(* -----
- Change the variable Warning to TRUE when the Spool usage of the DB Connection named MyDatabasel has exceeded 80%.
----- *)

// Start the sequence when the variable Trigger changes to TRUE.
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) ) THEN
    OperatingStart := TRUE;
    Operating := TRUE;
END_IF;
LastTrigger := Trigger;

// Sequence start processing
IF (OperatingStart=TRUE) THEN
    // Initialize the instruction instance.
    DB_GetConnectionStatus_instance( Execute:=FALSE );
    OperatingStart := FALSE;
END_IF;

IF (Operating=TRUE) THEN
    // Get the status of the DB Connection.
    DB_GetConnectionStatus_instance(
        Execute           := TRUE,
        DBConnectionName  := 'MyDatabasel',
        ConnectionStatus  => MyStatus
    );

    IF (DB_GetConnectionStatus_instance.Done=TRUE) THEN
        // Normal end processing
        // Change the variable Warning to TRUE when the Spool usage has exceeded 80%.
        IF (MyStatus.SpoolUsageRate > SINT#80) THEN
            Warning := TRUE;
        END_IF;
    END_IF;
END_IF;

```

```
        Operating := FALSE;  
END_IF;  
IF (DB_GetConnectionStatus_instance.Error=TRUE) THEN  
    // Error handler  
    Operating := FALSE;  
END_IF;  
END_IF;
```


DB_ControlSpool (Resend/Clear Spool Data)

The DB_ControlSpool instruction resends or clears the SQL statements spooled by DB_Insert (Insert DB Record) and DB_Update (Update DB Record) instructions.

Instruction	Name	FB/FUN	Graphic expression	ST expression
DB_ControlSpool	Resend/ Clear Spool Data	FB		DB_ControlSpool_instance (Execute, DBConnection, Cmd, Done, Busy, Error, ErrorID);

Note The DB_ControlSpool_instance is an instance of DB_ControlSpool instruction, which is declared as a variable.

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.
DBCon- nection	DB Con- nection	DWORD	16#00000000 to 16#FFFFFFF		16#00000000	Specify the DB connection established by a DB_Connect instruction.
Cmd	Com- mand	_eDBC_SPOOL_CMD	_DBC_SPOOL_CLEAR(1): Clear _DBC_SPOOL_RESEND(2): Resend		0	Specify the command to execute

Output Variable

Name	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.
Error	Error	BOOL	TRUE or FALSE		TRUE when the instruction is terminated due to an error.

Name	Meaning	Data type	Valid range	Unit	Description
ErrorID	Error Code	WORD	16#0000 to 16#FFFF	---	Contains the error code when an error occurs.

Related System-defined Variables

Name	Meaning	Data type	Description
_EIP_EtnOnlineSta	Online	BOOL	Status of the communications function of the built-in EtherNet/IP port. TRUE: Can be used. FALSE: Cannot be used.

Related Error Codes

Error code	Meaning	Description
0400 hex	Input Value Out of Range	A value that is not defined as an enumerator was specified in the <i>Cmd</i> input variable.
041D hex	Too Many Instructions Executed at the Same Time	More than 32 DB Connection Instructions were executed at the same time.
3000 hex	DB Connection Service not Started	The Resend Spool Data operation was executed by this instruction when the DB Connection Service was not running.
3002 hex	DB Connection Service Shut-down or Shutting Down	The instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
3008 hex	Invalid DB Connection	When the value of the <i>DBConnection</i> input variable is invalid or the specified DB Connection is already closed.
300B hex	SQL Execution Error	The executed SQL statement resulted in an error in the DB.
3011 hex	DB Connection Disconnected Error Status	The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
3013 hex	DB Connection Service Error Stop	The instruction was executed while the DB Connection Service was stopped due to an error.
3015 hex	DB Connection Service Initializing	The instruction was executed while the initialization processing of the DB Connection Service was in progress.

Function

This instruction is used to resend or clear the SQL statements stored in the Spool memory for the DB Connection specified in the *DBConnection* input variable.

When you select manual resend for Spool data, the SQL statements stored in the Spool memory are resent by executing this instruction.

Precautions for Correct Use

- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- When you execute this instruction to resend the Spool data, this instruction just starts the Spool data resending processing. When the value of the *Done* output variable changes to TRUE, the resending processing of the SQL statements stored in the Spool memory has not been completed. Confirm the

completion of resending processing by reading the number of "Spool data" using the DB_GetConnectionStatus instruction.

- When the Spool function is not enabled, this instruction will be completed normally without executing the resend or clear processing of the SQL statements stored in the Spool memory.
- The Clear Spool Data operation can be executed even when the DB Connection Service is not running.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual* (Cat. No. W502) for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- This instruction cannot be used on an event task. A compiling error will occur.
- An error occurs for this instruction in the following cases. *Error* will be TRUE.
 - a) The Resend Spool Data operation was executed by this instruction when the DB Connection Service was not running.
 - b) When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
 - c) When the instruction was executed while the DB Connection Service was stopped due to an error.
 - d) When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
 - e) When the value of the *DBConnection* input variable is invalid or the specified DB Connection is already closed.
 - f) A value that is not defined as an enumerator was specified in the *Cmd* input variable.
 - g) The executed SQL statement resulted in an error in the DB.
 - h) The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
 - i) When more than 32 DB Connection Instructions were executed at the same time.

Sample Programming

This section gives sample programming for resending the SQL statements stored in the Spool memory if the status of the DB Connection is "Connected" when the trigger variable changes to TRUE.

Ladder Diagram

● Main Variables

Name	Data type	Default	Comment
DB_GetConnectionStatus_instance	DB_GetConnectionStatus	---	Instance of DB_GetConnectionStatus instruction
DB_ControlSpool_instance	DB_ControlSpool	---	Instance of DB_ControlSpool instruction
Trigger	BOOL	FALSE	Variable used as a trigger for resending the Spool data
Operating	BOOL	FALSE	When this variable is TRUE, the resending processing of Spool data is executed if necessary.
OperatingEnd	BOOL	FALSE	This variable changes to TRUE when the resending processing of Spool data is completed.
RS_instance	RS	---	Instance of RS instruction

Name	Data type	Default	Comment
MyStatus	_sDBC_CONNECTION_STATUS	---	This variable is assigned to the ConnectionStatus output variable from DB_GetConnectionStatus_instance.
Resend	BOOL	FALSE	This variable changes to TRUE when the status of the DB Connection is "Connected".
Nosent	BOOL	FALSE	This variable changes to TRUE when the status of the DB Connection is not "Connected".
ControlSpool_OK	BOOL	FALSE	This variable changes to TRUE when the DB_ControlSpool instruction is completed normally.

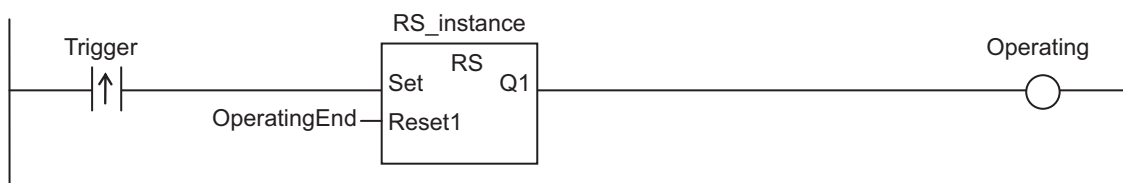
● Sample Programming

- Resend the SQL statements stored in the Spool memory when the status of the DB Connection is "Connected".

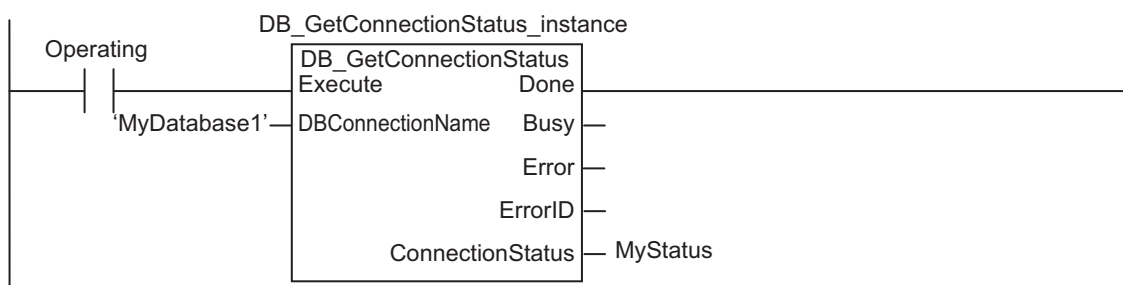
Check the completion of the instruction.



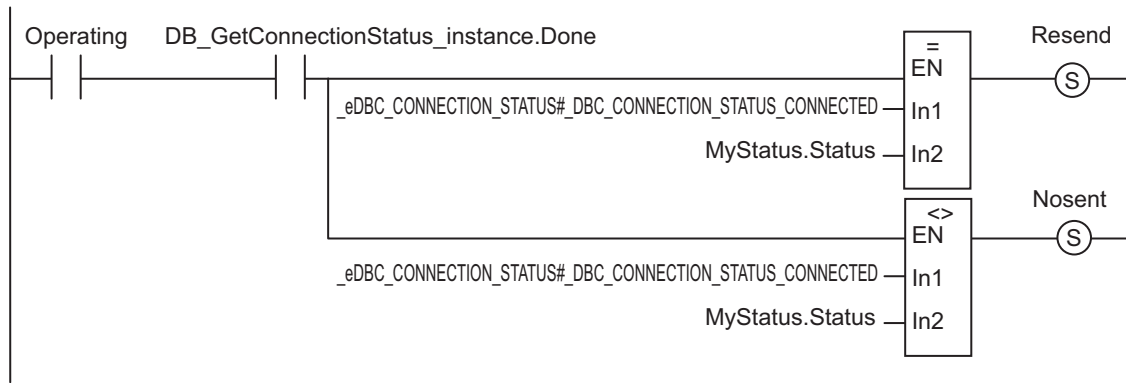
Accept the trigger.



Get the status of the DB Connection.

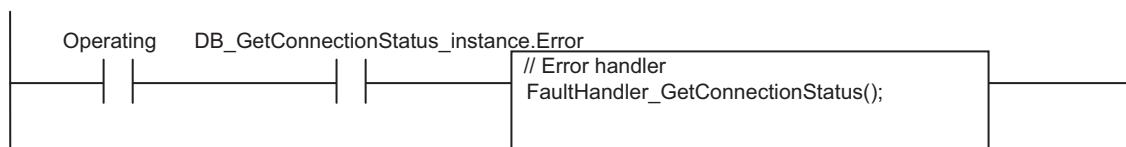


When the instruction is normally completed, change the Resend variable to TRUE if the status of the DB Connection is "Connected".

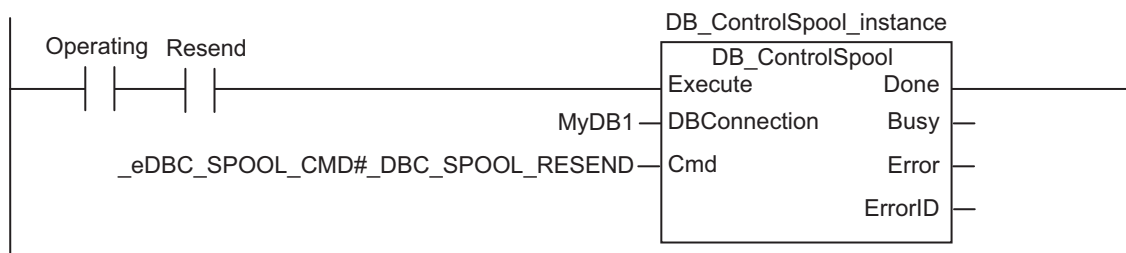


When the instruction is terminated due to an error, execute the error handler for the device (FaultHandler_GetConnectionStatus).

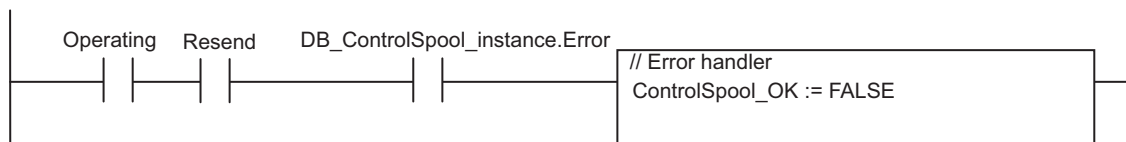
Program the FaultHandler_GetConnectionStatus according to the device.



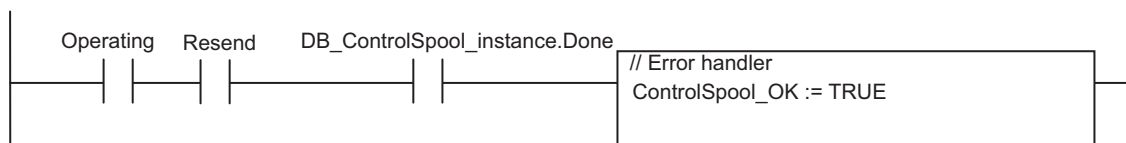
Resend the Spool data.



When the instruction is terminated due to an error, change the variable ControlSpool_OK to FALSE.



When the instruction is normally completed, change the variable ControlSpool_OK to TRUE.



Structured Text (ST)

● Main Variables

Name	Data type	Default	Comment
DB_GetConnectionStatus_instance	DB_GetConnectionStatus	---	Instance of DB_GetConnectionStatus instruction
DB_ControlSpool_instance	DB_ControlSpool	---	Instance of DB_ControlSpool instruction
Trigger	BOOL	FALSE	Variable used as a trigger for resending the Spool data

Name	Data type	Default	Comment
LastTrigger	BOOL	FALSE	Variable to retain the trigger status of the previous execution
Operating	BOOL	FALSE	When this variable is TRUE, the resending processing of Spool data is executed if necessary.
OperatingStart	BOOL	FALSE	The initialization processing is executed when this variable is TRUE.
Resend	BOOL	FALSE	This variable changes to TRUE when the status of the DB Connection is "Connected".
MyStatus	_sDBC_CONNECTION_STATUS	---	This variable is assigned to the ConnectionStatus output variable from DB_GetConnectionStatus_instance.
MyDB1	DWORD	---	This variable is assigned to the DBConnection input variable to DB_ControlSpool_instance.

● Sample Programming

```

(* -----
- Resend the SQL statements stored in the Spool memory when the status of the DB
Connection is Connected.
----- *)

// Start the sequence when the Trigger variable changes to TRUE.
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) ) THEN
    OperatingStart := TRUE;
    Operating := TRUE;
END_IF;
LastTrigger := Trigger;

// Sequence start processing
IF (OperatingStart=TRUE) THEN
    // Initialize the instruction instance.
    DB_GetConnectionStatus_instance( Execute:=FALSE );
    DB_ControlSpool_instance( Execute:=FALSE );
    OperatingStart := FALSE;
END_IF;

IF (Operating=TRUE) THEN
    // Get the status of the DB Connection.
    DB_GetConnectionStatus_instance(
        Execute           := TRUE,
        DBConnectionName  := 'MyDatabase1',
        ConnectionStatus  => MyStatus
    );

IF (DB_GetConnectionStatus_instance.Done=TRUE) THEN
    // Normal end processing
    // Change the variable Resend to TRUE when the status of the DB Connection is C
onnected.
    IF (MyStatus.Status = _DBC_CONNECTION_STATUS_CONNECTED) THEN

```

```

        Resend := TRUE;
    ELSE
        Resend := FALSE;
        Operating := FALSE;
    END_IF;
END_IF;
IF (DB_GetConnectionStatus_instance.Error=TRUE) THEN
    // Error handler
    Operating := FALSE;
END_IF;
END_IF;

IF ( (Operating=TRUE) AND (Resend=TRUE) ) THEN
    // Resend the Spool data.
    DB_ControlSpool_instance(
        Execute      := TRUE,
        DBConnection := MyDB1,
        Cmd          := _DBC_SPOOL_RESEND
    );

    IF (DB_ControlSpool_instance.Done=TRUE) THEN
        // Normal end processing
        Resend := FALSE;
        Operating := FALSE;
    END_IF;
    IF (DB_ControlSpool_instance.Error=TRUE) THEN
        // Error handler
        Resend := FALSE;
        Operating := FALSE;
    END_IF;
END_IF;

```

DB_PutLog (Record Operation Log)

The DB_PutLog instruction puts a user-specified record into the Execution Log or Debug Log.

Instruction	Name	FB/FUN	Graphic expression	ST expression
DB_PutLog	Record Operation Log	FB		DB_PutLog_instance (Execute, LogType, LogCode, LogName, LogMsg, Done, Busy, Error, ErrorID);

Note The DB_PutLog_instance is an instance of DB_PutLog instruction, which is declared as a variable.

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.
LogType	Log Type	_eDBC_LOGTYPE	_DBC_LOGTYPE_EXECUTION(1): Execution Log _DBC_LOGTYPE_DEBUG(2): Debug Log		0	Specify the type of log to output
LogCode	Log Code	INT	0 to 9999	---	0	Specify the code to record in the log.
LogName	Log Name	STRING	33 bytes max. (including the final NULL character)	---	"	Specify the name to record in the log.
LogMsg	Log Message	STRING	129 bytes max. (including the final NULL character)	---	"	Specify the message to record in the log.

Output Variable

Name	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.

Name	Meaning	Data type	Valid range	Unit	Description
Error	Error	BOOL	TRUE or FALSE		TRUE when the instruction is terminated due to an error.
ErrorID	Error Code	WORD	16#0000 to 16#FFFF	---	Contains the error code when an error occurs.

Related Error Codes

Error code	Meaning	Description
0400 hex	Input Value Out of Range	A value that is not defined as an enumerator was specified in the <i>LogType</i> input variable.
041D hex	Too Many Instructions Executed at the Same Time	More than 32 DB Connection Instructions were executed at the same time.
1400 hex	SD Memory Card Access Failure	The SD Memory Card is not available.
1401 hex	SD Memory Card Write-protected	The SD Memory Card is write-protected.
3002 hex	DB Connection Service Shut-down or Shutting Down	The instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
3010 hex	Log Code Out of Range	The value of the <i>LogCode</i> input variable is outside the valid range.
3013 hex	DB Connection Service Error Stop	The instruction was executed while the DB Connection Service was stopped due to an error.
3015 hex	DB Connection Service Initializing	The instruction was executed while the initialization processing of the DB Connection Service was in progress.
3017 hex	Operation Log Disabled	The log cannot be recorded because the specified Operation Log is disabled.

Function

This instruction is used to put a user-specified record into the Execution Log or Debug Log. Specify whether to record in the Execution Log or Debug Log in the *LogType* input variable. You can record any log code and log message into an Operation Log by specifying the *LogCode* and *LogMsg* input variables in the user program.

The log record format is shown below.

[Serial number]<tab>[Time]<tab>[Category]<tab>[Code]<tab>[Log name]<tab>[Result]<tab>[Details]<CR><LF>

[Serial number]:	A serial number from 0 to 65535. The value returns to 0 after 65535.
[Time]:	Time when the instruction is executed.
[Category]:	Always "USER"
[Code]:	Value of log code specified in the <i>LogCode</i> input variable Nothing is output for a text string consisting of NULL characters (16#00) only.
[Log name]:	Text string of log name specified in the <i>LogName</i> input variable Nothing is output for a text string consisting of NULL characters (16#00) only.
[Result]:	Always "0x0000"
[Details]:	Text string of log message specified in the <i>LogMsg</i> input variable

Precautions for Correct Use

- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual* (Cat. No. W502) for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- This instruction cannot be used on an event task. A compiling error will occur.
- When this instruction is executed during replacement of the SD Memory Card, the following operations are performed.

When the Execution Log is specified:

- a) The log is recorded to the internal buffer of the CPU Unit and the instruction is completed normally.
- b) When an SD Memory Card is inserted into the CPU Unit, the log records stored in the internal buffer are saved into the SD Memory Card.

When the Debug Log is specified:

- a) The Debug Log cannot be recorded. The instruction is terminated due to an error (Operation Log Disabled).
- An error occurs for this instruction in the following cases. *Error* will be TRUE.
 - a) When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
 - b) When the instruction was executed while the DB Connection Service was stopped due to an error.
 - c) When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
 - d) A value that is not defined as an enumerator was specified in the *LogType* input variable.
 - e) The value of the *LogCode* input variable is outside the valid range.
 - f) When the SD Memory Card is not available or write-protected
 - g) The log cannot be recorded because the specified Operation Log is disabled.
 - h) When more than 32 DB Connection Instructions were executed at the same time.

Sample Programming

This section gives sample programming for putting the following log record into the Execution Log when the trigger variable changes to TRUE.

- Log code: 100
- Log name: "Production Order"
- Log message: "Production Start, RecipeCode=12345678"

Ladder Diagram

● Main Variables

Name	Data type	Default	Comment
DB_PutLog_instance	DB_PutLog	---	Instance of DB_PutLog instruction
Trigger	BOOL	FALSE	Variable used as a trigger for recording the user-specified log

Name	Data type	Default	Comment
Operating	BOOL	FALSE	When this variable is TRUE, recording of the user-specified log is executed.
OperatingEnd	BOOL	FALSE	This variable changes to TRUE when recording of the user-specified log is completed.
RS_instance	RS	---	Instance of RS instruction
RecipeCode	UDINT	1234678	Recipe code used in the log message.
Msg	STRING[256]	"	Log message to record
PutLog_OK	BOOL	FALSE	This variable changes to TRUE when the DB_PutLog instruction is completed normally.

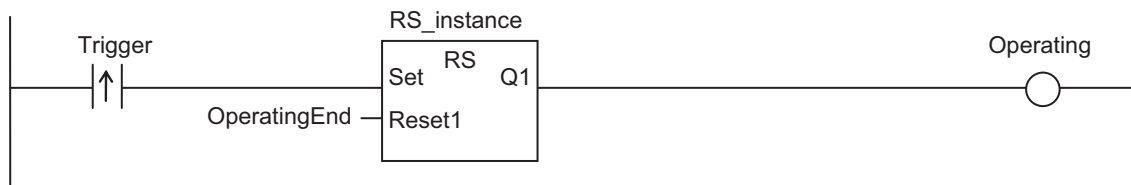
● Sample Programming

Record the log code 100, log name "Production Order", and log message "Production Start, RecipeCode=12345678" into the Execution Log.

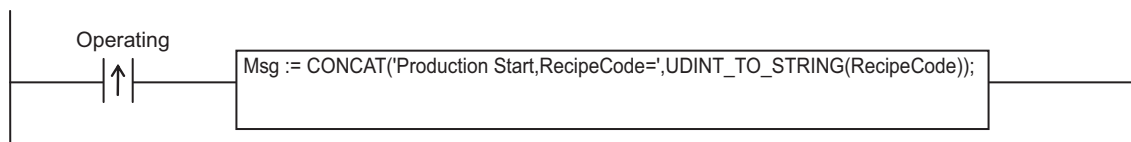
Check the completion of the DB_PutLog instruction.



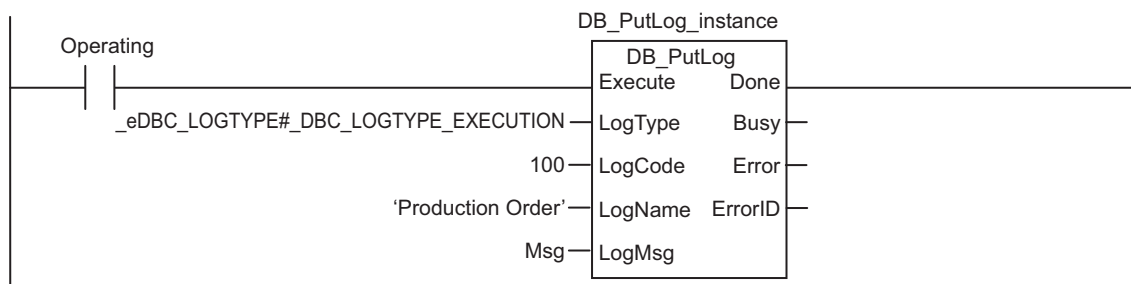
Accept the trigger.



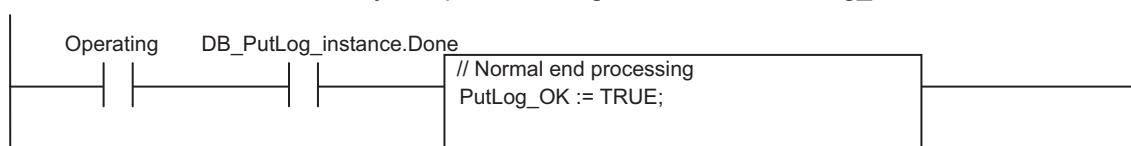
Create the log message.



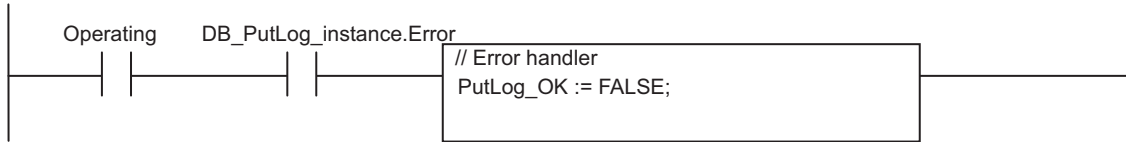
Record the log message into the Execution Log.



When the instruction is normally completed, change the variable PutLog_OK to TRUE.



When the instruction is terminated due to an error, change the variable PutLog_OK to FALSE.



Structured Text (ST)

● Main Variables

Name	Data type	Default	Comment
DB_PutLog_instance	DB_PutLog	---	Instance of DB_PutLog instruction
Trigger	BOOL	FALSE	Variable used as a trigger for recording the user-specified log
LastTrigger	BOOL	FALSE	Variable to retain the trigger status of the previous execution
Operating	BOOL	FALSE	When this variable is TRUE, recording of the user-specified log is executed.
OperatingStart	BOOL	FALSE	The initialization processing is executed when this variable is TRUE.
RecipeCode	UDINT	1234678	Recipe code used in the log message.
Msg	STRING[256]	"	Log message to record

● Sample Programming

```

(* -----
- Record the log code 100, log name Production Order, and log message Production
Start, RecipeCode=12345678 into the Execution Log.
----- *)

// Start the sequence when the variable Trigger changes to TRUE.
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) ) THEN
    OperatingStart := TRUE;
    Operating := TRUE;
END_IF;
LastTrigger := Trigger;

// Sequence start processing
IF (OperatingStart=TRUE) THEN
    // Initialize the instruction instance.
    DB_PutLog_instance( Execute:=FALSE );
    // Create the log message.
    Msg := CONCAT('Production Start,RecipeCode=',UDINT_TO_STRING(RecipeCode));

    OperatingStart := FALSE;
END_IF;

IF (Operating=TRUE) THEN
    // Record the log message into the Execution Log.
    DB_PutLog_instance(
        Execute := TRUE,

```

```

LogType      := _DBC_LOGTYPE_EXECUTION,
LogCode      := 100,
LogName      := 'Production Order',
LogMsg       := Msg );

IF (DB_PutLog_instance.Done=TRUE) THEN
    // Normal end processing
    Operating := FALSE;
END_IF;
IF (DB_PutLog_instance.Error=TRUE) THEN
    // Error handler
    Operating := FALSE;
END_IF;
END_IF;

```

DB_Shutdown (Shutdown DB Connection Service)

The DB_Shutdown instruction shuts down the DB Connection Service so as to prevent losing the Operation Log data.

Instruction	Name	FB/FUN	Graphic expression	ST expression
DB_Shutdown	Shutdown DB Connection Service	FB		DB_Shutdown_instance (Execute, Done, Busy, Error, ErrorID);

Note The DB_Shutdown_instance is an instance of DB_Shutdown instruction, which is declared as a variable.

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.

Output Variable

Name	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.
Error	Error	BOOL	TRUE or FALSE		TRUE when the instruction is terminated due to an error.
ErrorID	Error Code	WORD	16#0000 to 16#FFFF	---	Contains the error code when an error occurs.

Related System-defined Variables

Name	Meaning	Data type	Valid range	Description
_DBC_Status.Run	DB Connection Service Running Status	BOOL	TRUE or FALSE	This variable changes to FALSE when this instruction is executed.
_DBC_Status.Test	DB Connection Service Test Mode Status	BOOL	TRUE or FALSE	This variable changes to FALSE when this instruction is executed.

Name	Meaning	Data type	Valid range	Description
_DBC_Status.Shutdown	DB Connection Service Shutdown Status	BOOL	TRUE or FALSE	This variable changes to TRUE when this instruction is executed.

Related Error Codes

Error code	Meaning	Description
041D hex	Too Many Instructions Executed at the Same Time	More than 32 DB Connection Instructions were executed at the same time.
3001 hex	DB Connection Service Run Mode Change Failed	The instruction was executed while the stopping processing of the DB Connection Service was in progress.
3002 hex	DB Connection Service Shutdown or Shutting Down	The instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
3015 hex	DB Connection Service Initializing	The instruction was executed while the initialization processing of the DB Connection Service was in progress.

Function

This instruction is used to shut down the DB Connection Service.

Be sure to execute this instruction before turning OFF the power supply to the CPU Unit to prevent data loss of Operation Logs.

Precautions for Correct Use

- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual* (Cat. No. W502) for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- This instruction cannot be used on an event task. A compiling error will occur.
- The DB Connection Instructions cannot be executed during and after execution of this instruction. When a DB Connection Instruction is executed, it will be terminated due to an error.
- Be sure to execute this instruction before you turn OFF the power supply to the CPU Unit. If the power supply is turned OFF without executing this instruction, the Operation Log file may be corrupted or its contents may be lost.
- An error occurs for this instruction in the following cases. *Error* will be TRUE.
 - a) When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
 - b) The instruction was executed while the stopping processing of the DB Connection Service was in progress.
 - c) When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
 - d) When more than 32 DB Connection Instructions were executed at the same time.

Sample Programming

This section gives sample programming for shutting down the DB Connection Service when the trigger variable changes to TRUE.

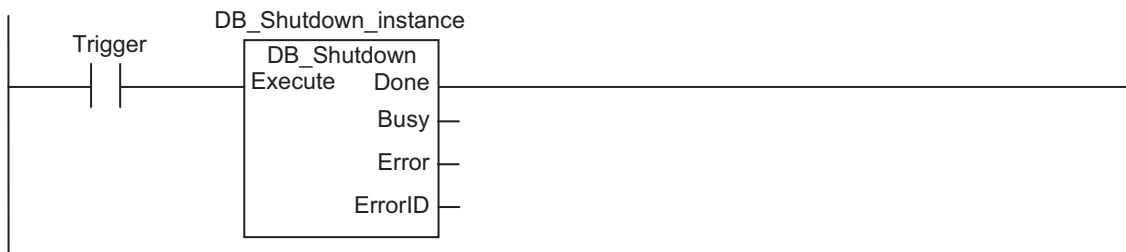
Ladder Diagram

● Main Variables

Name	Data type	Default	Comment
DB_Shutdown_instance	DB_Shutdown	---	Instance of DB_Shutdown instruction
Trigger	BOOL	FALSE	Variable used as a trigger for shutting down the DB Connection Service
Shutdown_OK	BOOL	FALSE	This variable changes to TRUE when the DB_Shutdown instruction is completed normally.

● Sample Programming

- Shut down the DB Connection Service.
- Shut down the DB Connection Service.



When the instruction is normally completed, change the variable Shutdown_OK to TRUE.



Structured Text (ST)

● Main Variables

Name	Data type	Default	Comment
DB_Shutdown_instance	DB_Shutdown	---	Instance of DB_Shutdown instruction
Trigger	BOOL	FALSE	Variable used as a trigger for shutting down the DB Connection Service
LastTrigger	BOOL	FALSE	Variable to retain the trigger status of the previous execution
Operating	BOOL	FALSE	Shutting down the DB Connection Service is executed when this variable is TRUE.
OperatingStart	BOOL	FALSE	The initialization processing is executed when this variable is TRUE.
ShutdownOK	BOOL	FALSE	This variable changes to TRUE when the DB_Shutdown instruction is completed normally.

● Sample Programming

```

(* -----
  ♦ Shut down the DB Connection Service.
  ----- *)

// Start the sequence when the variable Trigger changes to TRUE.
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) ) THEN
  OperatingStart := TRUE;
  Operating := TRUE;
END_IF;
LastTrigger := Trigger;

// Sequence start processing
IF (OperatingStart=TRUE) THEN
  // Initialize the instruction instance.
  DB_Shutdown_instance( Execute:=FALSE );

  OperatingStart := FALSE;
END_IF;

IF (Operating=TRUE) THEN
  // Shut down the DB Connection Service.
  DB_Shutdown_instance( Execute:=TRUE );

  IF (DB_Shutdown_instance.Done=TRUE) THEN
    // Normal end processing
    ShutdownOK := TRUE;
    Operating := FALSE;
  END_IF;
  IF (DB_Shutdown_instance.Error=TRUE) THEN
    // Error handler
    Operating := FALSE;
  END_IF;
END_IF;

```

DB_BatchInsert (DB Records Batch Insert)

The DB_Insert instruction collectively inserts values of array elements for a DB Map Variable into a database table as a single record.

Instruction	Name	FB/FUN	Graphic expression	ST expression
DB_BatchInsert	DB Records Batch Insert	FB		DB_BatchInsert_instance (Execute, DBConnection, MapVar, InsertCnt, SQLFailLog, QueryTimeOut, Done, Busy, Error, ErrorID, SendStatus) ;

Note The DB_BatchInsert_instance is an instance of DB_BatchInsert instruction, which is declared as a variable.

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.
DBConnection	DB Connection	DWORD	16#0 to FFFFFFFF	---	16#0	Specify the DB connection established by a DB_Connect instruction.
MapVar	DB Map Variable	Structure array (entire array)	Depends on the data type.	---	---	Specify the DB Map Variable mapped by a DB_CreateMapping instruction.
InsertCnt	Number of Inserted Records	DINT	0 to 65535	---	0	Records corresponding to the number of records specified in <i>InsertCnt</i> are inserted from the beginning of the structure array of the DB Map Variable <i>MapVar</i> .
SQLFailLog	SQL Execution Failure Log Output	BOOL	TRUE or FALSE	---	FALSE	Specify whether to output an SQL execution failure log.
QueryTimeOut	Query Execution Timeout Time	TIME	T#0s, T#1s to T#600s	---	T#0s	Specify the query execution timeout time. When T#0s is specified, it references the time specified in <i>Query Execution Timeout</i> in the DB Connection Settings.

Output Variable

Name	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.
Error	Error	BOOL	TRUE or FALSE	---	TRUE when the instruction is terminated due to an error.
ErrorID	Error code	WORD	16#0 to 16#FFFF	---	Contains the error code when an error occurs.
SendStatus	Send Status	_eDBC_SEND_STATUS	Depends on the data type.	---	Outputs the progress of transmission of the SQL statement.

Related System-defined Variables

Refer to *System-defined Variables Related to DB Connection Service* on page 7-5.

Related Error Codes

Error code	Meaning	Description
0400 hex	Input Value Out of Range	- The value of the <i>QueryTimeOut</i> input variable is outside the valid range. - The value of the <i>InsertCnt</i> input variable is outside the valid range.
041D hex	Too Many Instructions Executed at the Same Time	More than 32 DB Connection Instructions were executed at the same time.
3000 hex	DB Connection Service not Started	When the instruction was executed when the DB Connection Service was not running.
3002 hex	DB Connection Service Shutdown or Shutting Down	When the instruction was executed after the DB Connection Service has shut down or while the DB Connection Service was shutting down.
3008 hex	Invalid DB Connection	When the value of the <i>DBConnection</i> input variable is invalid or the specified DB Connection is already closed.
300A hex	DB Map Variable Unregistered	The variable specified in the <i>MapVar</i> input variable has not been mapped by a <i>DB_CreateMapping</i> instruction.
300B hex	SQL Execution Error	The executed SQL statement resulted in an error in the DB.
3011 hex	DB Connection Disconnected Error Status	The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
3013 hex	DB Connection Service Error Stop	When the instruction was executed while the DB Connection Service was stopped due to an error.
3014 hex	Data Already Spooled	This instruction cannot be executed because one or more SQL statements are already stored in the Spool memory.
3015 hex	DB Connection Service Initializing	When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
3016 hex	DB in Process	The instruction was executed before completion of the DB's processing for the DB Connection Instruction Execution Timeout that occurred for the previous <i>DB_Insert</i> , <i>DB_Update</i> , <i>DB_Select</i> , or <i>DB_Delete</i> instruction.
3019 hex	Instruction Executed for Unsupported Database Type	The instruction was executed for a database type that is not supported by this instruction.

Function

This instruction inserts values of the DB Map Variable *MapVar* into the table mapped by the *DB_CreateMapping* instruction as a batch record.

The *Spool* function is not available for the *DB_BatchInsert* instruction.

The following describes the relationship between the number of array elements in the DB Map Variable and the number of inserted records specified in the *InsertCnt* variable.

- Number of array elements in the DB Map Variable is equal to or less than the number of inserted records, or the number of inserted records is equal to 0:
Records are collectively inserted from the beginning until it reaches the maximum number of elements in the DB Map Variable.
- Number of array elements in the DB Map Variable > Number of inserted records:
Records corresponding to the number of inserted records specified in *InsertCnt* are inserted from the beginning.

To enable the SQL execution failure log, you need to set *SQL execution failure log* to *Record* and set *SQLFailLog* to TRUE in the DB Connection Service Settings.

The instruction execution timeout is not available for the *DB_BatchInsert* instruction. The *QueryTimeout* input variable is the timeout time for query execution. If a value other than 0 is set to the *QueryTimeout* input variable, the *QueryTimeout* input variable is enabled instead of the time specified in *Query Execution Timeout* in the DB Connection Settings. If the query execution timeout is reached, an instruction error (SQL Execution Error) occurs.

Precautions for Correct Use

- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual* (Cat. No. W502) for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- This instruction cannot be used on an event task. A compiling error will occur.
- If the values cannot be registered to the DB, for example, because the SQL statement is invalid, this instruction ends abnormally.
- When the DB Connection Service was started in Test Mode, this instruction is completed normally without executing the INSERT operation for the DB actually.
- When the error code is 300B hex (SQL Execution Error), you can get the detailed information of the SQL Execution Error by executing a *DB_GetConnectionStatus* instruction.
- When the SQL execution failure log is enabled, the execution time of the other processing may become longer. Before starting actual operation, you must test performance under all foreseeable conditions on the actual system and make sure that the DB Connection Instructions are executed within the appropriate execution time.
- If the batch insert of records does not complete within the specified timeout for query execution, an instruction error (SQL Execution Error) occurs. Refer to *A-2 Execution Time of DB Connection Instructions* on page A-15 and make sure to specify the time period considering the performance of the server where the database is placed, as well as the load fluctuation of the server.
- An error occurs for this instruction in the following cases. *Error* will be TRUE.

- a) When the instruction was executed when the DB Connection Service was not running.
- b) When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
- c) When the instruction was executed while the DB Connection Service was stopped due to an error.
- d) When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
- e) Start of the DB Connection Service was commanded while the DB Connection Service was being stopped.
- f) When the value of the *DBConnection* input variable is invalid or the specified DB Connection is already closed.
- g) The variable specified in the *MapVar* input variable has not been mapped by a DB_CreateMapping instruction.
- h) The executed SQL statement resulted in an error in the DB.
- i) The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
- j) When one or more SQL statements are already stored in the Spool memory.
- k) The instruction was not completed within the time specified for query execution timeout.
- l) The value of the *QueryTimeOut* input variable is outside the valid range.
- m) When the instruction was executed before completion of the DB's processing for the DB Connection Instruction Execution Timeout that occurred for the previous DB_Insert, DB_Update, DB_Select, or DB_Delete instruction.
- n) When more than 32 DB Connection Instructions were executed at the same time.

Sample Programming

This section gives sample programming for executing the DB records batch insert.

Structure Data Type Definition

The structure settings for the sample programming are specified below.

Name	Data type
PRODUCTION_BATCHINSERT	STRUCT
NAME	STRING[256]
LOTNO	STRING[32]
STATUS	STRING[8]
PRODUCTIONDATE	DATE

Ladder Diagram

● Main Variables

Name	Data type	Initial value	Comment
DB_Connect_instance	DB_Connect	---	Instance of the DB_Connect instruction

Name	Data type	Initial value	Comment
DB_CreateMapping_instance	DB_CreateMapping	---	Instance of the DB_CreateMapping instruction
DB_BatchInsert_instance	DB_BatchInsert	---	Instance of the DB_BatchInsert instruction
DB_Close_instance	DB_Close	---	Instance of the DB_Close instruction
MyDB1	DWORD	---	This variable is assigned to the DBConnection output variable from DB_Connect_instance.
Name	STRING[256]	'WORK001'	Production information: Product name
LotNo	UINT	1234	Production information: Lot number
Trigger_Connect	BOOL	---	Variable used as a trigger for establishing a DB Connection
Operating_Connect	BOOL	---	The DB_Connect instruction is executed when this variable is TRUE.
OperatingEnd_Connect	BOOL	---	This variable changes to TRUE when the DB_Connect instruction is completed.
RS_Connect_instance	RS	---	Instance of the RS instruction
MapVar_BatchInsert	ARRAY[0..99] OF PRODUCTION_BATCHINSERT		This variable is assigned to the MapVar input variable for an instance called DB_CreateMapping_instance of the DB_CreateMapping instruction.
Trigger_BatchInsert	BOOL	---	Variable used as a trigger for executing the DB records batch insert
Operating_BatchInsert	BOOL	---	The DB_BatchInsert instruction is executed when this variable is TRUE and Execute_BatchInsert is TRUE.
OperatingEnd_BatchInsert	BOOL	---	This variable changes to TRUE when the DB_BatchInsert instruction is completed.
RS_BatchInsert_instance	RS	---	Instance of the RS instruction
Trigger_Close	BOOL	---	Variable used as a trigger for closing the DB Connection
Operating_Close	BOOL	---	The DB_Close instruction is executed when this variable is TRUE.
OperatingEnd_Close	BOOL	---	This variable changes to TRUE when the DB_Close instruction is completed.
RS_Close_instance	RS	---	Instance of the RS instruction
Index	UINT	---	Variable representing a record number
TON_instance	TON	---	Instance of the TON instruction

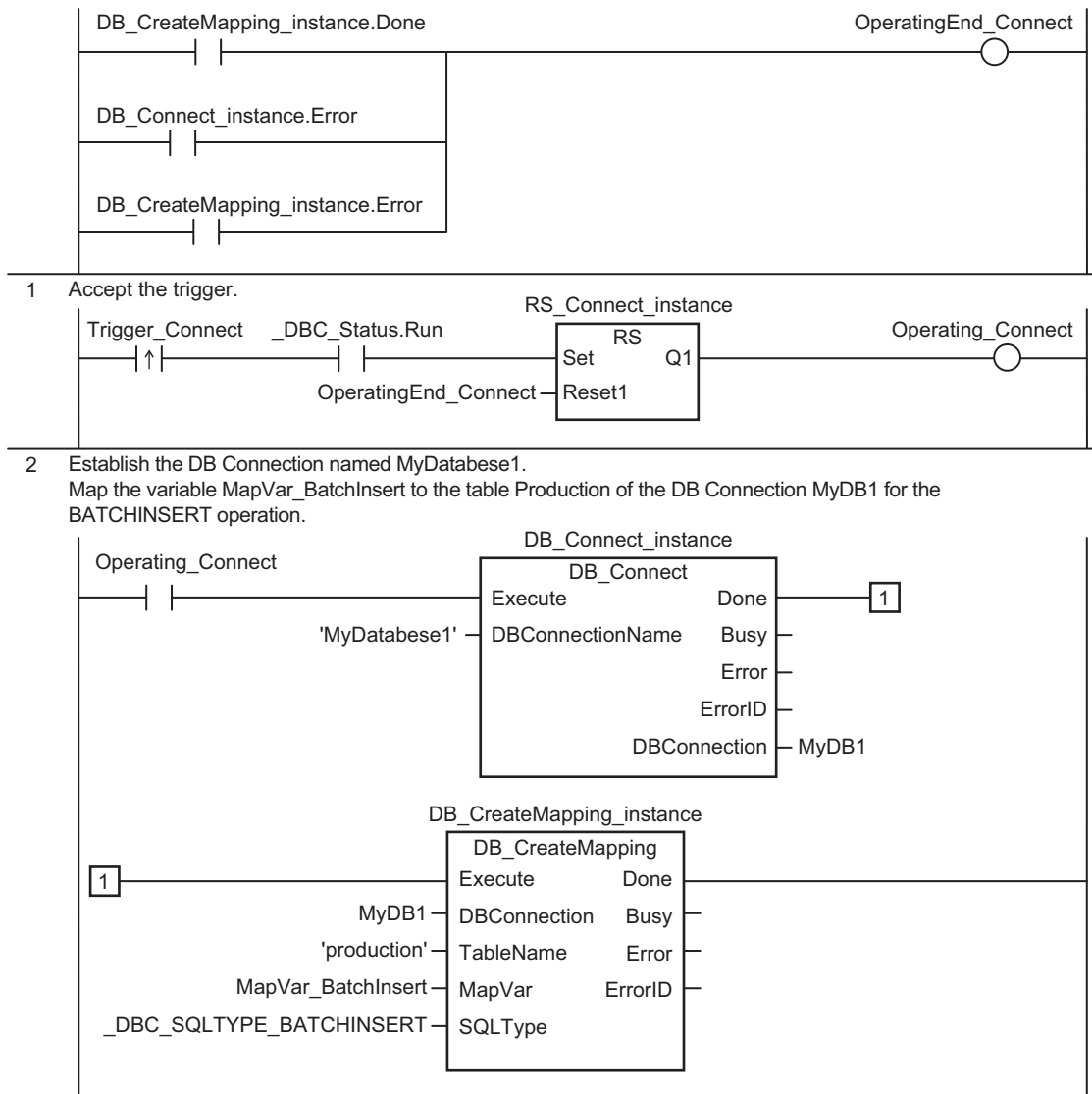
Name	Data type	Initial value	Comment
Execute_BatchInsert	BOOL	---	The DB_BatchInsert instruction is executed when this variable is TRUE.
RS_ExecuteBatchInsert_instance	RS	---	Instance of the RS instruction

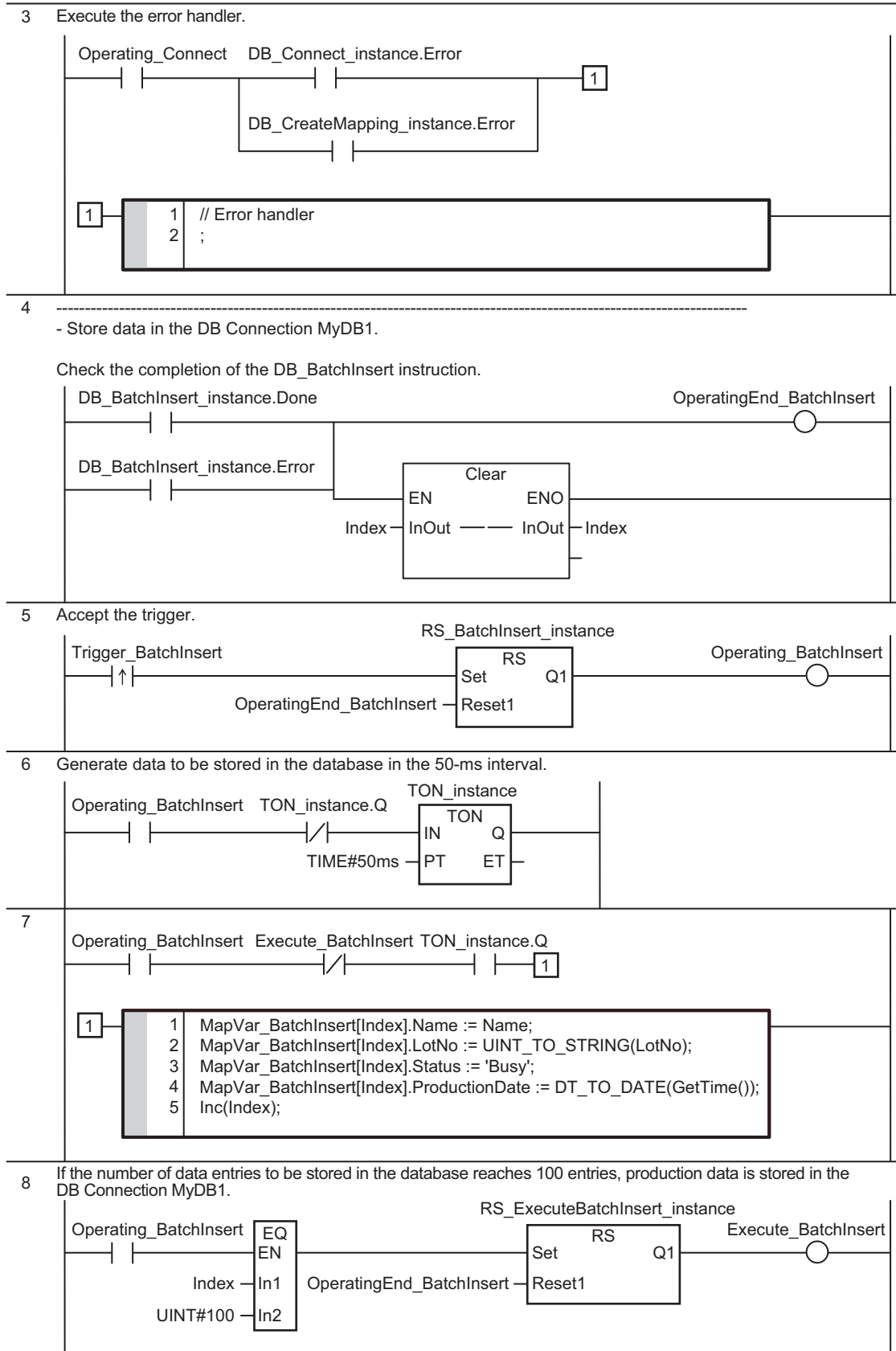
● Sample Programming

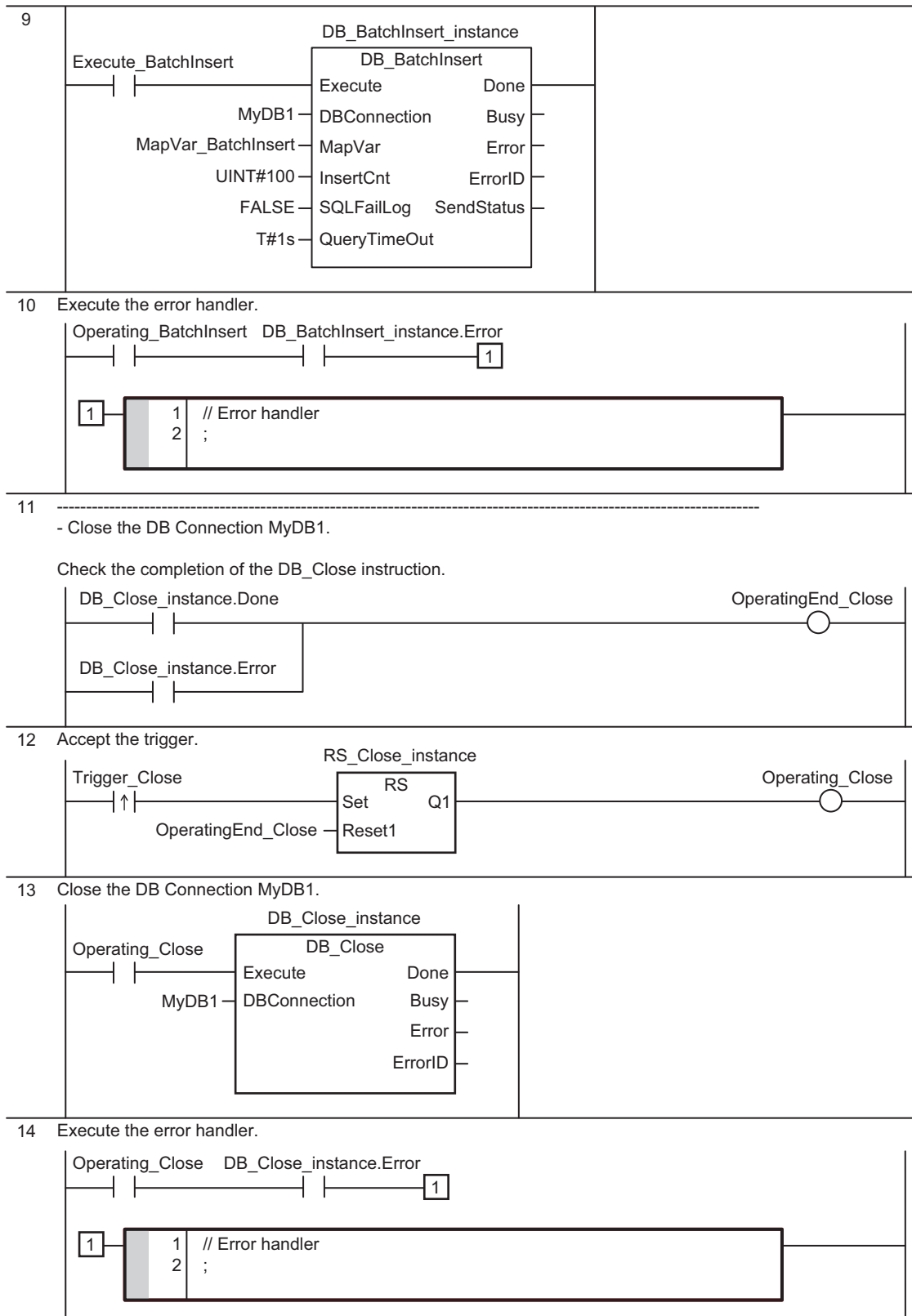
- 0 - This program is used for storing production data in the database.
The operation procedure is described below.
1. Use the DB_Connect instruction to establish connection with the database.
 2. Use the DB_CreateMapping instruction to map the database table with the variable.
 - 3-1. Prepare data to be stored in the database.
 - 3-2. Use the DB_BatchInsert instruction to store data in the database table.
 4. Use the DB_Close instruction to disconnect the database connection.

- Establish a DB Connection named MyDatabase1 and map a database table with a variable.

Check the completion of the DB_Connect and DB_CreateMapping instructions.







Structured Text (ST)

● Main Variables

Name	Data type	Initial value	Comment
DB_Connect_instance	DB_Connect	---	Instance of the DB_Connect instruction
DB_CreateMapping_instance	DB_CreateMapping	---	Instance of the DB_CreateMapping instruction
DB_BatchInsert_instance	DB_BatchInsert	---	Instance of the DB_BatchInsert instruction
DB_Close_instance	DB_Close	---	Instance of the DB_Close instruction
MyDB1	DWORD	---	This variable is assigned to the DBConnection output variable from DB_Connect_instance.
Name	STRING[256]	'WORK001'	Production information: Product name
LotNo	UINT	1234	Production information: Lot number
Trigger_Connect	BOOL	---	Variable used as a trigger for establishing a DB Connection
LastTrigger_Connect	BOOL	---	Variable to retain the trigger status of the previous execution
Operating_Connect	BOOL	---	The DB_Connect instruction is executed when this variable is TRUE.
OperatingStart_Connect	BOOL	---	The start processing for establishing a DB Connection is executed when this variable is TRUE.
MapVar_BatchInsert	ARRAY[0..99] OF PRODUCTION_BATCHINSERT		This variable is assigned to the MapVar input variable for an instance called DB_CreateMapping_instance of the DB_CreateMapping instruction.
Trigger_BatchInsert	BOOL	---	Variable used as a trigger for executing the DB records batch insert
LastTrigger_BatchInsert	BOOL	---	Variable to retain the trigger status of the previous execution
Operating_BatchInsert	BOOL	---	The DB_BatchInsert instruction is executed when this variable is TRUE and Execute_BatchInsert is TRUE.
Execute_BatchInsert	BOOL	---	The DB_BatchInsert instruction is executed when this variable is TRUE.
Trigger_Close	BOOL	---	Variable used as a trigger for closing the DB Connection

Name	Data type	Initial value	Comment
LastTrigger_Close	BOOL	---	Variable to retain the trigger status of the previous execution
Operating_Close	BOOL	---	The DB_Close instruction is executed when this variable is TRUE.
OperatingStart_Close	BOOL	---	The start processing for closing a DB Connection is executed when this variable is TRUE.
Stage	INT	---	Variable that shows the status of the DB Connection
Index	UINT	---	Variable representing a record number
TON_instance	TON	---	Instance of the TON instruction

● Sample Programming

```
(* -----
---
- This program is used for storing production data in the database.
The operation procedure is described below.
1. Use the DB_Connect instruction to establish connection with the database.
2. Use the DB_CreateMapping instruction to map the database table with the variable.
3-1. Prepare data to be stored in the database.
3-2. Use the DB_BatchInsert instruction to store data in the database table.
4. Use the DB_Close instruction to disconnect the database connection.
-----
--- *)

//-----
----
//- Establish a DB Connection named MyDatabase1 and map a database table with the variable.

// Start the sequence when Trigger_Connect changes to TRUE
IF ( (Trigger_Connect=TRUE) AND (LastTrigger_Connect=FALSE) AND (_DBC_Status.Run=TRUE) ) THEN
    OperatingStart_Connect := TRUE;
    Operating_Connect := TRUE;
END_IF;
LastTrigger_Connect:=Trigger_Connect;

// Sequence start processing
IF (OperatingStart_Connect=TRUE) THEN
    // Initialize the instances of the applicable DB Connection Instructions.
    DB_Connect_instance(Execute:=FALSE );
    DB_CreateMapping_instance(Execute := FALSE, MapVar:=MapVar_BatchInsert, SQL
```

```

Type:=_DBC_SQLTYPE_BATCHINSERT);
    Stage := INT#1;
    Index:=UINT#0;
    OperatingStart_Connect := FALSE;
END_IF;

// Establish the DB Connection named MyDatabase1.
// Map the variable MapVar_BatchInsert to the table Production of the DB Connection
MyDB1 for the BATCHINSERT operation.
IF (Operating_Connect=TRUE) THEN
    CASE Stage OF
        1 : // Establish the DB Connection
            DB_Connect_instance( Execute:=TRUE, DBConnectionName:='MyData
base1', DBConnection=>MyDB1 );

            IF (DB_Connect_instance.Done=TRUE) THEN
                Stage := INT#2; // Normal end
            ELSIF (DB_Connect_instance.Error=TRUE) THEN
                Stage := INT#99; // Error
            END_IF;

        2 : // Map the DB table with the variable
            DB_CreateMapping_instance( Execute := TRUE, DBConnection:=MyD
B1, TableName:='Production', MapVar:=MapVar_BatchInsert, SQLType:=_DBC_SQLTYPE_BATC
HINSERT);

            IF ( DB_CreateMapping_instance.Done=TRUE) THEN
                Operating_Connect:=FALSE; // Normal end
            ELSIF ( DB_CreateMapping_instance.Error=TRUE ) THEN
                Stage := INT#99; // Error
            END_IF;

        99 :
            // Error handler
            Operating_Connect := FALSE;
    END_CASE;
END_IF;

//-----
//-- Store data in the DB Connection named MyDB1.

// Start the sequence when Trigger_BatchInsert changes to TRUE
IF ( (Trigger_BatchInsert=TRUE) AND (LastTrigger_BatchInsert=FALSE) ) THEN
    Operating_BatchInsert := TRUE;
    // Initialize the instance of the applicable DB Connection Instructions.
    DB_BatchInsert_instance( Execute:=FALSE, MapVar:=MapVar_BatchInsert );

```

```

END_IF;
LastTrigger_BatchInsert := Trigger_BatchInsert;

// Generate data to be stored in the database in the 50-ms interval.
TON_instance( In:=NOT(TON_instance.Q), PT:=TIME#50ms );

IF( (Operating_BatchInsert=TRUE) AND (Execute_BatchInsert=FALSE) AND (TON_instance.
Q=TRUE)) THEN
    MapVar_BatchInsert[Index].Name := Name;
    MapVar_BatchInsert[Index].LotNo := UINT_TO_STRING(LotNo);
    MapVar_BatchInsert[Index].Status := 'Busy';
    MapVar_BatchInsert[Index].ProductionDate := DT_TO_DATE(GetTime( ));
    Inc(Index);
END_IF;

// If the number of data entries to be stored in the database reaches 100 entries,
production data is stored in the DB Connection named MyDB1.
IF( (Operating_BatchInsert=TRUE) AND (Index=UINT#100) ) THEN
    Execute_BatchInsert:=TRUE;
    Index:=0;
END_IF;

IF( (Operating_BatchInsert=TRUE) AND (Execute_BatchInsert=TRUE)) THEN
    DB_BatchInsert_instance(Execute:=TRUE, DBConnection:=MyDB1, MapVar:=MapVar_
BatchInsert, InsertCnt:=UINT#100, SQLFailLog:=FALSE, QueryTimeOut:=T#1s);
    IF (DB_BatchInsert_instance.Done=TRUE) THEN
        Execute_BatchInsert:=FALSE; // Normal end
        Operating_BatchInsert := FALSE;
    ELSIF (DB_BatchInsert_instance.Error=TRUE) THEN
        // Error handler
        Execute_BatchInsert:=FALSE;
        Operating_BatchInsert := FALSE;
    END_IF;
END_IF;

//-----
//-- Close the DB Connection MyDB1.

// Start the sequence when Trigger_Close changes to TRUE
IF ( (Trigger_Close=TRUE) AND (LastTrigger_Close=FALSE) ) THEN
    OperatingStart_Close := TRUE;
    Operating_Close := TRUE;
END_IF;
LastTrigger_Close := Trigger_Close;

// Sequence start processing

```

```
IF (OperatingStart_Close=TRUE) THEN
    // Initialize the instance of the applicable DB Connection Instructions.
    DB_Close_instance(Execute:=FALSE );
    OperatingStart_Close := FALSE;
END_IF;

// Close the DB Connection MyDB1.
IF (Operating_Close=TRUE) THEN
    // Close the DB Connection.
    DB_Close_instance(Execute:=TRUE, DBConnection:=MyDB1 );

    IF (DB_Close_instance.Done=TRUE) THEN
        Operating_Close := FALSE; // Normal end
    ELSIF (DB_Close_instance.Error=TRUE) THEN
        // Error handler
        Operating_Close := FALSE;
    END_IF;
END_IF;
```

DB_AttachProcedure (Generate DB Stored Procedure Handle)

The DB_AttachProcedure instruction obtains a procedure handle used for calling a stored procedure of the database.

Instruction	Name	FB/FUN	Graphic expression	ST expression
DB_Attach-Procedure	Generate DB Stored Procedure Handle	FB		DB_AttachProcedure_instance (Execute, DBConnection, ProcName, ArgIn, ArgOut, ArgInOut, ReturnVal, ResultSet, Done, Busy, Error, ErrorID, ProcHandle) ;

Note The DB_AttachProcedure_instance is an instance of DB_AttachProcedure instruction, which is declared as a variable.

DB_AttachProcedure (Generate DB Stored Procedure Handle)

7

Variables

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.
DBConnec-tion	DB Connection	DWORD	16#0 to FFFFFFFF	---	16#0	Specify the DB connection established by a DB_Connect instruction.
ProcName	Stored Procedure Name	STRING	Depends on the data type.	---	"	DB procedure or function name
ArgIn	IN Argument of Stored Procedure	Structure	Depends on the data type.	---	---	Variable (IN) associated with the stored procedure's argument
ArgOut	OUT Argument of Stored Procedure	Structure	Depends on the data type.	---	---	Variable (OUT) associated with the stored procedure's argument
ArgInOut	INOUT Argument of Stored Procedure	Structure	Depends on the data type.	---	---	Variable (INOUT) associated with the stored procedure's argument

Name	Meaning	Data type	Valid range	Unit	Default	Description
RetVal	Return Value of Stored Procedure	Basic type (excluding structure, union and enum)	Depends on the data type.	---	---	Variable associated with the stored procedure's return value
ResultSet	Result Set of Stored Procedure	Structure and structure array (entire array)	Depends on the data type.	---	---	Variable associated with the stored procedure's result set

Output Variable

Name	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.
Error	Error	BOOL	TRUE or FALSE	---	TRUE when the instruction is terminated due to an error.
ErrorID	Error code	WORD	16#0 to 16#FFFF	---	Contains the error code when an error occurs.
ProcHandle	Procedure Handle	DWORD	16#0 to FFFFFFFF	---	Handle for calling a stored procedure using a DB Connection Instruction.

Related System-defined Variables

Refer to *System-defined Variables Related to DB Connection Service* on page 7-5.

Related Error Codes

Error code	Meaning	Description
0406 hex	Illegal Data Position Specified	When the <i>ProcName</i> input variable is a text string consisting of NULL characters (16#00) only.
0410 hex	Text String Format Error	A space character is included in the text string specified for the <i>ProcName</i> input variable. When the <i>ProcName</i> input variable does not end in NULL.
041B hex	Data Capacity Exceeded	When the number of retrieved procedure handles exceeds the maximum number of DB Map Variables that are allowed.
041D hex	Too Many Instructions Executed at the Same Time	When more than 32 DB Connection Instructions were executed at the same time.
3000 hex	DB Connection Service not Started	When the instruction was executed when the DB Connection Service was not running.
3002 hex	DB Connection Service Shutdown or Shutting Down	When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
3008 hex	Invalid DB Connection	When the value of the <i>DBConnection</i> input variable is invalid or the specified DB Connection is already closed.

Error code	Meaning	Description
3009 hex	Invalid DB Map Variable	When the data type of the variables specified for <i>ArgIn</i> , <i>ArgOut</i> , <i>ArgInOut</i> , or <i>ResultSet</i> is not a structure When the structure members of the variable specified for <i>ArgIn</i> , <i>ArgOut</i> , <i>ArgInOut</i> , or <i>ResultSet</i> contain derivative type data When a structure array variable is specified for <i>ArgIn</i> , <i>ArgOut</i> , or <i>ArgInOut</i> When non-basic type data is specified for <i>ReturnVal</i>
3011 hex	DB Connection Disconnected Error Status	The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
3013 hex	DB Connection Service Error Stop	When the instruction was executed while the DB Connection Service was stopped due to an error.
3015 hex	DB Connection Service Initializing	When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
3019 hex	Instruction Executed for Unsupported Database Type	The instruction was executed for a database type that is not supported by this instruction.
301A hex	Invalid Stored Procedure Name	The specified stored procedure name does not exist. Note This includes when the specified stored procedure name does not find on the DB. Even if the stored procedure name exists, the DB Connection Service cannot find the stored procedure name due to the reason that the user does not have the access right to the stored procedure, or other reasons.
301B hex	Invalid Stored Procedure Argument	The attached argument information does not match the argument of the stored procedure.

Function

The `DB_AttachProcedure` instruction obtains a procedure handle used for calling a stored procedure of the database.

The obtained procedure handle *ProcHandle* is constrained by the maximum number of DB Map Variables that can be mapped. For example, if fourteen *MapVar* (DB Map Variables) are used when the maximum number is 15^{*1} , only one *ProcHandle* (Procedure Handle) can be obtained by the `DB_AttachProcedure` instruction.

*1. Refer to *1-2-1 DB Connection Service Specifications* on page 1-5 for the maximum number of the DB Map Variables supported by each model.

Use the variable *ProcName* to specify the name of a stored procedure you want to call.

For the variables *ArgIn*, *ArgOut*, *ArgInOut*, *ReturnVal*, and *ResultSet*, specify the corresponding arguments, return value, and result set of the stored procedure.

Associate the stored procedure specified in *ProcName* with the DB Map Variables specified in *ArgIn*, *ArgOut*, *ArgInOut*, *ReturnVal*, and *ResultSet*. In addition, associate it with a DB Connection.

This function retrieves metadata of the stored procedure specified in *ProcName* and checks the stored procedure's interface (arguments, return value, and result set).

When you use the Operation Logs, you can check an error that occurs during execution.

[Arguments of stored procedure]

Make sure that the procedure's arguments match the name, data type, and the number of arguments specified in *ArgIn/ArgOut/ArgInOut*.

If INOUT/OUT exists for the attribute of the procedure's arguments, the values are modified to the structure variable specified in *ArgOut/ArgInOut* after the *DB_ExecuteProcedure* instruction is executed.

To omit the arguments of the stored procedure, assign the system-defined variable *_DBC_Unused* for the corresponding *ArgIn/ArgOut/ArgInOut* input variables.*¹

Refer to 5-3-2 *Specifications of the Stored Procedure Call Function for Databases* on page 5-17 for specifications of the stored procedure call function for each database type.

[Return value of stored procedure]

Make sure that the procedure's return value matches the name and data type specified in *RetVal*.

To omit the return value of the stored procedure, assign the system-defined variable *_DBC_Unused* for the corresponding *RetVal* input variable.*¹

[Result set of stored procedure]

Make sure that the procedure's result set match the name, data type, and the number of arguments specified in *ResultSet*.

To retrieve multiple records, define the result set as an array.

To omit the result set of the stored procedure, assign the system-defined variable *_DBC_Unused* for the corresponding *ResultSet* input variable.*¹

Refer to 5-3-2 *Specifications of the Stored Procedure Call Function for Databases* on page 5-17 for specifications of the stored procedure call function for each database type.

*1. The execution result of the instruction is the same whether the *_DBC_Unused* value is set to either TRUE or FALSE.

Precautions for Correct Use

- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual* (Cat. No. W502) for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- This instruction cannot be used on an event task. A compiling error will occur.
- When the DB Connection Service was started in Test Mode, this instruction ends normally.
- An error occurs for this instruction in the following cases. *Error* will be TRUE.
 - a) When the instruction was executed when the DB Connection Service was not running.
 - b) When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
 - c) When the instruction was executed while the DB Connection Service was stopped due to an error.
 - d) When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
 - e) When the value of the *DBConnection* input variable is invalid or the specified DB Connection is already closed.
 - f) When the *ProcName* input variable is a text string consisting of NULL characters (16#00) only.
 - g) A space character is included in the text string specified for the *ProcName* input variable. When the *ProcName* input variable does not end in NULL.

- h) The maximum number of DB Map Variables for which a mapping can be created is exceeded.
When the number of retrieved procedure handles exceeds the maximum number of DB Map Variables that are allowed.
- i) When the data type of the variables specified for *ArgIn*, *ArgOut*, *ArgInOut*, or *ResultSet* is not a structure.
When the structure members of the variable specified for *ArgIn*, *ArgOut*, *ArgInOut*, or *ResultSet* contain derivative type data.
When a structure array variable is specified for *ArgIn*, *ArgOut*, or *ArgInOut*.
When non-basic type data is specified for *ReturnVal*.
- j) The specified stored procedure name does not exist.
- k) The attached argument information does not match the argument of the stored procedure.
- l) The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
- m) The instruction was executed for a database type that is not supported by this instruction.
- n) When more than 32 DB Connection Instructions were executed at the same time.

Sample Programming

Refer to *Sample Programming* on page 7-115 for the sample programming that is provided for the *DB_ExecuteProcedure* instruction.

DB_ExecuteProcedure (Execute DB Stored Procedure)

The DB_ExecuteProcedure instruction calls a stored procedure using the procedure handle obtained by a DB_AttachProcedure instruction.

Instruction	Name	FB/FUN	Graphic expression	ST expression
DB_ExecuteProcedure	Execute DB Stored Procedure	FB		DB_ExecuteProcedure_instance (Execute, DBConnection, ProcHandle, QueryTimeOut, Done, Busy, Error, ErrorID, RecCnt, Overflow, SendStatus) ;

Note The DB_ExecuteProcedure_instance is an instance of DB_ExecuteProcedure instruction, which is declared as a variable.

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.
DBConnection	DB Connection	DWORD	16#0 to FFFFFFFF	---	16#0	Specify the DB connection established by a DB_AttachProcedure instruction.
ProcHandle	Procedure Handle	DWORD	16#0 to FFFFFFFF	---	16#0	Procedure handle obtained by the DB_AttachProcedure instruction.
QueryTimeOut	Query Execution Timeout Time	TIME	T#0s, T#1s to T#600s	---	T#0s	Specify the query execution timeout time. When T#0s is specified, it references the time specified in <i>Query Execution Timeout</i> in the DB Connection Settings.

Output Variable

Name	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.

Name	Meaning	Data type	Valid range	Unit	Description
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.
Error	Error	BOOL	TRUE or FALSE	---	TRUE when the instruction is terminated due to an error.
ErrorID	Error Code	WORD	16#0 to 16#FFFF	---	Contains the error code when an error occurs.
RecCnt	Number of Records	DINT	0 to 65535	---	Displays the number of records stored in the result set variable <i>ResultSet</i> .
Overflow	Number of Overflowed Records	BOOL	TRUE or FALSE	---	Indicates that the number of records extracted in the result set exceeded the number of elements in the structure array variable <i>ResultSet</i> .
SendStatus	Send Status	_eDBC_SEND_STATUS	Depends on the data type.	---	Outputs the progress of transmission of the SQL statement.

Related System-defined Variables

Refer to *System-defined Variables Related to DB Connection Service* on page 7-5.

Related Error Codes

Error code	Meaning	Description
0400 hex	Input Value Out of Range	The value of the <i>QueryTimeOut</i> input variable is outside the valid range.
041D hex	Too Many Instructions Executed at the Same Time	More than 32 DB Connection Instructions were executed at the same time.
3000 hex	DB Connection Service not Started	When the instruction was executed when the DB Connection Service was not running.
3002 hex	DB Connection Service Shutdown or Shutting Down	When the instruction was executed after the DB Connection Service has shut down or while the DB Connection Service was shutting down.
3008 hex	Invalid DB Connection	When the value of the <i>DBConnection</i> input variable is invalid or the specified DB Connection is already closed.
300B hex	SQL Execution Error	The executed SQL statement resulted in an error in the DB.
3011 hex	DB Connection Disconnect- ed Error Status	The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
3013 hex	DB Connection Service Error Stop	When the instruction was executed while the DB Connection Service was stopped due to an error.
3014 hex	Data Already Spooled	This instruction cannot be executed because one or more SQL statements are already stored in the Spool memory.
3015 hex	DB Connection Service Initializing	When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
3016 hex	DB in Process	The instruction was executed before completion of the DB's processing for the DB Connection Instruction Execution Timeout that occurred for the previous <i>DB_Insert</i> , <i>DB_Update</i> , <i>DB_Select</i> , or <i>DB_Delete</i> instruction.
3018 hex	Invalid Stored Procedure Handle	The <i>ProcHandle</i> variable obtained by the <i>DB_AttachProcedure</i> instruction is not specified. The <i>ProcHandle</i> variable released by the <i>DB_DetachProcedure</i> is specified. The <i>ProcHandle</i> variable obtained in other than the relevant connection is specified.

Error code	Meaning	Description
3019 hex	Instruction Executed for Unsupported Database Type	The instruction was executed for a database type that is not supported by this instruction.
301C hex	Invalid Number of Columns for Stored Procedure Result Set	The number of columns in the retrieved result set is not consistent with the number of members in the structure variable <i>ResultSet</i> where the result is stored.

Function

This instruction calls a stored procedure using the procedure handle obtained by the `DB_AttachProcedure` instruction.

To call a stored procedure, the variables specified for the `DB_AttachProcedure` instruction are applied to the stored procedure's argument, return value, and result set. The `Spool` function is not available for the `DB_ExecuteProcedure` instruction.

After executing the `DB_ExecuteProcedure` instruction, the number of stored records is output to *RecCnt*, and whether the data fits in the array is output to *Overflow*. If the data did not fit into the array, the records are retrieved up to the maximum number of elements in the array.

Depending on the relationship between the number of members in the *ResultSet* input variable specified for the `DB_AttachProcedure` instruction and the number of data (columns) in the result set returned from the stored procedure called by the `DB_ExecuteProcedure` instruction, the `DB_ExecuteProcedure` instruction will generate the following results:

	Number of columns in the result set < Number of <i>ResultSet</i> members	Number of columns in the result set = Number of <i>ResultSet</i> members	Number of columns in the result set > Number of <i>ResultSet</i> members
Number of records in the result set < Number of elements in the <i>ResultSet</i> array	Instruction error (Invalid Number of Columns for Stored Procedure Result Set)	Normally completed. Values are written to the <i>ResultSet</i> array for the number of records in the result set. <i>Overflow</i> is False. <i>RecCnt</i> is the number of records in the result set.	Instruction error (Invalid Number of Columns for Stored Procedure Result Set)
Number of records in the result set = Number of elements in the <i>ResultSet</i> array			
Number of records in the result set > Number of elements in the <i>ResultSet</i> array		Normally completed. Values are written to the <i>ResultSet</i> array for the number of elements in the <i>ResultSet</i> array. <i>Overflow</i> is True. <i>RecCnt</i> is the number of elements in the <i>ResultSet</i> array.	

The instruction execution timeout is not available for the `DB_ExecuteProcedure` instruction. The *QueryTimeout* input variable is the timeout time for query execution. If a value other than 0 is set to the *QueryTimeout* input variable, the *QueryTimeout* input variable is enabled instead of the time specified in *Query Execution Timeout* in the DB Connection Settings. If the query execution timeout is reached, an instruction error (SQL Execution Error) occurs.

Precautions for Correct Use

- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual* (Cat. No.W502) for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- If the call processing of a stored procedure does not complete within the specified timeout for query execution, an instruction error (SQL Execution Error) occurs. Therefore, make sure that the execution time includes the extra time period considering the load fluctuation of the server where the stored procedures are executed.
- This instruction cannot be used on an event task. A compiling error will occur.
- When the DB Connection Service was started in Test Mode, this instruction ends normally.
- An error occurs for this instruction in the following cases. *Error* will be TRUE.
 - a) When the instruction was executed when the DB Connection Service was not running.
 - b) When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
 - c) When the instruction was executed while the DB Connection Service was stopped due to an error.
 - d) When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
 - e) When the value of the *DBConnection* input variable is invalid or the specified DB Connection is already closed.
 - f) When the value of the *ProcHandle* input variable is invalid.
 - g) The executed SQL statement resulted in an error in the DB.
 - h) The DB Connection Service cannot communicate with the DB due to a network failure or other causes.
 - i) When one or more SQL statements are already stored in the Spool memory.
 - j) The instruction was not completed within the time specified for query execution timeout.
 - k) The value of the *QueryTimeOut* input variable is outside the valid range.
 - l) When the instruction was executed before completion of the DB's processing for the DB Connection Instruction Execution Timeout that occurred for the previous *DB_Insert*, *DB_Update*, *DB_Select*, or *DB_Delete* instruction.
 - m) When more than 32 DB Connection Instructions were executed at the same time.

Sample Programming

This section gives sample programming for executing a stored procedure.

You will execute the stored procedure *Proc1*, which is used for performing the following processing:

- `Result_Proc1 = ArgIn.Add1 + ArgIn.Add2;`

Structure Data Type Definition

The structure settings for the sample programming are specified below.

Name	Data type
ADD_STOREDPROCEDURE	STRUCT
Add1	DINT

Name		Data type
	Add2	DINT

Ladder Diagram

● Main Variables

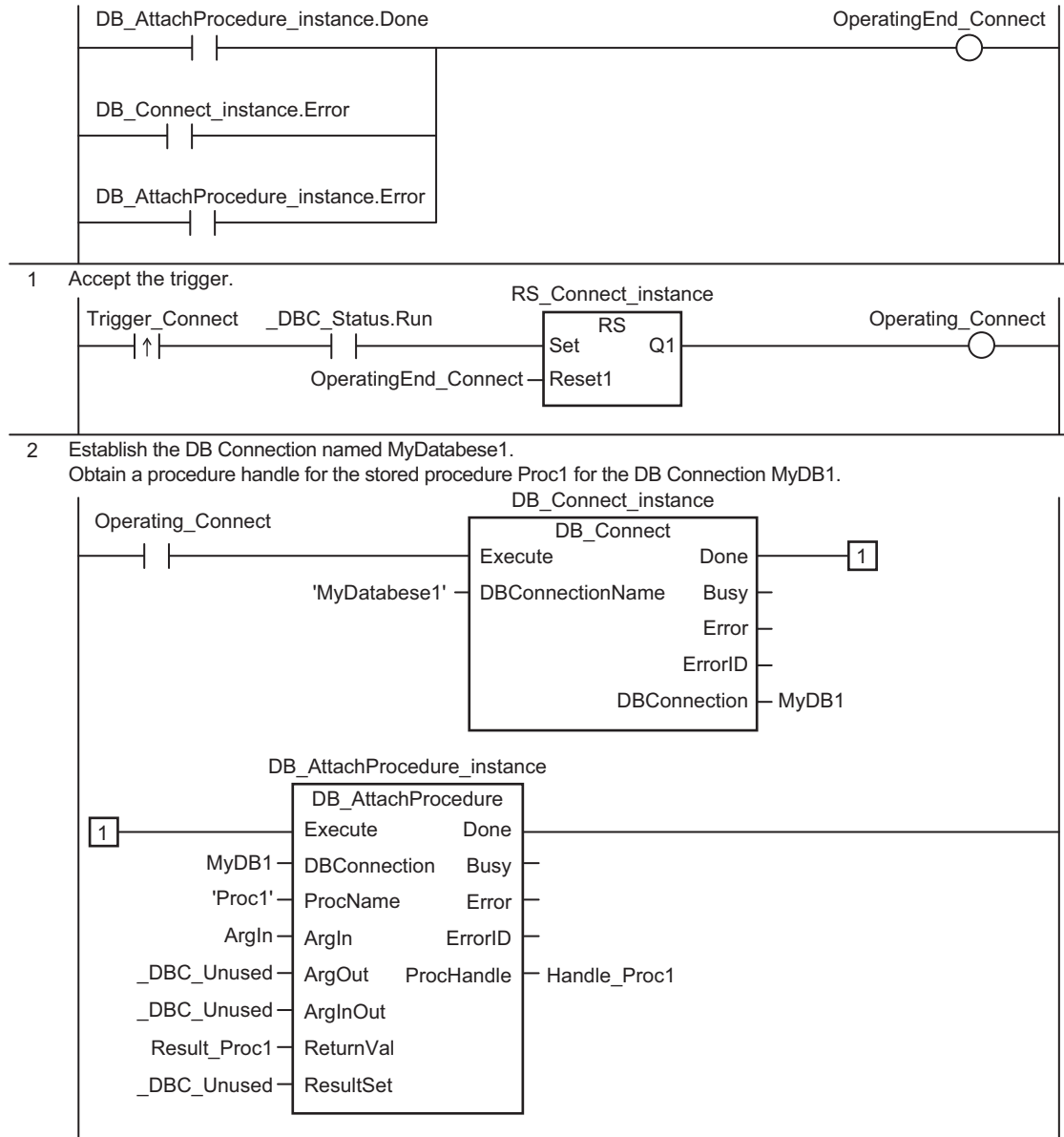
Name	Data type	Initial value	Comment
DB_Connect_instance	DB_Connect	---	Instance of the DB_Connect instruction
DB_Close_instance	DB_Close	---	Instance of the DB_Close instruction
MyDB1	DWORD	---	This variable is assigned to the DBConnection output variable from DB_Connect_instance.
Handle_Proc1	DWORD	---	Variable representing a procedure handle
Trigger_Connect	BOOL	---	Variable used as a trigger for establishing a DB Connection
Operating_Connect	BOOL	---	The DB_Connect instruction is executed when this variable is TRUE.
OperatingEnd_Connect	BOOL	---	This variable changes to TRUE when the DB_Connect instruction is completed.
RS_Connect_instance	RS	---	Instance of the RS instruction
Trigger_StoredProcedure	BOOL	---	Variable used as a trigger for executing a DB Stored Procedure
Operating_StoredProcedure	BOOL	---	The DB_ExecuteProcedure instruction is executed when this variable is TRUE.
OperatingEnd_StoredProcedure	BOOL	---	This variable changes to TRUE when the DB_ExecuteProcedure instruction is completed.
RS_StoredProcedure_instance	RS	---	Instance of the RS instruction
Trigger_Close	BOOL	---	Variable used as a trigger for closing the DB Connection
Operating_Close	BOOL	---	The DB_Close instruction is executed when this variable is TRUE.
OperatingEnd_Close	BOOL	---	The DB_Close instruction is executed when this variable is TRUE.
RS_Close_instance	RS	---	Instance of the RS instruction
DB_AttachProcedure_instance	DB_AttachProcedure	---	Instance of the DB_AttachProcedure instruction
DB_ExecuteProcedure_instance	DB_ExecuteProcedure	---	Instance of the DB_ExecuteProcedure instruction
DB_DetachProcedure_instance	DB_DetachProcedure	---	Instance of the DB_DetachProcedure instruction
Result_Proc1	DINT	---	Variable representing the return value of a stored procedure
ArgIn	ADD_STOREDPROCEDURE	---	Variable representing the IN argument of a stored procedure

● Sample Programming

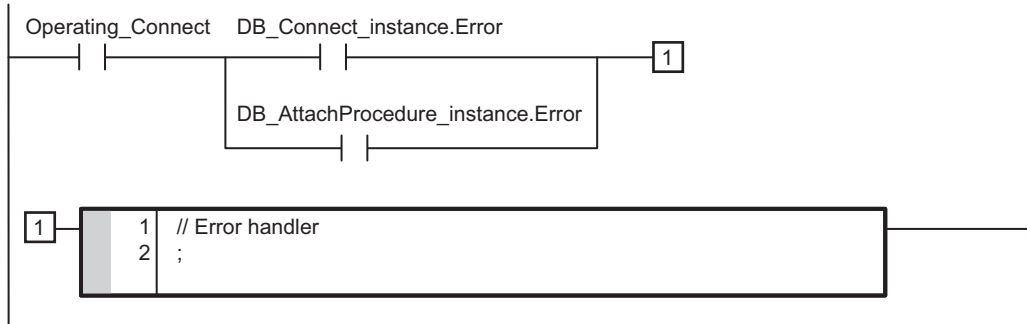
- 0 - The addition of two variables is processed on the database side by using a stored procedure.
The operation procedure is described below.
1. Use the DB_Connect instruction to establish connection with the database.
 2. Use the DB_AttachProcedure instruction to obtain a procedure handle used for calling a stored procedure to be executed.
 3. Use the DB_ExecuteProcedure instruction to execute the stored procedure.
 4. Use the DB_DetachProcedure instruction to release the procedure handle. Use the DB_Close instruction to disconnect the database connection.

- Establish a DB Connection named MyDatabase1 and obtain a procedure handle for the stored procedure.

Check the completion of the DB_Connect and DB_AttachProcedure instructions.



3 Execute the error handler.



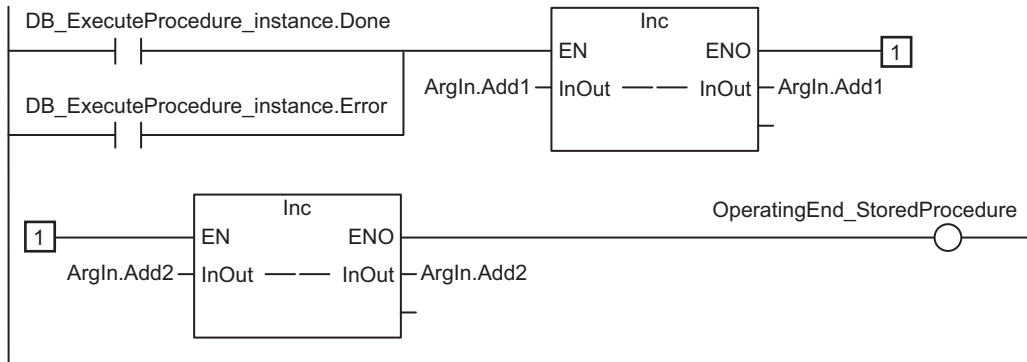
4

- Execute the following stored procedure called Proc1.

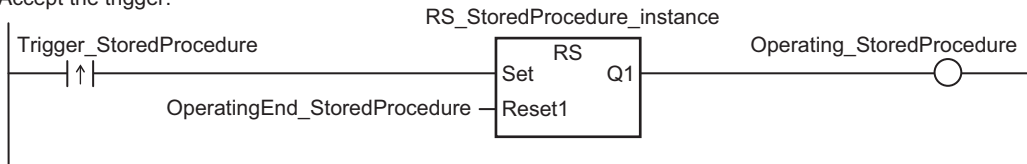
· Proc1

Result_Proc1 = ArgIn.Add1 + ArgIn.Add2;

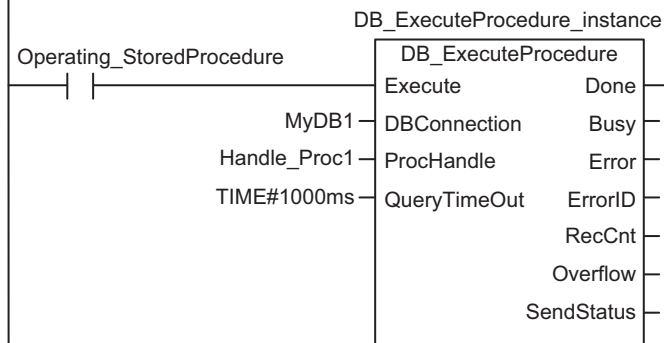
Check the completion of the DB_ExecuteStoredProcedure instruction.



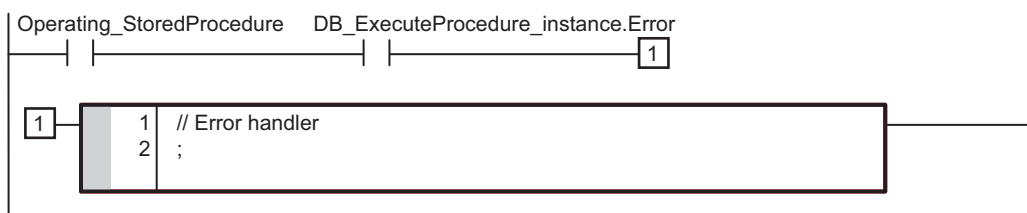
5 Accept the trigger.



6



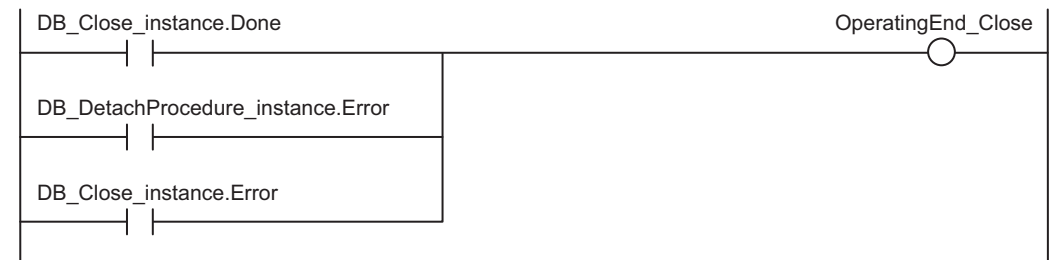
7 Execute the error handler.



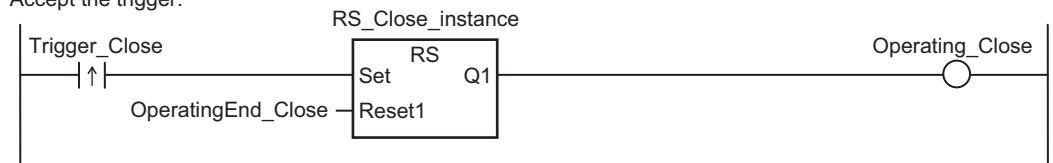
8

- Use the DB_DetachProcedure instruction to release the procedure handle.
- Close the DB Connection MyDB1.

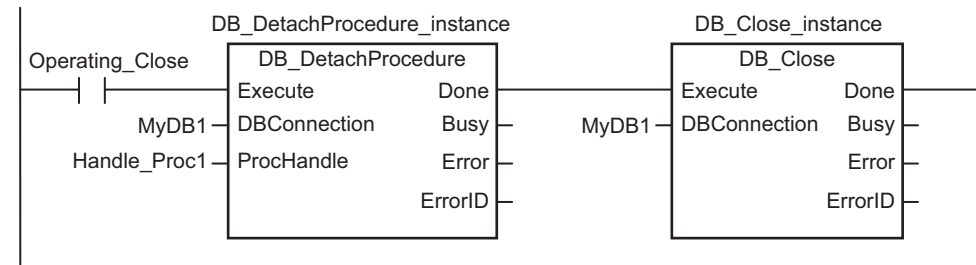
Check the completion of the DB_DetachProcedure and DB_Close instructions.



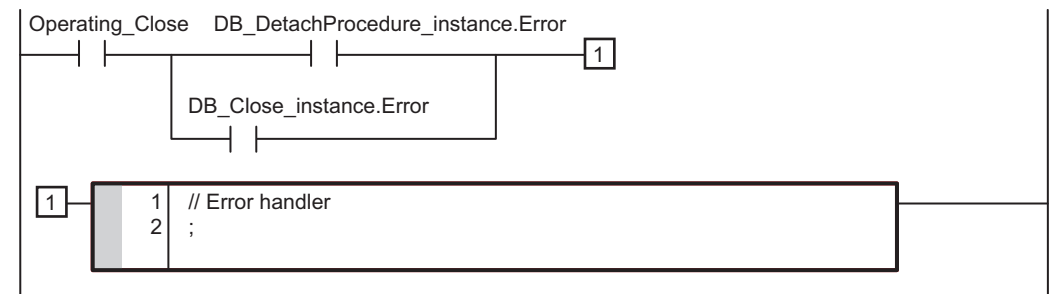
9 Accept the trigger.



10 Release the procedure handle for the stored procedure Proc1.
Close the DB Connection MyDB1.



11 Execute the error handler.



Structured Text (ST)

● Main Variables

Name	Data type	Initial value	Comment
DB_Connect_instance	DB_Connect	---	Instance of the DB_Connect instruction
DB_Close_instance	DB_Close	---	Instance of the DB_Close instruction
MyDB1	DWORD	---	This variable is assigned to the DBConnection output variable from DB_Connect_instance.
Trigger_Connect	BOOL	---	Variable used as a trigger for establishing a DB Connection
LastTrigger_Connect	BOOL	---	Variable to retain the trigger status of the previous execution

Name	Data type	Initial value	Comment
Operating_Connect	BOOL	---	The DB_Connect instruction is executed when this variable is TRUE.
OperatingStart_Connect	BOOL	---	The start processing for establishing a DB Connection is executed when this variable is TRUE.
Trigger_StoredProcedure	BOOL	---	Variable used as a trigger for executing a DB Stored Procedure
LastTrigger_StoredProcedure	BOOL	---	Variable to retain the trigger status of the previous execution
Operating_StoredProcedure	BOOL	---	The DB_ExecuteProcedure instruction is executed when this variable is TRUE.
Trigger_Close	BOOL	---	Variable used as a trigger for closing the DB Connection
LastTrigger_Close	BOOL	---	Variable to retain the trigger status of the previous execution
Operating_Close	BOOL	---	The DB_Close instruction is executed when this variable is TRUE.
OperatingStart_Close	BOOL	---	The start processing for closing a DB Connection is executed when this variable is TRUE.
Stage	INT	---	Variable that shows the status of the DB Connection
DB_AttachProcedure_instance	DB_AttachProcedure	---	Instance of the DB_AttachProcedure instruction
DB_ExecuteProcedure_instance	DB_ExecuteProcedure	---	Instance of the DB_ExecuteProcedure instruction
DB_DetachProcedure_instance	DB_DetachProcedure	---	Instance of the DB_DetachProcedure instruction
ArgIn	ADD_STOREDPROCEDURE	---	Variable representing the IN argument of a stored procedure
Handle_Proc1	DWORD	---	Variable representing a procedure handle
Result_Proc1	DINT	---	Variable representing the return value of a stored procedure

● Sample Programming

```
(* -----
---
- The addition of two variables is processed on the database side by using a stored
  procedure.
The operation procedure is described below.
1. Use the DB_Connect instruction to establish connection with the database.
2. Use the DB_AttachProcedure instruction to obtain a procedure handle used for calling a stored procedure to be executed.
3. Use the DB_ExecuteProcedure instruction to execute a stored procedure.
4. Use the DB_DetachProcedure instruction to release the procedure handle. Use the DB_Close instruction to disconnect the database connection.
*)

//-----
--
```

```

    //- Establish a DB Connection named MyDatabasesel and obtain a procedure handle for the
    stored procedure.

    // Start the sequence when Trigger_Connect changes to TRUE.
    IF ( (Trigger_Connect=TRUE) AND (LastTrigger_Connect=FALSE) AND ( _DBC_Status.Run=TRUE) ) THEN
        OperatingStart_Connect := TRUE;
        Operating_Connect := TRUE;
    END_IF;
    LastTrigger_Connect:=Trigger_Connect;

    // Sequence start processing
    IF (OperatingStart_Connect=TRUE) THEN
        // Initialize the instances of the applicable DB Connection Instructions.
        DB_Connect_instance(Execute:=FALSE );
        DB_AttachProcedure_instance( Execute:=FALSE, DBConnection:=MyDB1, ProcName:
        ='Proc1', ArgIn:=_DBC_Unused, ArgOut:=_DBC_Unused, ArgInOut:=_DBC_Unused, ReturnVal
        :=Result_Proc1, ResultSet:=_DBC_Unused);

        Stage := INT#1;
        OperatingStart_Connect := FALSE;
    END_IF;

    // Establish the DB Connection named MyDatabasesel
    // Obtain a procedure handle for the stored procedure Proc1 for the DB Connection MyDB1.
    IF (Operating_Connect=TRUE) THEN
        CASE Stage OF
            1 : // Establish the DB Connection
                DB_Connect_instance( Execute:=TRUE, DBConnectionName:= 'MyDatabasesel', DBConnection=>MyDB1 );

                IF (DB_Connect_instance.Done=TRUE) THEN
                    Stage := INT#2; // Normal end
                ELSIF (DB_Connect_instance.Error=TRUE) THEN
                    Stage := INT#99; // Error
                END_IF;

            2 : // Obtain a procedure handle
                DB_AttachProcedure_instance( Execute:=TRUE, DBConnection:=MyDB1, ProcName:='Proc1', ArgIn:=ArgIn, ArgOut:=_DBC_Unused, ArgInOut:=_DBC_Unused, ReturnVal:=Result_Proc1, ResultSet:=_DBC_Unused, ProcHandle=>Handle_Proc1);

                IF ( DB_AttachProcedure_instance.Done=TRUE) THEN
                    Operating_Connect:=FALSE; // Normal end
                ELSIF ( DB_AttachProcedure_instance.Error=TRUE ) THEN
                    Stage := INT#99; // Error

```

```

                                END_IF;

    99 :
                                // Error handler
                                Operating_Connect := FALSE;
    END_CASE;
END_IF;

//-----
--
// - Execute the following stored procedure called Proc1.
// • Proc1
// Result_Proc1 = ArgIn.Add1 + ArgIn.Add2;

// Start the sequence when Trigger_StoredProcedure changes to TRUE.
IF ( (Trigger_StoredProcedure=TRUE) AND (LastTrigger_StoredProcedure=FALSE) ) THEN
    Operating_StoredProcedure := TRUE;
    // Initialize the instance of the applicable DB Connection Instruction.
    DB_ExecuteProcedure_instance( Execute:=FALSE );
END_IF;
LastTrigger_StoredProcedure := Trigger_StoredProcedure;

IF( Operating_StoredProcedure=TRUE ) THEN
    DB_ExecuteProcedure_instance( Execute:=TRUE, DBConnection:=MyDB1, ProcHandle:=Handle_Proc1, QueryTimeOut:=TIME#1000ms );

    IF (DB_ExecuteProcedure_instance.Done=TRUE) THEN
        Operating_StoredProcedure := FALSE; // Normal end
        Inc(ArgIn.Add1);
        Inc(ArgIn.Add2);
    ELSIF (DB_ExecuteProcedure_instance.Error=TRUE) THEN
        // Error handler
        Operating_StoredProcedure := FALSE;
        Inc(ArgIn.Add1);
        Inc(ArgIn.Add2);
    END_IF;
END_IF;

(*-----
--
    Use the DB_DetachProcedure instruction to release the procedure handle.
- Close the DB Connection MyDB1.
*)

// Start the sequence when Trigger_Close changes to TRUE.
IF ( (Trigger_Close=TRUE) AND (LastTrigger_Close=FALSE) ) THEN
    OperatingStart_Close := TRUE;

```

```

        Operating_Close := TRUE;
    END_IF;
    LastTrigger_Close := Trigger_Close;

    // Sequence start processing
    IF (OperatingStart_Close=TRUE) THEN
        // Initialize the instance of the applicable DB Connection Instruction.
        DB_DetachProcedure_instance( Execute:=FALSE );
        DB_Close_instance( Execute:=FALSE );

        OperatingStart_Close := FALSE;
    END_IF;

    // Release the procedure handle for the stored procedure Procl.
    // Close the DB Connection MyDB1.
    IF (Operating_Close=TRUE) THEN
        // Release the procedure handle
        DB_DetachProcedure_instance(Execute:=TRUE, DBConnection:=MyDB1, ProcHandle:
=Handle_Procl );

        IF (DB_DetachProcedure_instance.Done=TRUE) THEN
            // Close the DB Connection.
            DB_Close_instance( Execute:=TRUE, DBConnection:=MyDB1 );
        ELSIF (DB_DetachProcedure_instance.Error=TRUE) THEN
            // Error handler
            // Close the DB Connection.
            DB_Close_instance( Execute:=TRUE, DBConnection:=MyDB1 );
        END_IF;

        IF (DB_Close_instance.Done=TRUE) THEN
            Operating_Close := FALSE; // Normal end
        ELSIF (DB_Close_instance.Error=TRUE) THEN
            // Error handler
            Operating_Close := FALSE;
        END_IF;
    END_IF;
END_IF;

```

DB_DetachProcedure (Release DB Stored Procedure Handle)

The DB_DetachProcedure instruction releases the stored procedure that is obtained by a DB_AttachProcedure instruction.

Instruction	Name	FB/FUN	Graphic expression	ST expression
DB_DetachProcedure	Release DB Stored Procedure Handle	FB		DB_DetachProcedure_instance (Execute, DBConnection, ProcHandle, Done, Busy, Error, ErrorID) ;

Note The DB_DetachProcedure_instance is an instance of DB_DetachProcedure instruction, which is declared as a variable.

Variables

Input Variable

Name	Meaning	Data type	Valid range	Unit	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	---	FALSE	Specify the execution condition.
DBConnection	DB Connection	DWORD	16#0 to FFFFFFFF	---	16#0	Specify the DB connection established by a DB_Connect instruction.
ProcHandle	Procedure Handle	DWORD	16#0 to FFFFFFFF	---	16#0	Procedure handle obtained by a DB_AttachProcedure instruction.

Output Variable

Name	Meaning	Data type	Valid range	Unit	Description
Done	Done	BOOL	TRUE or FALSE	---	TRUE when the instruction is normally completed.
Busy	Executing	BOOL	TRUE or FALSE	---	TRUE when the instruction is being executed.
Error	Error	BOOL	TRUE or FALSE	---	TRUE when the instruction is terminated due to an error.
ErrorID	Error Code	WORD	16#0 to FFFF	---	Contains the error code when an error occurs.

Related System-defined Variables

Refer to *System-defined Variables Related to DB Connection Service* on page 7-5.

Related Error Codes

Error code	Meaning	Description
041D hex	Too Many Instructions Executed at the Same Time	More than 32 DB Connection Instructions were executed at the same time.
3000 hex	DB Connection Service not Started	When the instruction was executed when the DB Connection Service was not running.
3002 hex	DB Connection Service Shutdown or Shutting Down	When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
3008 hex	Invalid DB Connection	When the value of the <i>DBConnection</i> input variable is invalid or the specified DB Connection is already closed.
3013 hex	DB Connection Service Error Stop	When the instruction was executed while the DB Connection Service was stopped due to an error.
3015 hex	DB Connection Service Initializing	When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
3018 hex	Invalid Stored Procedure Handle	The <i>ProcHandle</i> variable obtained by the DB_AttachProcedure instruction is not specified. The <i>ProcHandle</i> variable released by the DB_DetachProcedure is specified. The <i>ProcHandle</i> variable obtained in other than the relevant connection is specified.
3019 hex	Instruction Executed for Unsupported Database Type	The instruction was executed for a database type that is not supported by this instruction.

Function

This instruction releases the stored procedure that is obtained by a DB_AttachProcedure instruction.

Precautions for Correct Use

- Execution of this instruction is continued until processing is completed even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is completed. Use this to confirm normal completion of processing.
- Refer to "Using this Section" of the *NJ/NX-series Instructions Reference Manual* (Cat. No. W502) for a timing chart for *Execute*, *Done*, *Busy*, and *Error*.
- This instruction cannot be used on an event task. A compiling error will occur.
- When the DB Connection Service was started in Test Mode, this instruction ends normally.
- An error occurs for this instruction in the following cases. *Error* will be TRUE.
 - a) When the instruction was executed when the DB Connection Service was not running.
 - b) When the instruction was executed while the initialization processing of the DB Connection Service was in progress.
 - c) When the instruction was executed while the DB Connection Service was stopped due to an error.
 - d) When the instruction was executed after the DB Connection Service was shut down or while the DB Connection Service was being shut down.
 - e) When the value of the *DBConnection* input variable is invalid or the specified DB Connection is already closed.
 - f) When the value of the *ProcHandle* input variable is invalid.
 - g) When more than 32 DB Connection Instructions were executed at the same time.

Sample Programming

Refer to *Sample Programming* on page 7-115 for the sample programming that is provided for the DB_ExecuteProcedure instruction.

8

Troubleshooting

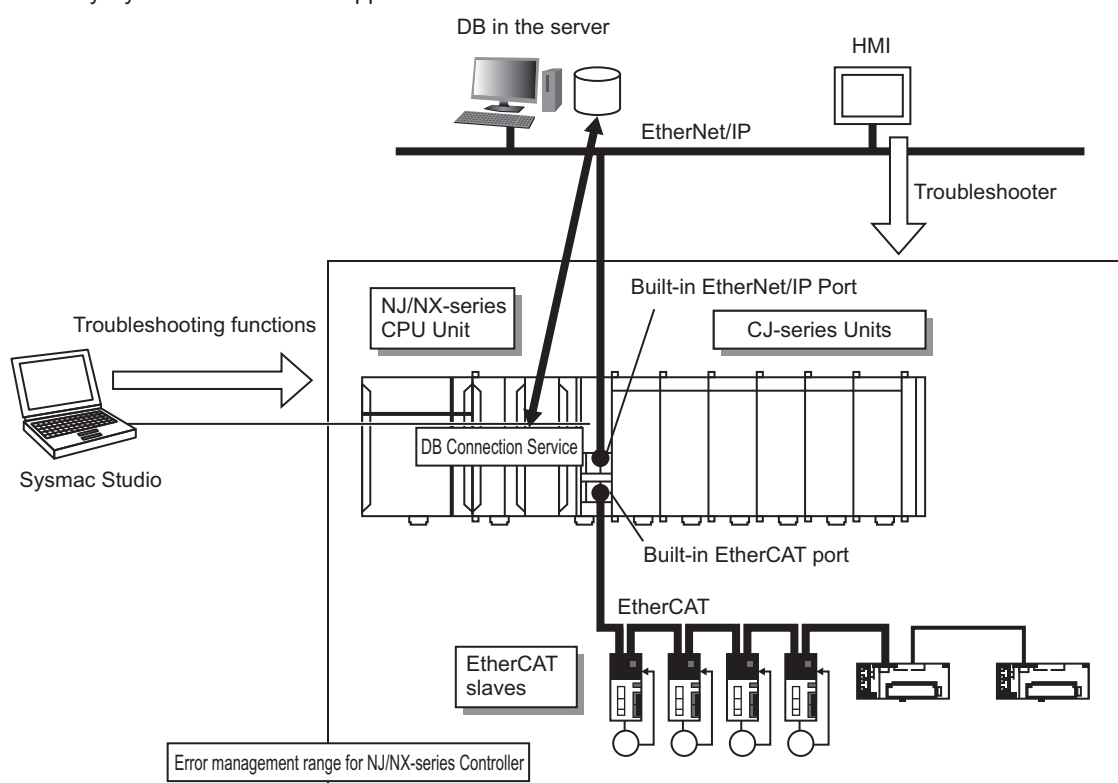
This section describes the error confirmation methods and corrections for errors that can occur in the DB Connection Service.

8-1	Overview of Errors	8-2
8-1-1	How to Check for Errors	8-2
8-1-2	Errors Related to the DB Connection Service	8-5
8-2	Troubleshooting	8-8
8-2-1	Error Table	8-8
8-2-2	Error Descriptions	8-16

8-1 Overview of Errors

You manage all of the errors that occur on the NJ/NX-series Controller as events. The same methods are used for all events. This allows you to see what errors have occurred and find corrections for them with the same methods for the entire range of errors that is managed (i.e., CPU Unit, EtherCAT slaves,^{*1} and CJ-series Units).

^{*1}. Only Sysmac devices are supported.



You can use the troubleshooting functions of Sysmac Studio or the Troubleshooter on an HMI to quickly check for errors that have occurred and find corrections for them.

This manual describes the errors that originate in the DB Connection Service.

Refer to the *NJ/NX-series Troubleshooting Manual* (Cat. No. W503) for specific corrections when errors occur and for troubleshooting information on the entire NJ/NX-series Controller.

For information on errors that occur when DB Connection Instructions are executed, refer to *Section 7 DB Connection Instructions* on page 7-1.

8-1-1 How to Check for Errors

You can check to see if an error has occurred with the following methods.

Checking method	What you can check
Checking the indicators	CPU Unit operating status
Checking with the troubleshooting function of Sysmac Studio	You can check for current Controller errors, a log of past Controller errors, error sources, error causes, corrections, and error log of CJ-series Special Units. ^{*1}
Checking with the Troubleshooter of an HMI ^{*2}	You can check for current Controller errors, a log of past Controller errors, error sources, causes, and corrections.

Checking method	What you can check
Checking with instructions that read function module error status	You can check the highest-level status and highest-level event code in the current Controller errors.
Checking with system-defined variables	You can check the current Controller error status for each function module.

*1. Detailed information such as error causes and corrections are not displayed.

*2. To perform troubleshooting from an HMI, connect the HMI to the built-in EtherNet/IP port on the CPU Unit.

This section describes the above checking methods.

Checking the Indicators

You can use the PWR indicator on the Power Supply Unit and the RUN and ERROR indicators on the CPU Unit to determine the event level for an error. The following table shows the relationship between the Controller's indicators and the event level.

Indicator			CPU Unit operating status	Error confirmation with Sysmac Studio or an HMI
PWR	RUN	ERROR		
Not lit	Not lit	Not lit	Power Supply Error	Not possible: Refer to the <i>NJ/NX-series Troubleshooting Manual</i> (Cat. No. W503).
Lit	Not lit	Not lit	CPU Unit Reset* ¹	
Lit	Flashing	Lit	Incorrect Power Supply Unit Connected	
Lit	Not lit	Lit	CPU Unit Watchdog Timer Error* ²	
Lit	Not lit	Lit	Major fault level* ²	Possible: Connect Sysmac Studio or an HMI and check the cause of and correction for the error in the troubleshooting functions of Sysmac Studio or the Troubleshooter of an HMI.
Lit	Lit	Flashing	Partial fault level	
Lit	Lit	Flashing	Minor fault level	
Lit	Lit	Not lit	Observation	
Lit	Lit	Not lit	Normal operation in RUN mode	---
Lit	Not lit	Not lit	Normal operation in PROGRAM mode* ¹	---
Lit	Flashing	Not lit	Normal operation in startup state	---

*1. If you can go online with the CPU Unit from Sysmac Studio with a "direct USB connection", the CPU Unit is in "PROGRAM mode". If you cannot go online, the "CPU Unit is being reset".³

*2. If you can go online with the CPU Unit from Sysmac Studio with a "direct USB connection", a "major fault level" error has occurred. If you cannot go online, a "watchdog timer error has occurred in the CPU Unit".³

*3. If you cannot go online with the CPU Unit from Sysmac Studio, it is also possible that the USB cable is faulty or that the "network type" on Sysmac Studio is not set for a "direct USB connection". Refer to the *NJ/NX-series Troubleshooting Manual* (Cat. No. W503) if you cannot go online with the CPU Unit.



Precautions for Correct Use

Since the NX102-□□20 CPU Units do not have a USB port, it is not possible to connect the Sysmac Studio via "direct USB connection".

Checking with the Troubleshooting Function of Sysmac Studio

When an error occurs, you can connect Sysmac Studio online to the Controller to check current Controller errors and the log of past Controller errors.

You can also check the cause of the error and corrections.

Refer to the *NJ/NX-series Troubleshooting Manual (Cat. No. W503)* for the procedures to check for errors with Sysmac Studio.

Checking with the Troubleshooter of an HMI

If you can connect communications between an HMI and the Controller when an error occurs, you can check for current Controller errors and the log of past Controller errors.

You can also check the cause of the error and corrections.

Refer to the *NJ/NX-series Troubleshooting Manual (Cat. No. W503)* for the procedures to check for errors with an HMI.

Checking with Instructions That Read Error Status

You can use instructions in the user program to check the error status of each function module.

The following table gives the instruction that is used to get error information for the DB Connection Service.

Instruction	Name	Function
GetPLCError	Get PLC Error Status	The GetPLCError instruction gets the highest level status (partial fault or minor fault) and highest level event code of the current Controller errors in the PLC Function Module.

For details on the instructions that get error status, refer to the *NJ/NX-series Instructions Reference Manual (Cat. No. W502)*.

Checking with System-defined Variables

You can use the "error status variables" and "status variables" in the system-defined variables to check for errors that have occurred in the DB Connection Service.

● Error Status Variables

You can check for errors in each function module of the NJ/NX-series Controller with error status variables.

The following variables show the error status of the PLC Function Module.

Variable name	Data type	Meaning	Function
_PLC_ErrSta	WORD	PLC Function Module Error Status	Gets the collective error status of all error status for the PLC Function Module.

● Status Variables

Variable name	Data type	Meaning	Function
_DBC_Status	_sDBC_STA-TUS	DB Connection Service Status	Shows the status of the DB Connection Service.
Run	BOOL	Running Flag	TRUE while the DB Connection Service is running. FALSE while the DB Connection Service is not running.
Test	BOOL	Test Mode	TRUE while the DB Connection Service is running in Test Mode. FALSE while the DB Connection Service is not running in Test Mode.
Idle	BOOL	Idle	TRUE while the DB Connection Service is idle. FALSE while the DB Connection Service is not idle.
Error	BOOL	Error Flag	TRUE when the DB Connection Service has an error. FALSE when the DB Connection Service has no error.
Shutdown	BOOL	Shutdown	TRUE when the DB Connection Service has been shut down. FALSE when the DB Connection Service has not been shut down.

8-1-2 Errors Related to the DB Connection Service

Classifications

There are the following two sources of errors in the DB Connection Service.

Classification	Event source	Source details	Log category		
			System log	Access log	User-defined event log
DB Connection Service	PLC Function Module	DB Connection Service	Yes	No	No
DB Connection Instruction	PLC Function Module	Instruction	Yes	No	No

Event Levels

This section describes the operation of the DB Connection Service for each event level.

Event level of the error	Operation
Major fault	All NJ/NX-series Controller control operations stop for errors in this event level.
Partial fault	All control operations for one of the function modules in the NJ/NX-series Controller stop for errors in this event level. If a partial fault level error occurs in the DB Connection Service, all functions of the DB Connection Service stop.
Minor fault	Some of the control operations for one of the function modules in the NJ/NX-series Controller stop for errors in this event level.
Observation	Errors in the observation level do not affect NJ/NX-series Controller control operations. Observations are reported in order to prevent them from developing into errors at the minor fault level or higher.

Event level of the error	Operation
Information	Events that are classified as information provide information that do not indicate errors.

DB Connection Service Errors by Source

The following tables list the errors in each event level that can occur for each source.

DB Connection Service Errors

Level	Error name
Major fault	None
Partial fault	None
Minor fault	• Spool Memory Corrupted • Execution Log Save Failed • SQL Execution Failure Log Save Failed • DB Connection Setting Error • DB Connection Disconnected Error
Observation	None
Information	• DB Connection Service Started • DB Connection Service Stopped • DB Connection Service Shutdown • Spool Memory Cleared

DB Connection Instruction Errors

Level	Error name
Major fault	None
Partial fault	None
Minor fault	None
Observation	• DB Connection Service not Started • DB Connection Service Run Mode Change Failed • DB Connection Service Shutdown or Shutting Down • Invalid DB Connection Name • DB Connection Rejected • DB Connection Failed • DB Connection Already Established • Too Many DB Connections • Invalid DB Connection • Invalid DB Map Variable • Unregistered DB Map Variable • SQL Execution Error • Spool Capacity Exceeded • Invalid Extraction Condition • Log Code Out of Range • DB Connection Disconnected Error Status • DB Connection Instruction Execution Timeout • DB Connection Service Error Stop • Data Already Spooled • DB Connection Service Initializing • DB in Process • Operation Log Disabled
Information	None

8-2 Troubleshooting

This section describes the errors that can occur in the DB Connection Service and the corrections for them.

8-2-1 Error Table

The errors (i.e., events) that can occur in the DB Connection Service and DB Connection Instructions are given on the following pages. The following abbreviations and symbols are used in the event level column.

Abbreviation	Name
Maj	Major fault level
Prt	Partial fault level
Min	Minor fault level
Obs	Observation
Info	Information

Symbol	Meaning
S	Event levels that are defined by the system.
U	Event levels that can be changed by the user.*1

*1. This symbol appears only for events for which the user can change the event level.

Refer to the *NJ/NX-series Troubleshooting Manual* (Cat. No. W503) for all NJ/NX-series event codes.

Errors Related to DB Connection Service

Event code	Event name	Meaning	Assumed cause	Level					Reference
				Maj	Prt	Min	Obs	Info	
14D00000 hex	Spool Memory Corrupted	The Spool memory is corrupted.	The user application made an invalid writing to the Spool memory.			S			page 8-17
14D20000 hex	Execution Log Save Failed	Failed to save the Execution Log to the SD Memory Card.	- An SD Memory Card is not inserted. - The SD Memory Card is not the correct type of card. - The format of the SD Memory Card is not correct. - The SD Memory Card is write-protected. - The capacity of the SD Memory Card is insufficient. - The SD Memory Card is damaged.			S	U		page 8-18

Event code	Event name	Meaning	Assumed cause	Level					Reference
				Maj	Prt	Min	Obs	Info	
14D30000 hex	SQL Execution Failure Log Save Failed	Failed to save the SQL Execution Failure Log to the SD Memory Card.	- An SD Memory Card is not inserted. - The SD Memory Card is not the correct type of card. - The format of the SD Memory Card is not correct. - The SD Memory Card is write-protected. - The capacity of the SD Memory Card is insufficient. - The SD Memory Card is damaged.			S	U		page 8-19
35300000 hex	DB Connection Setting Error	The DB Connection settings are not correct.	- The power supply to the Controller was interrupted during a download of the DB Connection settings. - The DB Connection settings are not correct because the power supply to the Controller was interrupted during a Clear All Memory operation. - The DB Connection settings are not correct because the power supply to the Controller was interrupted during a Restore operation. - Non-volatile memory failed.			S			page 8-20
35310000 hex	DB Server Certificate Error	The format of the DB server certificate is invalid.	- A DB server certificate in the invalid format (X.509) was downloaded. - The power supply to the CPU Unit was interrupted during transfer of DB connection settings. - The DB Connection settings are not correct because the power supply to the Controller was interrupted during a Clear All Memory operation. - The DB Connection settings are not correct because the power supply to the Controller was interrupted during a Restore operation. - Non-volatile memory failed.			S			page 8-21
85100000 hex	DB Connection Disconnected Error	The DB Connection was disconnected due to an error.	- The power supply to the server is OFF. - The DB is stopped in the server. - The Ethernet cable connector is disconnected. - The Ethernet cable is broken. - Noise			S			page 8-22

Event code	Event name	Meaning	Assumed cause	Level					Reference
				Maj	Prt	Min	Obs	In-fo	
95300000 hex	DB Connection Service Started	The DB Connection Service was started.	The DB Connection Service was successfully started.					S	page 8-23
95310000 hex	DB Connection Service Stopped	The DB Connection Service was stopped.	The DB Connection Service was stopped.					S	page 8-23
95320000 hex	DB Connection Service Shutdown	The DB Connection Service was shut down.	The DB Connection service was shut down.					S	page 8-23
95330000 hex	Spool Memory Cleared	The SQL statements was cleared from the spool memory.	The SQL statements was cleared from the spool memory.					S	page 8-24

Errors Related to DB Connection Instructions

Errors are given as event codes that use the error code as the lower four digits. For descriptions of an error code, refer to the description of the corresponding event code. For example, if the error code for the instruction is 16#3000, refer to the description for event code 54013000 hex.

Event code	Event name	Meaning	Assumed cause	Level					Reference
				Maj	Prt	Min	Obs	In-fo	
54013000 hex	DB Connection Service Not Started	The DB Connection Service has not been started.	- A command to start the DB Connection Service was not given before the execution of relevant instruction. - A command to stop the DB Connection Service was given before the execution of relevant instruction.				S		page 8-25
54013001 hex	DB Connection Service Run Mode Change Failed	Failed to change the Run mode of the DB Connection Service.	- Run mode change to "Test Mode" was executed by the relevant instruction while running in "Operation Mode". - Run mode change to "Operation Mode" was executed by the relevant instruction while running in "Test Mode". - Start of the DB Connection Service was commanded while the DB Connection Service was being stopped. - Shutdown of the DB Connection Service was commanded while the DB Connection Service was being stopped.				S		page 8-26

Event code	Event name	Meaning	Assumed cause	Level					Reference
				Maj	Prt	Min	Obs	In-fo	
54013002 hex	DB Connection Service Shutdown or Shutting Down	The DB Connection Service is already shut down or being shut down.	- The relevant instruction was executed after the DB Connection Service was shut down. - The relevant instruction was executed while the shutdown processing of the DB Connection Service was in progress.				S		page 8-27
54013003 hex	Invalid DB Connection Name	The specified DB Connection Name is not set in any DB Connection settings.	- The DB Connection Name specified in the <i>DBConnectionName</i> input variable of the relevant instruction is wrong. - The DB Connection Name set in the DB Connection settings is wrong.				S		page 8-27
54013004 hex	DB Connection Rejected	The DB rejected the connection.	The user name or password set in the DB Connection settings is wrong.				S		page 8-29
54013005 hex	DB Connection Failed	Failed to connect to the DB.	- A server does not exist for the specified IP address or the specified host name. - The power supply to the server is OFF. - The DB is stopped in the server. - The Ethernet cable connector is disconnected. - The Ethernet cable is broken.				S		page 8-29
54013006 hex	DB Connection Already Established	A same-name DB Connection is already established.	The relevant instruction was executed when a same-name DB Connection was already established.				S		page 8-31
54013007 hex	Too Many DB Connections	The number of DB Connections that can be established at the same time is exceeded.	The relevant instruction was executed when the maximum number of DB Connections that can be established at the same time were already established.				S		page 8-31
54013008 hex	Invalid DB Connection	The specified DB Connection is not correct, or the DB Connection is already closed.	- The DB Connection specified in the <i>DBConnection</i> input variable of the relevant instruction is wrong. - The DB Connection specified in the <i>DBConnection</i> input variable of the relevant instruction is closed.				S		page 8-32
54013009 hex	Invalid DB Map Variable	The specified DB Map Variable is not correct.	- A structure variable that contains a derivative data type of member was specified as a DB Map Variable. - A non-structure variable was specified as a DB Map Variable. - A structure array variable was specified as a DB Map Variable for INSERT or UPDATE.				S		page 8-32

Event code	Event name	Meaning	Assumed cause	Level					Reference
				Maj	Prt	Min	Obs	Info	
5401300A hex	Unregistered DB Map Variable	The specified DB Map Variable has not been registered.	- The DB Map Variable has not been created by a DB_CreateMapping instruction. - A variable that is not registered as a DB Map Variable was specified in <i>MapVar</i>. - The DB Connection specified in the relevant instruction is different from the one specified at the execution of DB_CreateMapping instruction.				S		page 8-34
5401300B hex	SQL Execution Error	The executed SQL statement resulted in an error.	- There is no column with the same name as a structure member of the DB Map Variable. - The table specified in the DB_CreateMapping instruction does not exist in the DB. - One or more structure member values of the DB Map Variable cannot be converted to the corresponding column's data type. - One or more column values cannot be converted to the corresponding structure member's data type of the DB Map Variable. - One or more structure member values of the DB Map Variable exceed the valid range of the corresponding column's data type. - The column specified in the extraction condition does not exist in the DB's records. (DB_Select instruction, DB_Update instruction, DB_Delete instruction) - The extraction condition has a syntax error. (DB_Select instruction, DB_Update instruction, DB_Delete instruction) - The column specified in the sort condition does not exist in the DB's records. (DB_Select instruction) - The sort condition has a syntax error. (DB_Select instruction) - The user does not have the access rights to the table.				S		page 8-35

Event code	Event name	Meaning	Assumed cause	Level					Reference
				Maj	Prt	Min	Obs	In-fo	
5401300C hex	Spool Capacity Exceeded	The SQL statement could not be stored in the Spool memory because its maximum capacity was exceeded.	- The DB connection failure has been continuing due to network failure or other factors. - The resend processing of the SQL statements stored in the Spool memory has not been executed (when the Resend spool data parameter is set to "Manual").				S		page 8-37
5401300E hex	Invalid Extraction Condition	The entered extraction condition is invalid.	A text string that consists of a NULL (16#00) character only was specified in the <i>Where</i> input variable.				S		page 8-38
54013010 hex	Log Code Out of Range	The value of the entered log code is outside the valid range.	A value outside the valid range from 0 to 9999 was specified.				S		page 8-39
54013011 hex	DB Connection Disconnected Error Status	The instruction could not be executed because the DB Connection had been disconnected due to an error.	- The power supply to the server is OFF. - The DB is stopped in the server. - The Ethernet cable connector is disconnected. - The Ethernet cable is broken. - Noise				S		page 8-39
54013012 hex	DB Connection Instruction Execution Timeout	The instruction was not completed within the time specified for instruction execution timeout.	- The power supply to the server is OFF. - The Ethernet cable connector is disconnected. - The Ethernet cable is broken. - The server's processing time is long.				S		page 8-41
54013013 hex	DB Connection Service Error Stop	The instruction could not be executed because the DB Connection Service was stopped due to an error.	The DB Connection settings are corrupted.				S		page 8-41

Event code	Event name	Meaning	Assumed cause	Level					Reference
				Maj	Prt	Min	Obs	Info	
54013014 hex	Data Already Spooled	One or more SQL statements are already stored in the Spool memory.	- A DB_Insert or DB_Update instruction was executed when one or more SQL statements were already stored in the Spool memory. - A DB_Select or DB_Delete instruction was executed when one or more SQL statements were already stored in the Spool memory.				S		page 8-43
54013015 hex	DB Connection Service Initializing	The instruction could not be executed because the initialization processing of the DB Connection Service is in progress.	The relevant instruction was executed during the initialization processing of the DB Connection Service.				S		page 8-43
54013016 hex	DB in Process	The instruction could not be executed because the DB is under processing in the server.	Though a DB Connection Instruction Execution Timeout occurred for the previous instruction, the relevant instruction was executed before completion of the DB's processing in the server.				S		page 8-45
54013017 hex	Operation Log Disabled	The log could not be recorded because the specified Operation Log is disabled.	- Though Execution Log was specified in the <i>LogType</i> input variable, the Execution Log is disabled. - Though Debug Log was specified in the <i>LogType</i> input variable, recording to the Debug Log is stopped.				S		page 8-45
54013018 hex	Invalid Procedure Handle	The specified procedure handle is invalid.	The procedure handle specified in the <i>ProcHandle</i> input variable of the relevant instruction is wrong.				S		page 8-46
54013019 hex	Instruction Executed for Unsupported Database Type	The instruction was executed for a database type that is not supported by this instruction.	The database type specified in the DB Connection Settings is not supported by the relevant instruction.				S		page 8-47

Event code	Event name	Meaning	Assumed cause	Level					Reference
				Maj	Prt	Min	Obs	Info	
5401301A hex	Invalid Stored Procedure Name	The specified stored procedure name does not exist.	The stored procedure name specified in the <i>ProcName</i> input variable of the relevant instruction does not exist in the server-side database. Note This includes when the specified stored procedure name does not find on the DB. Even if the stored procedure name exists, the DB Connection Service cannot find the stored procedure name due to the reason that the user does not have the access right to the stored procedure, or other reasons.				S		page 8-48
5401301B hex	Invalid Stored Procedure Argument	The attached argument information does not match the argument of the stored procedure.	The name, number, and type of the stored procedure argument data that is retrieved from the server-side database do not match those of the input variables <i>ArgIn</i>, <i>ArgOut</i>, and <i>ArgInOut</i> of the relevant instruction.				S		page 8-49
5401301C hex	Invalid Number of Columns for Stored Procedure Result Set	The number of columns in the stored procedure result set do not match the number of structure variable members where the result is stored.	The number of columns in the result set retrieved by the relevant instruction do not match the number of structure variable members where the result is stored.				S		page 8-50
5401301D hex	Invalid Stored Procedure Execution	An error occurred when a stored procedure is executed. Troubleshoot the error according to the attached information 4.	Check the cause according to the attached information 4 and the manual.				S		page 8-51

8-2-2 Error Descriptions

Controller Error Descriptions

The items that are used to describe individual errors (events) are described in the following copy of an error table.

Event name	Gives the name of the error.		Event code	Gives the code of the error.	
Meaning	Gives a short description of the error.				
Source	Gives the source of the error.		Source details	Gives details on the source of the error.	Detection timing Tells when the error is detected.
Error attributes	Level	Tells the level of influence on control.*1	Recovery	Gives the recovery method.*2	Log category Tells which log the error is saved in.*3
Effects	User program	Tells what will happen to execution of the user program.*4	Operation	Provides special information on the operation that results from the error.	
System-defined variables	Variable	Data type		Name	
	Lists the variable names, data types, and meanings for system-defined variables that provide direct error notification, that are directly affected by the error, or that contain settings that cause the error.				
Cause and correction	Assumed cause		Correction		Prevention
	Lists the possible causes, corrections, and preventive measures for the error.				
Attached information	This is the attached information that is displayed by Sysmac Studio or HMI.				
Precautions/Remarks	Provides precautions, restrictions, and supplemental information. If the user can set the event level, the event levels that can be set, the recovery method, operational information, and other information is also provided.				

*1. One of the following:

Major fault: Major fault level
 Partial fault: Partial fault level
 Minor fault: Minor fault level
 Observation
 Information

*2. One of the following:

Automatic recovery: Normal status is restored automatically when the cause of the error is removed.
 Error reset: Normal status is restored when the error is reset after the cause of the error is removed.
 Cycle the power supply: Normal status is restored when the power supply to the Controller is turned OFF and then back ON after the cause of the error is removed.
 Controller reset: Normal status is restored when the Controller is reset after the cause of the error is removed.
 Depends on cause: The recovery method depends on the cause of the error.

*3. One of the following:

System: System event log
 Access: Access event log

*4. One of the following:

Continues: Execution of the user program will continue.
 Stops: Execution of the user program stops.
 Starts: Execution of the user program starts.

Errors Related to DB Connection Service

Event name	Spool Memory Corrupted			Event code	14D00000 hex	
Meaning	The Spool memory is corrupted.					
Source	PLC Function Module		Source details	DB Connection Service	Detection timing	When the DB Connection Service is started
Error attributes	Level	Minor fault	Recovery	Error reset	Log category	System
Effects	User program	Continues.	Operation	Not affected.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	The user application made an invalid writing to the Spool memory.		Check for writing from the user application to the Spool memory area. Correct the user application, and then execute the Clear Spool Data operation.		Do not write to the Spool memory area from the user application.	
Attached information	None					
Precautions/Remarks	None					

Event name	Execution Log Save Failed			Event code	14D20000 hex	
Meaning	Failed to save the Execution Log to the SD Memory Card.					
	PLC Function Module		Source details	DB Connection Service	Detection timing	Continuously
Error attributes	Level	Minor fault	Recovery	Error reset	Log category	System
Effects	User program	Continues.	Operation	Not affected.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	An SD Memory Card is not inserted.		Insert an SD Memory Card.		Insert an SD Memory Card.	
	The SD Memory Card is not the correct type of card.		Replace the SD Memory Card with an SD or SDHC card.		Use an SD or SDHC card.	
	The format of the SD Memory Card is not correct.		Format the SD Memory Card with Sysmac Studio.		Use a formatted SD Memory Card. Also, do not remove the SD Memory Card or turn OFF the power supply while the SD BUSY indicator is lit.	
	The SD Memory Card is write-protected.		Remove write protection from the SD Memory Card.		Make sure that the SD Memory Card is not write-protected.	
	The capacity of the SD Memory Card is insufficient.		Replace the SD Memory Card for one with sufficient available space.		Use an SD Memory Card that has sufficient available space.	
	The SD Memory Card is damaged.		If none of the above causes applies, replace the SD Memory Card.		Do not remove the SD Memory Card or turn OFF the power supply while the SD BUSY indicator is lit. Do not remove the SD Memory Card while the SD PWR indicator is lit. Replace the SD Memory Card periodically according to the write life of the SD Memory Card.	
Attached information	Attached information 1: Error Details 0001 hex: An SD Memory Card is not inserted. 0002 hex: The SD Memory Card is damaged, the format of the SD Memory Card is not correct, or the SD Memory Card is not the correct type of card. 0003 hex: The SD Memory Card is write-protected. 0302 hex: The SD Memory Card is damaged, the capacity of the SD Memory Card is insufficient, or failed to save a file to the SD Memory Card due to other factors.					
Precautions/Remarks	You can change the error level to the observation.					

Event name	SQL Execution Failure Log Save Failed			Event code	14D30000 hex	
Meaning	Failed to save the SQL Execution Failure Log to the SD Memory Card.					
Source	PLC Function Module		Source details	DB Connection Service	Detection timing	Continuously
Error attributes	Level	Minor fault	Recovery	Error reset	Log category	System
Effects	User program	Continues.	Operation	Not affected.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	An SD Memory Card is not inserted.		Insert an SD Memory Card.		Insert an SD Memory Card.	
	The SD Memory Card is not the correct type of card.		Replace the SD Memory Card with an SD or SDHC card.		Use an SD or SDHC card.	
	The format of the SD Memory Card is not correct.		Format the SD Memory Card with Sysmac Studio.		Use a formatted SD Memory Card. Also, do not remove the SD Memory Card or turn OFF the power supply while the SD BUSY indicator is lit.	
	The SD Memory Card is write-protected.		Remove write protection from the SD Memory Card.		Make sure that the SD Memory Card is not write-protected.	
	The capacity of the SD Memory Card is insufficient.		Replace the SD Memory Card for one with sufficient available space.		Use an SD Memory Card that has sufficient available space.	
	The SD Memory Card is damaged.		If none of the above causes applies, replace the SD Memory Card.		Do not remove the SD Memory Card or turn OFF the power supply while the SD BUSY indicator is lit. Do not remove the SD Memory Card while the SD PWR indicator is lit. Replace the SD Memory Card periodically according to the write life of the SD Memory Card.	
Attached information	Attached information 1: Error Details 0001 hex: An SD Memory Card is not inserted. 0002 hex: The SD Memory Card is damaged, the format of the SD Memory Card is not correct, or the SD Memory Card is not the correct type of card. 0003 hex: The SD Memory Card is write-protected. 0302 hex: The SD Memory Card is damaged, the capacity of the SD Memory Card is insufficient, or failed to save a file to the SD Memory Card due to other factors.					
Precautions/Remarks	You can change the error level to the observation.					

Event name	DB Connection Setting Error			Event code	35300000 hex	
Meaning	The DB Connection settings are not correct.					
Source	PLC Function Module		Source details	DB Connection Service	Detection timing	At download, power ON, or Controller reset
Error attributes	Level	Minor fault	Recovery	Automatic recovery	Log category	System
Effects	User program	Continues.	Operation	The DB Connection Service cannot be started. The operation status of the DB Connection Service is changed to "Error Stop".		
System-defined variables	Variable		Data type		Name	
	_DBC_Status		_sDBC_STATUS		DB Connection Service Status	
Cause and correction	Assumed cause		Correction		Prevention	
	The power supply to the Controller was interrupted during a download of the DB Connection settings.		Transfer the DB Connection settings again from Sysmac Studio.		Do not turn OFF the power supply to the Controller during a download of the user program or the Controller Configurations and Setup.	
	The DB Connection settings are not correct because the power supply to the Controller was interrupted during a Clear All Memory operation.				Do not interrupt the power supply to the Controller during a Clear All Memory operation.	
	The DB Connection settings are not correct because the power supply to the Controller was interrupted during a Restore operation.				Do not interrupt the power supply to the Controller during a Restore operation.	
	Non-volatile memory failed.		If the error persists even after you make the above correction, replace the CPU Unit.		None	
Attached information	None					
Precautions/Remarks	None					

Event name	DB Server Certificate Error			Event code	35310000 hex	
Meaning	The format of the DB server certificate is invalid.					
Source	PLC Function Module		Source details	DB Connection Service	Detection timing	At download, power ON, Controller re-set, or restore
Error attributes	Level	Minor fault	Recovery	Transfer the DB Connection settings again and then start the DB Connection Service.	Log category	System
Effects	User program	Continues.	Operation	The DB Connection Service cannot be started. The operation status of the DB Connection Service is changed to "Error Stop".		
System-defined variables	Variable		Data type		Name	
	_DBC_Status		_sDBC_STATUS		DB Connection Service Status	
Cause and correction	Assumed cause		Correction		Prevention	
	A DB server certificate in the invalid format (X.509) was downloaded.		Set a DB server certificate in the correct format and transfer the DB Connection settings again from Sysmac Studio.			
	The power supply to the Controller was interrupted during a download of the DB Connection settings.		Transfer the DB Connection settings again from Sysmac Studio.		Do not turn OFF the power supply to the Controller during a download of the user program or the Controller Configurations and Setup.	
	The DB Connection settings are not correct because the power supply to the Controller was interrupted during a Clear All Memory operation.				Do not interrupt the power supply to the Controller during a Clear All Memory operation.	
	The DB Connection settings are not correct because the power supply to the Controller was interrupted during a Restore operation.				Do not interrupt the power supply to the Controller during a Restore operation.	
	Non-volatile memory failed.		If the error persists even after you make the above correction, replace the CPU Unit.		None	
Attached information	Attached information 1: DB Server Certificate File Name					
Precautions/Remarks	None					

Event name	DB Connection Disconnected Error			Event code	85100000 hex	
Meaning	The DB Connection was disconnected due to an error.					
Source	PLC Function Module		Source details	DB Connection Service	Detection timing	When a DB Connection Instruction is executed, or when Spool data is resent
Error attributes	Level	Minor fault	Recovery	Automatic recovery	Log category	System
Effects	User program	Continues.	Operation	Not affected.		
System-defined variables	Variable		Data type		Name	
	_DBC_Status		_sDBC_STATUS		DB Connection Service Status	
Cause and correction	Assumed cause		Correction		Prevention	
	The power supply to the server is OFF.		Check the server status and start it properly.		Check the server status and start it properly.	
	The DB is stopped in the server.					
	The Ethernet cable connector is disconnected.		Reconnect the connector and make sure it is mated correctly.		Connect the connector securely.	
	The Ethernet cable is broken.		Replace the Ethernet cable.		None	
	Noise		Implement noise countermeasures if there is excessive noise.		Implement noise countermeasures if there is excessive noise.	
Attached information	Attached information 1: DB Connection Name					
Precautions/Remarks	None					

Event name	DB Connection Service Started			Event code	95300000 hex	
Meaning	The DB Connection Service was started.					
Source	PLC Function Module		Source details	DB Connection Service	Detection timing	When the DB Connection Service is started
Error attributes	Level	Information	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	Not affected.		
System-defined variables	Variable		Data type		Name	
	_DBC_Status		_sDBC_STATUS		DB Connection Service Status	
Cause and correction	Assumed cause		Correction		Prevention	
	The DB Connection Service was successfully started.		---		---	
Attached information	Attached information 1: Start reason 01 hex: Execution of a DB_ControlService instruction or operation from Sysmac Studio 02 hex: Controller's operating mode change (from PROGRAM to RUN mode)					
Precautions/Remarks	None					

Event name	DB Connection Service Stopped			Event code	95310000 hex	
Meaning	The DB Connection Service was stopped.					
Source	PLC Function Module		Source details	DB Connection Service	Detection timing	When the DB Connection Service is stopped
Error attributes	Level	Information	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	Not affected.		
System-defined variables	Variable		Data type		Name	
	_DBC_Status		_sDBC_STATUS		DB Connection Service Status	
Cause and correction	Assumed cause		Correction		Prevention	
	The DB Connection Service was stopped.		---		---	
Attached information	Attached information 1: Stop reason 01 hex: Execution of a DB_ControlService instruction or operation from Sysmac Studio 02 hex: Controller's operating mode change (from RUN to PROGRAM mode) 03 hex: Execution of Synchronization (download), Clear All Memory, or Restore operation 04 hex: A major fault level Controller error					
Precautions/Remarks	None					

Event name	DB Connection Service Shutdown			Event code	95320000 hex	
Meaning	The DB Connection Service was shut down.					
Source	PLC Function Module		Source details	DB Connection Service	Detection timing	When the DB Connection Service is shut down.
Error attributes	Level	Information	Recovery	---	Log category	System

Effects	User program	Continues.	Operation	Not affected.
System-defined variables	Variable		Data type	Name
	_DBC_Status		_sDBC_STATUS	DB Connection Service Status
Cause and correction	Assumed cause		Correction	Prevention
	The DB Connection service was shut down.		---	---
Attached information	Attached information 1: Shutdown reason 01 hex: Execution of a DB_Shutdown instruction or operation from Sysmac Studio			
Precautions/Remarks	None			

Event name	Spool Memory Cleared			Event code	95330000 hex	
Meaning	The SQL statements was cleared from the spool memory.					
Source	PLC Function Module		Source details	DB Connection Service	Detection timing	When spooled data was cleared
Error attributes	Level	Information	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	Not affected.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	The SQL statements was cleared from the spool memory.		---		---	
Attached information	Attached information 1: DB Connection Name - when all spool memory cleared regardless of DB Connection. Attached information 2: Clear reason 01 hex: Execution of a DB_ControlSpool instruction or operation from Sysmac Studio 02 hex: The specified clear condition was met. 03 hex: The automatic clear condition was met.					
Precautions/Remarks	None					

Errors Related to DB Connection Instructions

Event name	DB Connection Service Not Started			Event code	54013000 hex	
Meaning	The DB Connection Service has not been started.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	A command to start the DB Connection Service was not given before the execution of relevant instruction.		Start the DB Connection Service. Or, correct the user program so that the relevant instruction is executed while the DB Connection Service is running.		Write the user program so that the relevant instruction is executed while the DB Connection Service is running.	
	A command to stop the DB Connection Service was given before the execution of relevant instruction.					
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)					
Precautions/Remarks	None					

Event name	DB Connection Service Run Mode Change Failed		Event code	54013001 hex	
Meaning	Failed to change the Run mode of the DB Connection Service.				
Source	PLC Function Module		Source details	Instruction	Detection timing At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.	
System-defined variables	Variable		Data type		Name
	None		---		---
Cause and correction	Assumed cause		Correction		Prevention
	Run mode change to "Test Mode" was executed by the relevant instruction while running in "Operation Mode".		Stop the DB Connection Service, and then execute the relevant instruction. Or, correct the user program so that the relevant instruction is executed when the operation status of the DB Connection Service is "Idle".		Write the user program so that the relevant instruction is executed when the operation status of the DB Connection Service is "Idle".
	Run mode change to "Operation Mode" was executed by the relevant instruction while running in "Test Mode".				
	Start of the DB Connection Service was commanded while the DB Connection Service was being stopped.		Execute the relevant instruction later.		While a DB_Insert, DB_Update, DB_Select, or DB_Delete instruction is being executed, the DB Connection Service becomes stopping status If stop of the DB Connection Service is commanded. Stop the DB Connection Service after completion of the DB_Insert, DB_Update, DB_Select, or DB_Delete instruction.
	Shutdown of the DB Connection Service was commanded while the DB Connection Service was being stopped.				
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)				
Precautions/Remarks	None				

Event name	DB Connection Service Shutdown or Shutting Down		Event code	54013002 hex	
Meaning	The DB Connection Service is already shut down or being shut down.				
Source	PLC Function Module		Source details	Instruction	Detection timing At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.	
System-defined variables	Variable		Data type		Name
	None		---		---
Cause and correction	Assumed cause		Correction		Prevention
	The relevant instruction was executed after the DB Connection Service was shut down.		Cycle the power supply to the Controller, start the DB Connection Service, and then execute the relevant instruction.		Write the user program so that the relevant instruction is not executed after the execution of DB_Shutdown instruction. Or, write the user program so that the relevant instruction is not executed after shutdown is commanded from Sysmac Studio.
	The relevant instruction was executed while the shutdown processing of the DB Connection Service was in progress.				
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)				
Precautions/Remarks	None				

Event name	Invalid DB Connection Name			Event code	54013003 hex	
Meaning	The specified DB Connection Name is not set in any DB Connection settings.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	The DB Connection Name specified in the <i>DBConnectionName</i> input variable of the relevant instruction is wrong.		Specify a correct DB Connection Name in the <i>DBConnectionName</i> input variable of the relevant instruction.		Confirm that a DB Connection Name is correctly specified in the <i>DBConnectionName</i> input variable of the relevant instruction.	
	The DB Connection Name set in the DB Connection settings is wrong.		Specify a correct DB Connection Name in the DB Connection settings.		Confirm that a DB Connection Name is correctly set in the DB Connection Settings.	

Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)
Precautions/Remarks	None

Event name	DB Connection Rejected			Event code	54013004 hex	
Meaning	The DB rejected the connection.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	The user name or password set in the DB Connection settings is wrong.		Enter the correct user name and password in the DB Connection settings.		Enter the correct user name and password in the DB Connection settings.	
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)					
Precautions/Remarks	None					

Event name	DB Connection Failed			Event code	54013005 hex	
Meaning	Failed to connect to the DB.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	A server does not exist for the specified IP address or the specified host name.		Enter the correct IP address or host name in the DB Connection settings.		Enter the correct IP address or host name in the DB Connection settings.	
	The power supply to the server is OFF.		Check the server status and start it properly.		Check the server status and start it properly.	
	The DB is stopped in the server.					
	The Ethernet cable connector is disconnected.		Reconnect the connector and make sure it is mated correctly.		Connect the connector securely.	
	The Ethernet cable is broken.		Replace the Ethernet cable.		None	
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)					

Precautions/ Remarks	None

Event name	DB Connection Already Established			Event code	54013006 hex	
Meaning	A same-name DB Connection is already established.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	The relevant instruction was executed when a same-name DB Connection was already established.		Correct the user program so that the relevant instruction is executed when the DB Connection is closed.		Write the user program so that the relevant instruction is executed when the DB Connection is closed.	
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)					
Precautions/Remarks	None					

Event name	Too Many DB Connections			Event code	54013007 hex	
Meaning	The number of DB Connections that can be established at the same time is exceeded.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	The relevant instruction was executed when the maximum number of DB Connections that can be established at the same time were already established.		Correct the user program so that the number of established DB Connections does not exceed the maximum number of DB Connections that can be established at the same time.		Write the user program so that the number of established DB Connections does not exceed the maximum number of DB Connections that can be established at the same time.	
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)					
Precautions/Remarks	None					

Event name	Invalid DB Connection			Event code	54013008 hex	
Meaning	The specified DB Connection is not correct, or the DB Connection is already closed.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	The DB Connection specified in the <i>DBConnection</i> input variable of the relevant instruction is wrong.		Specify a correct DB Connection in the <i>DBConnection</i> input variable of the relevant instruction.		Confirm that a correct DB Connection is specified in the <i>DBConnection</i> input variable of the relevant instruction.	
	The DB Connection specified in the <i>DBConnection</i> input variable of the relevant instruction is closed.		Correct the user program so that the relevant instruction is executed after the DB Connection is established by a DB_Connect instruction.		Write the user program so that the relevant instruction is executed after the DB Connection is established by a DB_Connect instruction.	
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)					
Precautions/Remarks	None					

Event name	Invalid DB Map Variable			Event code	54013009 hex	
Meaning	The specified DB Map Variable is not correct.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	A structure variable that contains a derivative data type of member was specified as a DB Map Variable.		Specify a basic data type for the members of the structure data used in the DB Map Variable.		Confirm the data type of the variables to be specified as a DB Map Variable when writing the user program.	
	A non-structure variable was specified as a DB Map Variable.		Specify a structure variable for the DB Map Variable.			
	A structure array variable was specified as a DB Map Variable for INSERT or UPDATE.		Specify a structure variable for the DB Map Variable for INSERT or UPDATE.			

Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)
Precautions/Remarks	None

Event name	Unregistered DB Map Variable			Event code	5401300A hex	
Meaning	The specified DB Map Variable has not been registered.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	The DB Map Variable has not been created by a DB_Create-Mapping instruction.		Correct the user program so that the relevant instruction is executed after the DB Map Variable is created by a DB_CreateMapping instruction.		Write the user program so that the relevant instruction is executed after the DB Map Variable is created by a DB_CreateMapping instruction.	
	A variable that is not registered as a DB Map Variable was specified in <i>MapVar</i> .		Check the input parameters of the relevant instruction and correct the user program.		In the input parameters of the relevant instruction, specify the DB Connection specified in the DB_CreateMapping instruction and the DB Map Variable created by the DB_CreateMapping instruction.	
	The DB Connection specified in the relevant instruction is different from the one specified at the execution of DB_CreateMapping instruction.					
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)					
Precautions/Remarks	None					

Event name	SQL Execution Error			Event code	5401300B hex	
Meaning	The executed SQL statement resulted in an error.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	

Cause and correction	Assumed cause	Correction	Prevention
	There is no column with the same name as a structure member of the DB Map Variable.	Check whether the column names match the structure member names of the DB Map Variable.	Confirm that the column names match the structure member names of the DB Map Variable.
	The table specified in the DB_CreateMapping instruction does not exist in the DB.	Check whether the table name specified in the DB_CreateMapping instruction is correct.	Confirm that the table name specified in the DB_CreateMapping instruction is correct.
	One or more structure member values of the DB Map Variable cannot be converted to the corresponding column's data type.	Check whether the data types of the structure members of the DB Map Variable can be converted to the corresponding column's data type.	Confirm that the data types of the structure members of the DB Map Variable can be converted to the corresponding column's data type.
	One or more column values cannot be converted to the corresponding structure member's data type of the DB Map Variable.	Check whether the data types of the columns can be converted to the corresponding structure member's data type of the DB Map Variable. Or, confirm that the values of the mapped columns are not NULL.	Check whether the data types of the columns can be converted to the corresponding structure member's data type of the DB Map Variable. Or, define the structure members so as not to map a column whose value can be NULL.
	One or more structure member values of the DB Map Variable exceed the valid range of the corresponding column's data type.	Check the structure member values of the DB Map Variable.	Write the user program so that the structure member values of the DB Map Variable are within the valid range of the corresponding column's data type.
	The column specified in the extraction condition does not exist in the DB's records. (DB_Select instruction, DB_Update instruction, DB_Delete instruction)	Confirm that the column name specified in the extraction condition is correct. Or, check whether the syntax of the extraction condition is correct.	Confirm that the column name specified in the extraction condition is correct. Or, confirm that the syntax of the extraction condition is correct.
	The extraction condition has a syntax error. (DB_Select instruction, DB_Update instruction, DB_Delete instruction)		
	The column specified in the sort condition does not exist in the DB's records. (DB_Select instruction)	Check whether the column name specified in the sort condition is correct. Or, check whether the syntax of the sort condition is correct.	Check whether the column name specified in the sort condition is correct. Or, confirm that the syntax of the sort condition is correct.
	The sort condition has a syntax error. (DB_Select instruction)		
	The user does not have the access rights to the table.	Check the access rights to the table.	Confirm the access rights to the table.
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)		
Precautions/Remarks	None		

Event name	Spool Capacity Exceeded			Event code	5401300C hex	
Meaning	The SQL statement could not be stored in the Spool memory because its maximum capacity was exceeded.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	The DB connection failure has been continuing due to network failure or other factors.		Recover from the network failure.		Control from the user program like below. Check the Spool memory usage using a DB_GetConnectionStatus instruction, and when the Spool memory usage has exceeded a certain value, do not execute the DB_Insert nor DB_Update instructions. Or, check the DB Connection status using a DB_GetConnectionStatus instruction, and when the status has changed to "Connected", resend the SQL statements stored in the Spool memory using a DB_ControlSpool instruction.	
	The resend processing of the SQL statements stored in the Spool memory has not been executed (when the Resend spool data parameter is set to "Manual").		Resend the SQL statements stored in the Spool memory using a DB_ControlSpool instruction after establishing the DB Connection again.		Check the DB Connection status using a DB_GetConnectionStatus instruction, and when the status has changed to "Connected", resend the SQL statements stored in the Spool memory using a DB_ControlSpool instruction.	
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)					
Precautions/Remarks	None					

Event name	Invalid Extraction Condition			Event code	5401300E hex	
Meaning	The entered extraction condition is invalid.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	A text string that consists of a NULL (16#00) character only was specified in the <i>Where</i> input variable.		Enter a text string that specifies the extraction condition in the <i>Where</i> input variable.		Enter a text string that specifies the extraction condition in the <i>Where</i> input variable.	
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)					
Precautions/Remarks	None					

Event name	Log Code Out of Range		Event code	54013010 hex		
Meaning	The value of the entered log code is outside the valid range.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	A value outside the valid range from 0 to 9999 was specified.		Correct the user program so that the log code is within the valid range from 0 to 9999.		Write the user program so that the log code is within the valid range from 0 to 9999.	
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)					
Precautions/Remarks	None					

Event name	DB Connection Disconnected Error Status		Event code	54013011 hex	
Meaning	The instruction could not be executed because the DB Connection had been disconnected due to an error.				
Source	PLC Function Module		Source details	Instruction	Detection timing At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.	
System-defined variables	Variable		Data type		Name
	None		---		---
Cause and correction	Assumed cause		Correction		Prevention
	The power supply to the server is OFF.		Check the server status and start it properly.		Check the server status and start it properly.
	The DB is stopped in the server.				
	The Ethernet cable connector is disconnected.		Reconnect the connector and make sure it is mated correctly.		Connect the connector securely.
	The Ethernet cable is broken.		Replace the Ethernet cable.		None
	Noise		Implement noise countermeasures if there is excessive noise.		Implement noise countermeasures if there is excessive noise.
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)				

Precautions/ Remarks	None

Event name	DB Connection Instruction Execution Timeout		Event code	54013012 hex		
Meaning	The instruction was not completed within the time specified for instruction execution timeout.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	The power supply to the server is OFF.		Check the server status and start it properly.		Check the server status and start it properly.	
	The Ethernet cable connector is disconnected.		Reconnect the connector and make sure it is mated correctly.		Connect the connector securely.	
	The Ethernet cable is broken.		Replace the Ethernet cable.		None	
	The server's processing time is long.		Check the server's response time in the Debug Log and change the timeout parameter to an appropriate value.		Check the server's response time in the Debug Log and specify an appropriate value in the timeout parameter.	
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)					
Precautions/Remarks	None					

Event name	DB Connection Service Error Stop			Event code	54013013 hex	
Meaning	The instruction could not be executed because the DB Connection Service was stopped due to an error.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	The DB Connection settings are corrupted.		Transfer the DB Connection settings again using the synchronization function of Sysmac Studio.		Do not interrupt the power supply to the Controller during a download of the DB Connection settings.	
Attached information	Attached information 1: Error Location					
	Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given.					
	Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given.					
	Attached information 4: Expansion Error Code (ErrorIDEx)					

**Precautions/
Remarks**

None

Event name	Data Already Spooled			Event code	54013014 hex	
Meaning	One or more SQL statements are already stored in the Spool memory.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	A DB_Insert or DB_Update instruction was executed when one or more SQL statements were already stored in the Spool memory.		None		None	
	A DB_Select or DB_Delete instruction was executed when one or more SQL statements were already stored in the Spool memory.		Execute the instruction again after the resend processing of the SQL statements stored in the Spool memory is completed.		Execute the relevant instruction when no SQL statements are stored in the Spool memory.	
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)					
Precautions/Remarks	None					

Event name	DB Connection Service Initializing			Event code	54013015 hex	
Meaning	The instruction could not be executed because the initialization processing of the DB Connection Service is in progress.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	_DBC_Status		_sDBC_STATUS		DB Connection Service Status	
Cause and correction	Assumed cause		Correction		Prevention	
	The relevant instruction was executed during the initialization processing of the DB Connection Service.		Execute the relevant instruction after the operation status of the DB Connection Service changes to Running or Idle.		Execute the relevant instruction after confirming the operation status of the DB Connection Service with the <i>_DBC_Status</i> system-defined variable.	

Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)
Precautions/Remarks	None

Event name	DB in Process			Event code	54013016 hex	
Meaning	The instruction could not be executed because the DB is under processing in the server.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	Though a DB Connection Instruction Execution Timeout occurred for the previous instruction, the relevant instruction was executed before completion of the DB's processing in the server.		Re-execute the relevant instruction from the user program. However, if you execute a DB_Insert or DB_Update instruction and the Spool function is enabled, you do not have to re-execute the relevant instruction because the SQL statement will be stored in the Spool memory.		Estimate the processing time of the DB in the server and adjust the execution timing of the DB Connection Instruction to an appropriate frequency.	
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)					
Precautions/Remarks	None					

Event name	Operation Log Disabled			Event code	54013017 hex	
Meaning	The log could not be recorded because the specified Operation Log is disabled.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	Though Execution Log was specified in the <i>LogType</i> input variable, the Execution Log is disabled.		Enable the Execution Log in the DB Connection Service settings.		Execute the instruction when the Execution Log is enabled.	
	Though Debug Log was specified in the <i>LogType</i> input variable, recording to the Debug Log is stopped.		Start recording to the Debug Log using a DB_ControlService instruction. Or, start recording to the Debug Log from Sysmac Studio.		Execute the instruction after the recording to the Debug Log is started.	

Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)
Precautions/Remarks	None

Event name	Invalid Procedure Handle			Event code	54013018 hex	
Meaning	The specified procedure handle is invalid.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	The procedure handle specified in the <i>ProcHandle</i> input variable of the relevant instruction is wrong.		Specify a correct procedure handle in the <i>ProcHandle</i> input variable of the relevant instruction.		Confirm that a correct procedure handle is specified in the <i>ProcHandle</i> input variable of the relevant instruction.	
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)					
Precautions/Remarks	None					

Event name	Instruction Executed for Unsupported Database Type		Event code	54013019 hex	
Meaning	The instruction was executed for a database type that is not supported by this instruction.				
Source	PLC Function Module		Source details	Instruction	Detection timing At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.	
System-defined variables	Variable		Data type		Name
	None		---		---
Cause and correction	Assumed cause		Correction		Prevention
	The database type specified in the DB Connection Settings is not supported by the relevant instruction.		Correct the database specified in the DB Connection Settings to a supported database type.		Confirm that the database type specified in the DB Connection Settings is a supported database type.
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)				
Precautions/Remarks	None				

Event name	Invalid Stored Procedure Name		Event code	5401301A hex	
Meaning	The specified stored procedure name does not exist.				
Source	PLC Function Module		Source details	Instruction	Detection timing At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.	
System-defined variables	Variable		Data type		Name
	None		---		---
Cause and correction	Assumed cause		Correction		Prevention
	The stored procedure name specified in the <i>ProcName</i> input variable of the relevant instruction does not exist in the database on the server. Note This includes when the specified stored procedure name does not find on the DB. Even if the stored procedure name exists, the DB Connection Service cannot find the stored procedure name due to the reason that the user does not have the access right to the stored procedure, or other reasons.		Specify an existing stored procedure name in the <i>ProcName</i> input variable of the relevant instruction.		Confirm that an existing stored procedure name is specified in the <i>ProcName</i> input variable of the relevant instruction.
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)				
Precautions/Remarks	None				

Event name	Invalid Stored Procedure Argument			Event code	5401301B hex	
Meaning	The attached argument information does not match the argument of the stored procedure.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	The name, number, and type of the stored procedure argument data that is retrieved from the server-side database do not match those of the input variables <i>ArgIn</i> , <i>ArgOut</i> , and <i>ArgInOut</i> of the relevant instruction.		Make sure that the name, number, and type of the stored procedure argument data that is retrieved from the server-side database match those of the input variables <i>ArgIn</i> , <i>ArgOut</i> , and <i>ArgInOut</i> of the relevant instruction.		Confirm that the name, number, and type of the stored procedure argument data that is retrieved from the server-side database match those of the input variables <i>ArgIn</i> , <i>ArgOut</i> , and <i>ArgInOut</i> of the relevant instruction.	
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)					
Precautions/Remarks	None					

Event name	Invalid Number of Columns for Stored Procedure Result Set		Event code	5401301C hex	
Meaning	The number of columns in the stored procedure result set do not match the number of structure variable members where the result is stored.				
Source	PLC Function Module		Source details	Instruction	Detection timing At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.	
System-defined variables	Variable		Data type		Name
	None		---		---
Cause and correction	Assumed cause		Correction		Prevention
	The number of columns in the result set retrieved by the relevant instruction do not match the number of structure variable members where the result is stored.		Make sure that the number of columns in the result set retrieved by the relevant instruction match the number of structure variable members where the result is stored.		Confirm that the number of columns in the result set retrieved by the relevant instruction match the number of structure variable members where the result is stored.
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)				
Precautions/Remarks	None				

Event name	Invalid Stored Procedure Execution			Event code	5401301D hex	
Meaning	An error occurred when a stored procedure is executed. Troubleshoot the error according to the attached information 4.					
Source	PLC Function Module		Source details	Instruction	Detection timing	At instruction execution
Error attributes	Level	Observation	Recovery	---	Log category	System
Effects	User program	Continues.	Operation	The relevant instruction will end according to specifications.		
System-defined variables	Variable		Data type		Name	
	None		---		---	
Cause and correction	Assumed cause		Correction		Prevention	
	Check the cause according to the attached information 4 and the manual.		Remove the cause of the error according to the attached information 4 and the manual.		Confirm that the cause is removed according to the attached information 4 and the manual.	
Attached information	Attached information 1: Error Location Attached information 2: Error Location Detail, Rung Number. For a program section, the rung number from the start of the section is given. For ST, the line number is given. Attached information 3: Instruction Name and Instruction Instance Name Where Error Occurred. If there is more than one instruction, all of them are given. If the instruction cannot be identified, nothing is given. Attached information 4: Expansion Error Code (ErrorIDEx)					
Precautions/Remarks	None					

A

Appendix

A-1	Task Design Procedure.....	A-2
A-1-1	Startup Time of DB Connection Service	A-2
A-1-2	Reference Values for Execution Time of DB Connection Instructions	A-4
A-1-3	How to Measure Execution Time of DB Connection Instructions	A-11
A-1-4	Guideline for System Service Execution Time Ratio	A-11
A-1-5	Checking the System Service Execution Time Ratio	A-13
A-2	Execution Time of DB Connection Instructions.....	A-15
A-2-1	Restrictions to Execution Time of DB Connection Instructions.....	A-15
A-2-2	Impact of Operation Log Recording on Execution Time of DB Connection Instructions	A-22
A-2-3	How to Measure DB Response Time.....	A-23
A-2-4	Ensuring Equipment Performance (Takt Time) by Monitoring Instruction Execution Timeout.....	A-23
A-3	Specifications	A-25
A-3-1	General Specifications	A-25
A-3-2	Performance Specifications	A-25
A-3-3	Function Specifications	A-25
A-4	Version Information	A-26
A-4-1	Unit Versions and Corresponding DB Connection Service Versions	A-26
A-4-2	DB Connection Functions that were Added or Changed for Each Unit Version	A-26
A-4-3	Unit Version, DB Connection Service Version, and Unit Version Set in the Sysmac Studio Project	A-27
A-4-4	DB Connection Service Version and Connection Database Types After Changing Devices	A-30
A-4-5	DB Connection Service Versions and Connection Database Types/Versions.....	A-31

A-1 Task Design Procedure

This section describes the task design procedure for using the DB Connection function.

Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for task and system service operation specifications of the NJ/NX-series Controllers.

A-1-1 Startup Time of DB Connection Service

The time required to get the DB Connection Service ready for operation (i.e. until the `_DBC_Status.Run` system-defined variable changes to True) after turning ON the power supply to the CPU Unit (hereinafter called "startup time") depends on the database type to connect and the percentage of task execution time.

The following table shows the reference values for some combinations.

Please design your system in reference to these values.

- NX701-1620

DB type	Reference value for startup time of the DB Connection Service (Average)
Oracle	9.49 s
SQL Server	8.33 s
DB2	8.84 s
MySQL	8.32 s
Firebird	8.32 s
PostgreSQL	8.32 s

- NX102-1220

DB type	Percentage of task execution time*1	Reference value for startup time of the DB Connection Service (Average)
Oracle	50%	70.85 s
	80%	74.37 s
SQL Server	50%	58.41 s
	80%	61.16 s
DB2	50%	70.77 s
	80%	72.56 s
MySQL	50%	56.52 s
	80%	59.52 s
FireBird	50%	60.72 s
	80%	61.77 s
PostgreSQL	50%	57.30 s
	80%	61.55 s

*1. Percentage of task execution time on the Task Execution Time Monitor of Sysmac Studio. The load during task execution was added as part of a simulation.

- NJ501-1520

DB type	Percentage of task execution time*1	Reference value for startup time of the DB Connection Service (Average)
Oracle	50%	58.43 s
	80%	124.95 s
SQL Server	50%	54.02 s
	80%	120.95 s
DB2	50%	56.26 s
	80%	128.49 s
MySQL	50%	57.41 s
	80%	131.33 s
Firebird	50%	56.65 s
	80%	129.07 s
PostgreSQL	50%	59.06 s
	80%	124.26 s

*1. Percentage of task execution time on the Task Execution Time Monitor of Sysmac Studio. The load during task execution was added as part of a simulation.

• NJ101-1020

DB type	Percentage of task execution time*1	Reference value for startup time of the DB Connection Service (Average)
Oracle	50%	75.59 s
	60%	89.31 s
SQL Server	50%	56.36 s
	60%	67.17 s
DB2	50%	61.90 s
	60%	73.35 s
MySQL	50%	54.46 s
	60%	66.83 s
Firebird	50%	57.61 s
	60%	70.98 s
PostgreSQL	50%	63.61 s
	60%	76.63 s

*1. Percentage of task execution time on the Task Execution Time Monitor of Sysmac Studio. The load during task execution was added as part of a simulation.

The following table shows the measurement conditions and items.

Measurement conditions		Description
Item	Subitem	
CPU Unit	Task composition	Primary periodic task only Task period: 1 ms
System configuration	Basic configuration	- No EtherCAT network - No CJ-series Units - USB connection with Sysmac Studio*1
	Network configuration	- No connection with other controllers - No connection with HMI

*1. For NX102-1220, hub connection with the built-in EtherNet/IP port 1



Precautions for Correct Use

- The DB Connection Service is executed as a system service. Therefore, the execution time of each processing may require time if the startup processing of the DB Connection Service and other system service processing are executed at the same time.
- The guidelines for the system service execution time ratio depend on the models. Therefore, the measurement condition of the task execution time ratio varies depending on the models. Refer to *A-1-4 Guideline for System Service Execution Time Ratio* on page A-11 for details of the guidelines.

A-1-2 Reference Values for Execution Time of DB Connection Instructions

The DB Connection Instructions are function block type of instructions that are executed over multiple task periods.

The following table gives the reference values for execution time of each DB Connection Instruction. Refer to *A-2-1 Restrictions to Execution Time of DB Connection Instructions* on page A-15 for the factors that fluctuate execution time of DB Connection Instructions.

- The following table specifies the measurement names, applicable instructions, and measurement methods that are used in the reference value list in the subsequent sections.

Measurement name	Applicable Instructions	Conditions	Remarks
DB_Insert	DB_Insert	When executing an INSERT operation for 100-column record	
DB_Select	DB_Select	When searching for one record from 100,000 records and retrieving 100-column data	The primary key is specified for the retrieval condition.
DB_BatchInsert_100	DB_BatchInsert	When executing BATCHINSERT for 100 records	One record contains 100 columns
DB_BatchInsert_500		When executing BATCHINSERT for 500 records	*Same as INSERT

Measurement Condition 1

The following table shows the measurement conditions and items.

Measurement conditions		Description
Item	Subitem	
CPU Unit	Task composition	Primary periodic task only Task period: 1 ms Percentage of task execution time to the task period: 80%, 50%

Measurement conditions		Description
Item	Subitem	
Server	Computer	CPU: Intel Xeon(R) CPU E31220 @ 3.10 GHz, Quad-Core Memory: 8.00 GB
	Operating system	Windows Server 2008 Standard 64bit
	DB type	Oracle Database Express Edition 12c SQL Server 2012 MySQL Community Edition 5.6 DB2 for Linux and Windows 10.5 Firebird 2.5 PostgreSQL 9.4
SQL statement to execute	Record composition	INT: 40 columns REAL: 40 columns STRING[16]: 16 columns DATE_AND_TIME: 4 columns
Operation Logs	Execution Log	Recorded
	Debug Log	Stopped
	SQL Execution Failure Log	Not recorded

● NX701-1620

DB type	Measurement name	Reference value for instruction execution time (ms)	
		Average	Maximum
Oracle Database 11g	DB_Insert	2.4	12
	DB_Select	5.11	22
SQL Server 2012	DB_Insert	2.35	13
	DB_Select	2.62	21
MySQL 5.6 Storage engine: InnoDB	DB_Insert	15.76	74
	DB_Select	2.21	10
FireBird 2.5	DB_Insert	12.95	32
	DB_Select	21.22	32
DB2 10.5	DB_Insert	3.44	21
	DB_Select	7.39	20
PostgreSQL 9.4	DB_Insert	2.93	21
	DB_Select	5.16	14

● NX102-1120

DB type	Percentage of task execution time	Measurement name	Reference value for instruction execution time (ms)	
			Average	Maximum
Oracle Database 11g	50%	DB_Insert	16.53	109
		DB_Select	23.02	244
	80%	DB_Insert	20.15	119
		DB_Select	22.56	168
SQL Server 2012	50%	DB_Insert	15.64	114
		DB_Select	16.36	107
	80%	DB_Insert	19.31	117
		DB_Select	18.98	89

DB type	Percentage of task execution time	Measurement name	Reference value for instruction execution time (ms)	
			Average	Maximum
MySQL 5.6 Storage engine: InnoDB	50%	DB_Insert	31.73	137
		DB_Select	12.77	114
	80%	DB_Insert	35.48	118
		DB_Select	13.58	100
FireBird 2.5	50%	DB_Insert	24.71	124
		DB_Select	56.92	152
	80%	DB_Insert	26.99	115
		DB_Select	60.22	135
DB2 10.5	50%	DB_Insert	20.78	130
		DB_Select	29.26	350
	80%	DB_Insert	24.49	127
		DB_Select	32.98	333
PostgreSQL 9.4	50%	DB_Insert	16.52	119
		DB_Select	22.28	113
	80%	DB_Insert	19.69	117
		DB_Select	23.86	112

Measurement Condition 2

The following table shows the measurement conditions and items.

Measurement conditions		Description
Item	Subitem	
CPU Unit	Task composition	Primary periodic task only Task period: 1 ms Percentage of task execution time to the task period: 80%, 50%
Server	Computer	CPU: Intel Xeon(R) CPU E5-2603 @ 1.7 GHz, 6 Core Memory: 32.00 GB
	Operating system	Windows Server 2016 Standard 64bit
	DB type	Oracle Database Express Edition 12c SQL Server 2016 MySQL Community Edition 5.6 DB2 for Linux and Windows 10.5 Firebird 2.5 PostgreSQL 9.4
SQL statement to execute	Record composition	INT: 40 columns REAL: 40 columns STRING[16]: 16 columns DATE_AND_TIME: 4 columns
Operation Logs	Execution Log	Recorded
	Debug Log	Stopped
	SQL Execution Failure Log	Not recorded

● NX701-1620

DB type	Measurement name	Reference value for instruction execution time (ms)	
		Average	Maximum
Oracle Database 11g	DB_Insert	2.21	11
	DB_Select	6.07	14
	DB_BatchInsert_100	16.2	67
	DB_BatchInsert_500	74.6	397
SQL Server 2012	DB_Insert	2.49	8
	DB_Select	2.69	11
	DB_BatchInsert_100	29.47	44
	DB_BatchInsert_500	133	219
MySQL 5.6 Storage engine: InnoDB	DB_Insert	3.02	11
	DB_Select	2.09	10
	DB_BatchInsert_100	158	178
	DB_BatchInsert_500	775	840
Firebird 2.5	DB_Insert	5.29	20
	DB_Select	22.62	57
DB2 10.5	DB_Insert	3.11	11
	DB_Select	10.36	22
PostgreSQL 9.4	DB_Insert	2.48	10
	DB_Select	6.75	14
	DB_BatchInsert_100	69.79	105
	DB_BatchInsert_500	338	415

● NX102-1120

DB type	Percentage of task execution time	Measurement name	Reference value for instruction execution time (ms)	
			Average	Maximum
Oracle Database 11g	50%	DB_Insert	16.35	119
		DB_Select	20.54	186
		DB_BatchInsert_100	158	224
		DB_BatchInsert_500	745	921
	80%	DB_Insert	19.32	118
		DB_Select	22.55	170
		DB_BatchInsert_100	175	310
		DB_BatchInsert_500	775	964
SQL Server 2012	50%	DB_Insert	15.4	107
		DB_Select	15.84	114
		DB_BatchInsert_100	234	328
		DB_BatchInsert_500	1134	1369
	80%	DB_Insert	19	115
		DB_Select	19.19	118
		DB_BatchInsert_100	255	354
		DB_BatchInsert_500	1162	1347

DB type	Percentage of task execution time	Measurement name	Reference value for instruction execution time (ms)	
			Average	Maximum
MySQL 5.6 Storage engine: InnoDB	50%	DB_Insert	17.27	113
		DB_Select	12.42	107
		DB_BatchInsert_100	520	619
		DB_BatchInsert_500	2564	2966
	80%	DB_Insert	19.87	110
		DB_Select	13.64	106
		DB_BatchInsert_100	525	610
		DB_BatchInsert_500	2676	3150
Firebird 2.5	50%	DB_Insert	21.6	121
		DB_Select	56.94	402
	80%	DB_Insert	24.98	120
		DB_Select	60.6	580
DB2 10.5	50%	DB_Insert	19.76	124
		DB_Select	31.54	335
	80%	DB_Insert	23.47	120
		DB_Select	35.19	204
PostgreSQL 9.4	50%	DB_Insert	16.61	116
		DB_Select	22.56	120
		DB_BatchInsert_100	198	587
		DB_BatchInsert_500	926	1162
	80%	DB_Insert	19.2	107
		DB_Select	25.16	123
		DB_BatchInsert_100	204	590
		DB_BatchInsert_500	958	1329

Measurement Condition 3

The following table shows the measurement conditions and items.

Measurement conditions		Description
Item	Subitem	
CPU Unit	Task composition	Primary periodic task only Task period: 1 ms Percentage of task execution time to the task period: 80%, 50%
Server	Computer	CPU: Intel Xeon(R) CPU E31220 @ 3.10 GHz 3.09 GHz Memory: 8.00 GB
	Operating system	Windows Server 2008 Standard SP2 64 bits
	DB type	Oracle Database Express Edition 11g 11.2.0 SQL Server 2012 DB2 for Linux, UNIX and Windows 10.5 MySQL Community Edition 5.6 Firebird 2.5 PostgreSQL 9.4
SQL statement to execute	Record composition	INT: 40 columns REAL: 40 columns STRING[16]: 16 columns DATE_AND_TIME: 4 columns

Measurement conditions		Description
Item	Subitem	
Operation Logs	Execution Log	Recorded
	Debug Log	Stopped
	SQL Execution Failure Log	Not recorded

● NJ501-1520

DB type	Percentage of task execution time	Measurement name	Reference value for instruction execution time (ms)	
			Average	Maximum
Oracle Database 11g	50%	DB_Insert	16.2	65
		DB_Select	37.1	75
	80%	DB_Insert	49.2	184
		DB_Select	101.6	272
SQL Server 2012	50%	DB_Insert	16.1	57
		DB_Select	23.8	98
	80%	DB_Insert	45.5	112
		DB_Select	72.5	236
DB2 10.5	50%	DB_Insert	27.5	115
		DB_Select	37.1	80
	80%	DB_Insert	69.4	176
		DB_Select	99.5	352
MySQL 5.6 Storage engine: InnoDB	50%	DB_Insert	40.3	273
		DB_Select	32.0	41
	80%	DB_Insert	65.0	315
		DB_Select	69.4	164
Firebird 2.5	50%	DB_Insert	23.8	156
		DB_Select	71.7	153
	80%	DB_Insert	52.8	139
		DB_Select	118.4	234
PostgreSQL 9.4	50%	DB_Insert	17.0	78
		DB_Select	30.9	83
	80%	DB_Insert	48.3	175
		DB_Select	89.1	250

Measurement Condition 4

The following table shows the measurement conditions and items.

Measurement conditions		Description
Item	Subitem	
CPU Unit	Task composition	Primary periodic task only Task period: 1 ms Percentage of task execution time to the task period: 60%, 50%

Measurement conditions		Description
Item	Subitem	
Server	Computer	CPU: Intel Xeon(R) CPU E5-2603 @ 1.7 GHz, 6 Core Memory: 32.00 GB
	Operating system	Windows Server 2016 Standard 64bit
	DB type	Oracle Database Express Edition 11g 11.2.0 SQL Server 2016 DB2 for Linux, UNIX and Windows 10.5 MySQL Community Edition 5.6 Firebird 2.5 PostgreSQL 9.4
SQL statement to execute	Record composition	INT: 40 columns REAL: 40 columns STRING[16]: 16 columns DATE_AND_TIME: 4 columns
Operation Logs	Execution Log	Recorded
	Debug Log	Stopped
	SQL Execution Failure Log	Not recorded

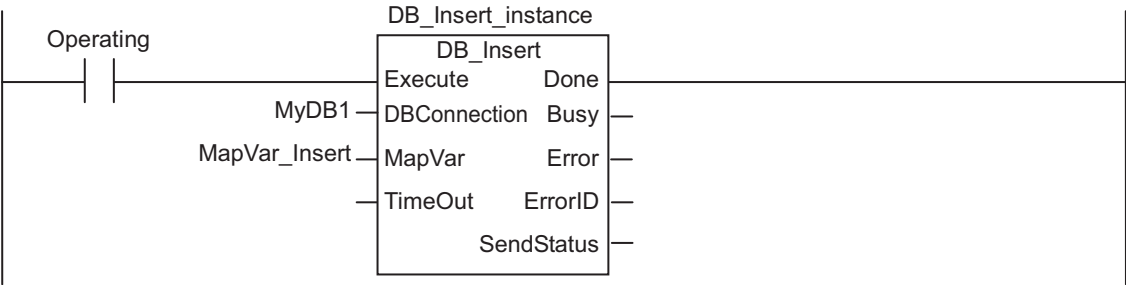
● NJ101-9020

DB type	Percentage of task execution time	Measurement name	Reference value for instruction execution time (ms)	
			Average	Maximum
Oracle Database 11g	50%	DB_Insert	27.8	311
		DB_Select	42.0	311
	60%	DB_Insert	39.0	342
		DB_Select	62.4	369
SQL Server 2012	50%	DB_Insert	26.7	287
		DB_Select	36.2	626
	60%	DB_Insert	37.5	621
		DB_Select	52.1	456
DB2 10.5	50%	DB_Insert	39.8	544
		DB_Select	59.0	467
	60%	DB_Insert	52.3	397
		DB_Select	81.0	655
MySQL 5.6 Storage engine: InnoDB	50%	DB_Insert	44.2	365
		DB_Select	36.0	599
	60%	DB_Insert	54.6	834
		DB_Select	52.4	450
Firebird 2.5	50%	DB_Insert	34.0	314
		DB_Select	78.4	403
	60%	DB_Insert	45.4	388
		DB_Select	101.0	472
PostgreSQL 9.4	50%	DB_Insert	28.7	306
		DB_Select	45.0	291
	60%	DB_Insert	41.3	471
		DB_Select	66.0	433

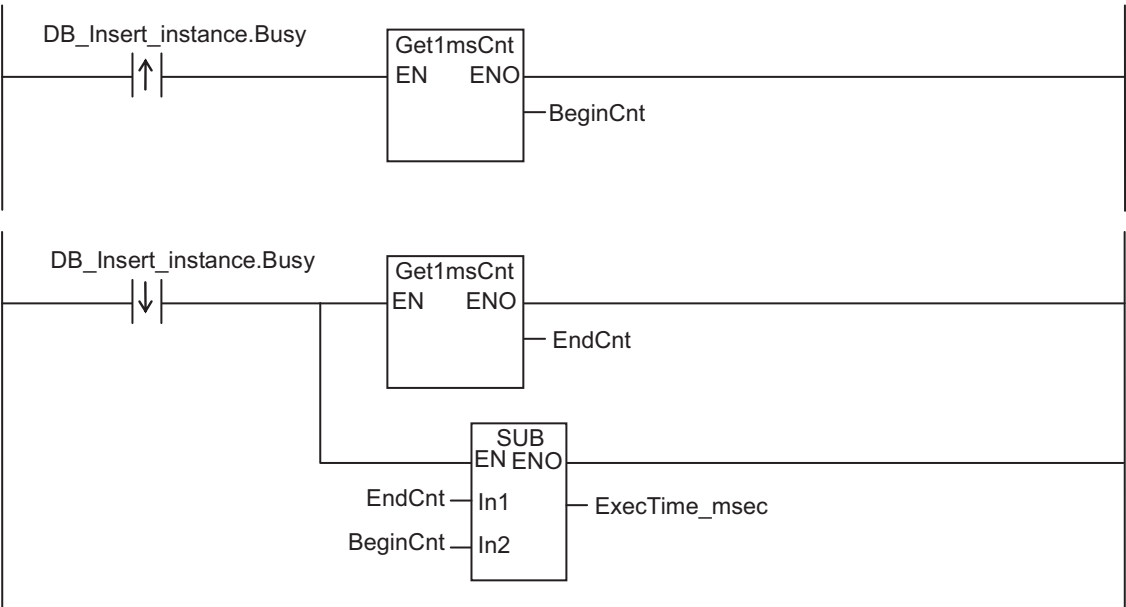
A-1-3 How to Measure Execution Time of DB Connection Instructions

The execution time of DB Connection Instructions can be measured by a Get1msCnt instruction. The instruction calculates the value of free-running counter of the cycle from when the Busy output variable changes to TRUE to when the variable changes to FALSE.

- Example for measuring execution time of a DB_Insert instruction
Insert a record to the DB Connection *MyDB1*.



Measure execution time of the DB_Insert instruction and output the result to the ExecTime_msec output variable of the SUB instruction.



A-1-4 Guideline for System Service Execution Time Ratio

The DB Connection Service is executed as a system service.
When a DB Connection Instruction is executed by a user program, the DB Connection Service executes the processing as a system service.
The method of executing the system service depends on the CPU Unit model.



Precautions for Safe Use

The above system service execution time ratio (CPU usage) is just a guideline.
The value of system service execution time ratio (CPU usage) depends on the usage of other services executed as a system service.
Before starting actual operation, you must test performance under all foreseeable conditions on the actual system and make sure that the DB Connection Instructions are executed within the appropriate execution time.

● NJ501-□□20 or NJ101-□□20

For NJ501-□□20 and NJ101-□□20, if sufficient execution time cannot be allocated to the system services, the DB Connection Instruction may take long execution time. Or, other processing executed in the system services may take long execution time. To execute the DB Connection Instructions according to the performance specifications, design the task so that the system service execution time ratio (CPU usage) meets the following.

CPU Unit model	Guideline for system service execution time ratio
NJ501-□□20	20% or greater
NJ101-□□20	40% or greater



Precautions for Correct Use

- If the system service execution time ratio is reduced, operation failures or communications errors may occur when each operation is executed from Sysmac Studio. If an operation failure or communications error occurs when you execute an operation from Sysmac Studio, retry the operation after performing the following:
 - a) Check the cable connection.
 - b) Check the communications settings.
 - c) Increase the response monitoring time in the Communications Setup.
 - d) Check that the operation status of the DB Connection Service is not "Initializing", "Error", or "Shutdown".

For details of the operation status of the DB Connection Service, refer to *4-3-1 Operation Status of the DB Connection Service* on page 4-7.

- When Sysmac Studio cannot go online, refer to the *NJ/NX-series Troubleshooting Manual (Cat. No. W503)*.
- If the time set for system service monitoring cannot be secured for system services, an "Insufficient System Service Time Error" will occur. "The error" is a major fault level Controller error. When the error has occurred, user program execution stops. To secure enough time for system services and task execution, set the minimum value that can satisfy the response performance of the system service processing for system service monitoring. The system service monitoring setting is just for monitoring whether or not the specified time can be secured for system service execution. It does not guarantee that system services are executed for the specified time.
- The system service execution time is affected by task execution time and tag data links. Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for details of task specifications, tag data link service, and system services.
- The startup processing of a CJ-series EtherNet/IP Unit depends on the system service execution time when the power supply to the CPU Unit is turned ON. For the NJ101-□□20, the system service execution time may be insufficient when the power supply to the CPU Unit is turned ON. Therefore, if you use a CJ-series EtherNet/IP Unit with the NJ101-□□20, check the startup operation in advance.

If the system service execution time is not sufficient when the power supply to the CPU Unit is turned ON, review the setting of **Start delay time at startup**.

Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for details on the setting of **Start delay time at startup**.

Refer to the *CJ-series EtherNet/IP Units Operation Manual for NJ-series CPU Unit (Cat. No. W495)* for details on the CJ-series EtherNet/IP Units.

● NX701-□□20

For NX701-□□20 the system services are executed at any time without being affected by tasks and tag data link services.

Therefore, there is no shortage of system service execution time.

● NX102-□□20

For NX102-□□20, the system services are executed without being affected by the tasks.
However, the system services will not be executed while the tag data link service is running.

A-1-5 Checking the System Service Execution Time Ratio

For NJ501-□□20 and NJ101-□□20, when you design the tasks, confirm that sufficient execution time can be allocated to system services by the following methods.

● Desktop Calculations

This is an example for a project that consists of one primary periodic task.

Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* to make a rough estimate of the "average task execution time" on paper.

- NJ501-□□20
"Average task execution time " < "Task period" x 0.8
- NJ101-□□20
"Average task execution time " < "Task period" x 0.6

Design the task using the following as a guideline:

● Estimating with the Simulator on Sysmac Studio

Check the value of "Estimated CPU usage rate" with the "Task Execution Time Monitor" of the Simulator on Sysmac Studio.

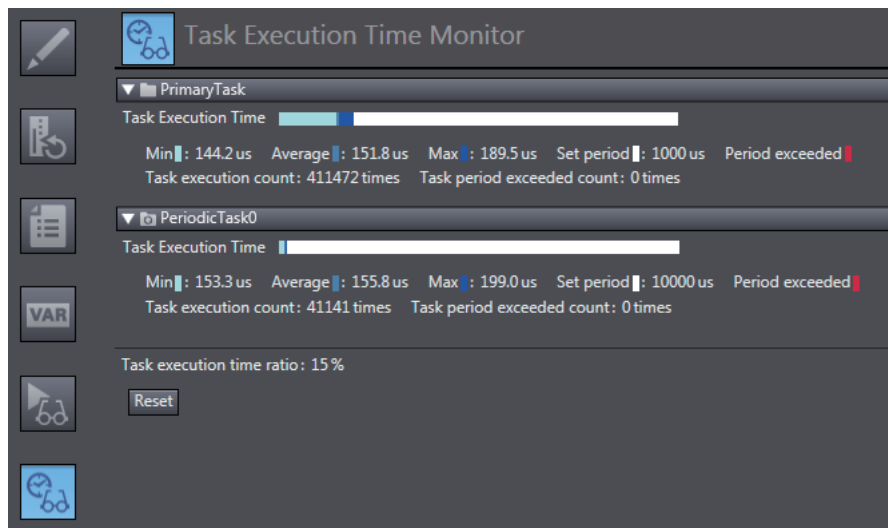
Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for the procedure to check the operation in the Simulator.

- NJ501-□□20
"Estimated CPU usage rate" - "System service execution ratio" < 80%
- NJ101-□□20
"Estimated CPU usage rate" - "System service execution ratio" < 60%

Design the task using the following as a guideline:

The "Estimated CPU usage rate" shows the percentage of the total of the following times in the task period: Estimated maximum value of the task processing time + Tag data link service execution ratio + Required system service processing time for system service monitoring.

The value found by subtracting the "System service execution ratio" from the "Estimated CPU usage rate" is the percentage for the execution time of processing other than system services.



● Calculating Times on the Physical Controller

When the project consists of one primary periodic task, check the "average task execution time" using the "Task Execution Time Monitor function" of Sysmac Studio while online with the physical Controller.

- NJ501-□□20
"Average task execution time " < "Task period" x 0.8
- NJ101-□□20
"Average task execution time " < "Task period" x 0.6

Design the task using the following as a guideline:

When the project consists of multiple tasks, test performance under all foreseeable conditions on the actual system and make sure that the DB Connection Instructions are executed within the appropriate execution time before starting actual operation.

A-2 Execution Time of DB Connection Instructions

This section describes execution time of DB Connection Instructions.

A-2-1 Restrictions to Execution Time of DB Connection Instructions

Execution time of DB Connection Instructions varies according to the following factors.

- Status of the NJ/NX-series CPU Unit
- DB type
- Processing capability and load status of the server that contains the DB
- DB response time
- Contents of the SQL statement to execute
- Number of retrieved records in the execution of DB_Select instruction
- Using the encrypted communication function

Due to the above factors, execution time of a DB Connection Instruction may exceed the reference value given in *A-1-2 Reference Values for Execution Time of DB Connection Instructions* on page A-4. The following table lists the phenomena that we confirmed under our measurement environment and their countermeasures.

No.	Phenomena
1	After the power supply to the CPU Unit was turned ON, execution time of the first DB Connection Instruction (i.e. DB_Insert, DB_Update, DB_Select, or DB_Delete instruction) got longer.
2	After execution of a DB_CreateMapping instruction, execution time of the first DB_Insert instruction got longer.
3	When communications or SD Memory Card processing was executed in the CPU Unit, execution time of a DB Connection Instruction got longer.
4	Execution time of DB Connection Instructions is steadily long.
5	Depending on the DB's status, execution time of a DB Connection Instruction (i.e., DB_Insert, DB_Update, DB_Select, or DB_Delete instruction) got longer.

Refer to *A-1-2 Reference Values for Execution Time of DB Connection Instructions* on page A-4 for the measurement conditions and items.

Phenomenon 1: After the Power Supply to the CPU Unit was Turned ON, Execution Time of the First DB Connection Instruction (i.e. DB_Insert, DB_Update, DB_Select, or DB_Delete instruction) Got Longer

● Possible causes

The following can be the causes:

- For the first record processing executed after the power supply to the CPU Unit is turned ON, the CPU Unit may require longer processing time than usual.
- For the first DB_Insert instruction that is executed after execution of a DB_CreateMapping instruction, the DB may require longer processing time than usual.

The following table gives the reference values for execution time of the first DB Connection Instruction after the power supply to the CPU Unit is turned ON. The percentage of task execution time is 50%.

- NX701-□□20

DB type	Instruction	Reference value for instruction execution time	Measurement condition
Oracle Database 11g	DB_Insert	99 ms	When executing an INSERT operation for 100-column record
	DB_Select	72 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
SQL Server 2012	DB_Insert	312 ms	When executing an INSERT operation for 100-column record
	DB_Select	63 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
DB2 10.5	DB_Insert	145 ms	When executing an INSERT operation for 100-column record
	DB_Select	395 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
MySQL 5.6 Storage engine: InnoDB	DB_Insert	130 ms	When executing an INSERT operation for 100-column record
	DB_Select	245 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
Firebird 2.5	DB_Insert	162 ms	When executing an INSERT operation for 100-column record
	DB_Select	450 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
PostgreSQL 9.4	DB_Insert	277 ms	When executing an INSERT operation for 100-column record
	DB_Select	379 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}

*1. The primary key is specified for the retrieval condition.

- NX102-□□20

DB type	Instruction	Reference value for instruction execution time	Measurement condition
Oracle Database 11g	DB_Insert	227 ms	When executing an INSERT operation for 100-column record
	DB_Select	195 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
SQL Server 2012	DB_Insert	218 ms	When executing an INSERT operation for 100-column record
	DB_Select	163 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}

DB type	Instruction	Reference value for instruction execution time	Measurement condition
DB2 10.5	DB_Insert	428 ms	When executing an INSERT operation for 100-column record
	DB_Select	457 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
MySQL 5.6 Storage engine: InnoDB	DB_Insert	245 ms	When executing an INSERT operation for 100-column record
	DB_Select	220 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
Firebird 2.5	DB_Insert	202 ms	When executing an INSERT operation for 100-column record
	DB_Select	318 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
PostgreSQL 9.4	DB_Insert	287 ms	When executing an INSERT operation for 100-column record
	DB_Select	265 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}

*1. The primary key is specified for the retrieval condition.

- NJ501-□□20

DB type	Instruction	Reference value for instruction execution time	Measurement condition
Oracle Database 11g	DB_Insert	124 ms	When executing an INSERT operation for 100-column record
	DB_Select	175 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
SQL Server 2012	DB_Insert	136 ms	When executing an INSERT operation for 100-column record
	DB_Select	130 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
DB2 10.5	DB_Insert	315 ms	When executing an INSERT operation for 100-column record
	DB_Select	839 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
MySQL 5.6 Storage engine: InnoDB	DB_Insert	62 ms	When executing an INSERT operation for 100-column record
	DB_Select	38 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
Firebird 2.5	DB_Insert	35 ms	When executing an INSERT operation for 100-column record
	DB_Select	175 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}

DB type	Instruction	Reference value for instruction execution time	Measurement condition
PostgreSQL 9.4	DB_Insert	87 ms	When executing an INSERT operation for 100-column record
	DB_Select	111 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}

*1. The primary key is specified for the retrieval condition.

- NJ101-□□20

DB type	Instruction	Reference value for instruction execution time	Measurement condition
Oracle Database 11g	DB_Insert	219 ms	When executing an INSERT operation for 100-column record
	DB_Select	406 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
SQL Server 2012	DB_Insert	213 ms	When executing an INSERT operation for 100-column record
	DB_Select	248 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
DB2 10.5	DB_Insert	373 ms	When executing an INSERT operation for 100-column record
	DB_Select	395 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
MySQL 5.6 Storage engine: InnoDB	DB_Insert	219 ms	When executing an INSERT operation for 100-column record
	DB_Select	245 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
Firebird 2.5	DB_Insert	162 ms	When executing an INSERT operation for 100-column record
	DB_Select	450 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}
PostgreSQL 9.4	DB_Insert	277 ms	When executing an INSERT operation for 100-column record
	DB_Select	379 ms	When searching for one record from 100,000 records and retrieving 100-column data ^{*1}

*1. The primary key is specified for the retrieval condition.

● Countermeasures

Measure the execution time of each DB Connection Instruction in reference to *A-1-3 How to Measure Execution Time of DB Connection Instructions* on page A-11. If the execution time of a DB Connection Instruction exceeds the acceptable range of the equipment, take the following actions.

- 1 Set an instruction execution timeout for the DB Connection Instruction. Refer to *A-2-4 Ensuring Equipment Performance (Takt Time) by Monitoring Instruction Execution Timeout* on page A-23 for details.

- 2** Execute a dummy DB_Insert instruction once after executing the DB_CreateMapping instruction as a preparation for starting the actual operation.

Phenomenon 2: After Execution of a DB_CreateMapping Instruction, Execution Time of the First DB_Insert Instruction Got Longer

● Possible causes

The following can be the causes:

- For the first DB_Insert instruction that is executed after execution of a DB_CreateMapping instruction, the DB may require longer processing time than usual.

The following table gives the reference values for execution time of the first DB_Insert instruction that is executed after execution of a DB_CreateMapping instruction. The percentage of task execution time is 50%.

- NX701-□□20

DB type	Instruction	Reference value for instruction execution time	Measurement condition
Oracle Database 11g	DB_Insert	3.32 ms	When executing an INSERT operation for 100-column record
SQL Server 2012	DB_Insert	6.04 ms	When executing an INSERT operation for 100-column record
DB2 10.5	DB_Insert	86.08 ms	When executing an INSERT operation for 100-column record
MySQL 5.6 Storage engine: InnoDB	DB_Insert	21.13 ms	When executing an INSERT operation for 100-column record
Firebird 2.5	DB_Insert	5.31 ms	When executing an INSERT operation for 100-column record
PostgreSQL 9.4	DB_Insert	8.69 ms	When executing an INSERT operation for 100-column record

- NX102-□□20

DB type	Instruction	Reference value for instruction execution time	Measurement condition
Oracle Database 11g	DB_Insert	18.6 ms	When executing an INSERT operation for 100-column record
SQL Server 2012	DB_Insert	18.32 ms	When executing an INSERT operation for 100-column record
DB2 10.5	DB_Insert	24.37 ms	When executing an INSERT operation for 100-column record
MySQL 5.6 Storage engine: InnoDB	DB_Insert	35.77 ms	When executing an INSERT operation for 100-column record
Firebird 2.5	DB_Insert	27.66 ms	When executing an INSERT operation for 100-column record

DB type	Instruction	Reference value for instruction execution time	Measurement condition
PostgreSQL 9.4	DB_Insert	22.22 ms	When executing an INSERT operation for 100-column record

- NJ501-□□20

DB type	Instruction	Reference value for instruction execution time	Measurement condition
Oracle Database 11g	DB_Insert	29.9 ms	When executing an INSERT operation for 100-column record
SQL Server 2012	DB_Insert	17.5 ms	When executing an INSERT operation for 100-column record
DB2 10.5	DB_Insert	26.4 ms	When executing an INSERT operation for 100-column record
MySQL 5.6 Storage engine: InnoDB	DB_Insert	41.7 ms	When executing an INSERT operation for 100-column record
Firebird 2.5	DB_Insert	22.5 ms	When executing an INSERT operation for 100-column record
PostgreSQL 9.4	DB_Insert	14.1 ms	When executing an INSERT operation for 100-column record

- NJ101-□□20

DB type	Instruction	Reference value for instruction execution time	Measurement condition
Oracle Database 11g	DB_Insert	28.2 ms	When executing an INSERT operation for 100-column record
SQL Server 2012	DB_Insert	35.6 ms	When executing an INSERT operation for 100-column record
DB2 10.5	DB_Insert	52.7 ms	When executing an INSERT operation for 100-column record
MySQL 5.6 Storage engine: InnoDB	DB_Insert	59.3 ms	When executing an INSERT operation for 100-column record
Firebird 2.5	DB_Insert	32.6 ms	When executing an INSERT operation for 100-column record
PostgreSQL 9.4	DB_Insert	32.1 ms	When executing an INSERT operation for 100-column record

● Countermeasures

- 1 Measure the execution time of each DB Connection Instruction in reference to *A-1-3 How to Measure Execution Time of DB Connection Instructions* on page A-11. If the execution time of a DB Connection Instruction exceeds the acceptable range of the equipment, take the following actions.
 - Execute a dummy DB_Insert instruction once after executing the DB_CreateMapping instruction as a preparation for starting the actual operation.

Phenomenon 3: When Communications or SD Memory Card Processing was Executed in the CPU Unit, Execution Time of a DB Connection Instruction Got Longer

● Possible causes

The following can be the causes:

- The sufficient processing time may not be allocated to the DB Connection Service that is executed as a system service due to execution of communications or SD Memory Card processing.

● Countermeasures

- 1 Reconsider the task design so that the sufficient execution time can be allocated to the system services in reference to *A-1-4 Guideline for System Service Execution Time Ratio* on page A-11.

Phenomenon 4: Execution Time of DB Connection Instructions is Steadily Long

● Possible causes

The following can be the causes:

- The sufficient execution time may not be allocated to the system services.

● Countermeasures

- 1 Reconsider the task design so that the sufficient execution time can be allocated to the system services in reference to *A-1-4 Guideline for System Service Execution Time Ratio* on page A-11.

Phenomenon 5: Depending on the DB's Status, Execution Time of a DB Connection Instruction (i.e., DB_Insert, DB_Update, DB_Select, or DB_Delete Instruction Got Longer.

● Possible causes

The following can be the causes:

- Load on the server was temporarily increased.
- The specified table contains many records.
- The data clear operation was executed for the specified table.
- The specified table was temporarily locked.

● Countermeasures

- 1 Measure the processing time in the DB in reference to *A-2-3 How to Measure DB Response Time* on page A-23.

- 2** Identify the cause based on the timing when the processing time got longer in the DB and take a countermeasure in the server.

A-2-2 Impact of Operation Log Recording on Execution Time of DB Connection Instructions

When the Operation Logs are recorded, execution time of DB Connection Instructions (i.e. DB_Insert, DB_Update, DB_Select, and DB_Delete instructions) gets longer.

The following table gives the reference values for increased execution time of DB Connection Instructions while the Operation Logs are recorded. The percentage of task execution time is 50%.

Confirm that the equipment will not be adversely affected before starting recording to the Operation Logs.

- NX701-□□20

Log type	Instruction	Reference value for increase in instruction execution time	Measurement condition
Execution Log	DB_Insert	+0.1 ms	When executing an INSERT operation for 100-column record
Debug Log	DB_Insert	+0.5 ms	When executing an INSERT operation for 100-column record

- NX102-□□20

Log type	Instruction	Reference value for increase in instruction execution time	Measurement condition
Execution Log	DB_Insert	+1.0 ms	When executing an INSERT operation for 100-column record
Debug Log	DB_Insert	+5.7 ms	When executing an INSERT operation for 100-column record

- NJ501-□□20

Log type	Instruction	Reference value for increase in instruction execution time	Measurement condition
Execution Log	DB_Insert	+1.4 ms	When executing an INSERT operation for 100-column record
Debug Log	DB_Insert	+3.3 ms	When executing an INSERT operation for 100-column record

- NJ101-□□20

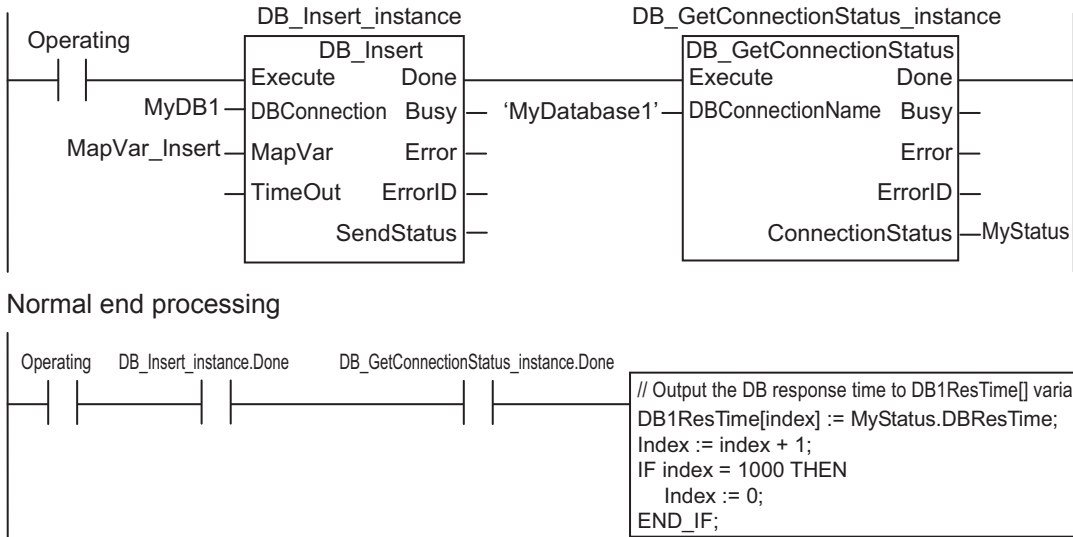
Log type	Instruction	Reference value for increase in instruction execution time	Measurement condition
Execution Log	DB_Insert	+2.0 ms	When executing an INSERT operation for 100-column record
Debug Log	DB_Insert	+7.6 ms	When executing an INSERT operation for 100-column record

A-2-3 How to Measure DB Response Time

The DB response time refers to the time since an SQL statement is sent from the CPU Unit until the SQL execution result is returned from the DB. You can find the DB response time by executing a DB_GetConnectionStatus instruction after executing an instruction that sends an SQL statement.

An example user program is given below.

- Measurement example of DB response time for a DB_Insert instruction
Find the DB response time for a DB_Insert instruction.



You can also check the DB response time with the Execution Log or Debug Log.

A-2-4 Ensuring Equipment Performance (Takt Time) by Monitoring Instruction Execution Timeout

If you do not want to lower the equipment performance (or extend the takt time) when the execution time of DB Connection Instruction is increased, set an instruction execution timeout for the instructions.

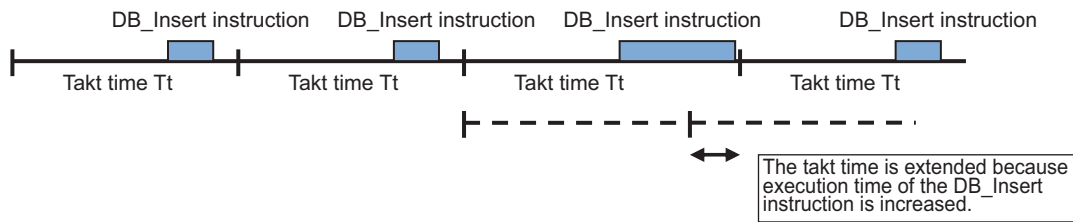
You can specify an instruction execution timeout in the Timeout input variable to the DB_Insert, DB_Update, DB_Select, and DB_Delete instructions.

For the instruction execution timeout of instructions, specify the maximum time that can be used for DB access in the takt time.

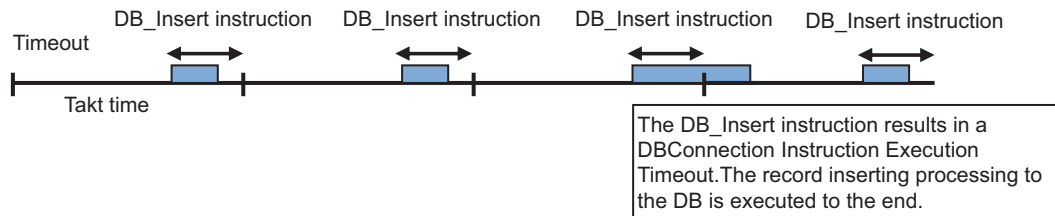
If you set an instruction execution timeout for a DB_Insert instruction for the equipment where production data is stored into the DB using the DB_Insert instruction at the end of the takt time, for example, a DB Connection Instruction Execution Timeout will occur for the DB_Insert instruction when the record inserting processing to the DB is not completed in the takt time. In this case, the record inserting processing to the DB is executed to the end.

You can continue the operation without lowering the equipment performance (or extending the takt time) by specifying an instruction execution timeout for the instruction even if execution time of DB Connection Instructions is temporarily increased.

- When instruction execution timeout not specified



- When instruction execution timeout specified



Precautions for Correct Use

- When a DB Connection Instruction Execution Timeout occurred for a DB_Select instruction, the values of the retrieved record are not stored in the MapVar in-out variable.
- When a DB Connection Instruction Execution Timeout occurs repeatedly, reconsider the task design and the server environment that contains the DB.

A-3 Specifications

This section gives the specifications of the Database Connection CPU Units.

A-3-1 General Specifications

Refer to the following manual.

- *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)*

A-3-2 Performance Specifications

Refer to the following manual.

- *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)*

A-3-3 Function Specifications

Refer to the following manual.

- *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)*
- Common Specifications to NJ/NX-series CPU Units

Item			NX701-□ □20	NX102-□ □20	NJ501-1□20		NJ501-4320	NJ101-□ □20
					Version 1.07 or earlier	Version 1.08 or later		
De- bug- ging	Data tracing	Maximum number of si- multaneous data traces	4	2 ^{*1}	4	2 ^{*1}	2 ^{*1}	2 ^{*1}

*1. If the trace number is set to 2 or greater when executing a data trace related instruction, an error (Illegal Data Position Specified) will occur for the instruction. ENO of the instruction will become FALSE.

- DB Connection Service Functionality

Refer to *1-2-1 DB Connection Service Specifications* on page 1-5 for detailed specifications of the DB Connection Service.

A-4 Version Information

This section describes the relationship between the unit versions of CPU Units and the Sysmac Studio versions, and the DB Connection functions that were added or changed for each unit version of the CPU Units.

A-4-1 Unit Versions and Corresponding DB Connection Service Versions

The following table gives the relationship between unit versions of CPU Units and the DB Connection Service versions.

● NX701-□□20

Unit version of CPU Unit	DB Connection Service version
1.21 or later	2.00
1.16 or later	1.03

● NX102-□□20

Unit version of CPU Unit	DB Connection Service version
1.33 or later	2.00
1.30 or later	1.04

● NJ501-□□20 or NJ101-□□20

Unit version of CPU Unit	DB Connection Service version
1.10 or later*1	1.02
1.10*1	1.01
1.09	
1.08	
1.07 or earlier	1.00

*1. The CPU Units with unit version 1.10 come with DB Connection Service version 1.01 or 1.02. The version can be checked with the Production Information Dialog Box of Sysmac Studio while online. Refer to *Versions* on page 21 for how to check the versions of the CPU Units and DB Connection Service.

A-4-2 DB Connection Functions that were Added or Changed for Each Unit Version

This section gives the DB Connection functions that were added or changed for version upgrades of CPU Units.

Additions and Changes to Function Specifications

The following table gives the unit version of the CPU Units and the Sysmac Studio version for each addition or change to the function specifications.

Refer to the following manual for other function specifications.

- *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)*

● NX701-□□20

Function	Addition/change	Unit version	Sysmac Studio version	Reference
Stored procedure	Addition	1.21	1.29 or higher	5-3 <i>Stored Procedure Call Function</i> on page 5-16
Batch insert	Addition			5-4 <i>Batch Insert Function</i> on page 5-24

● NX-102-□□20

Function	Addition/change	Unit version	Sysmac Studio version	Reference
Stored procedure	Addition	1.33	1.29 or higher	5-3 <i>Stored Procedure Call Function</i> on page 5-16
Batch insert	Addition			5-4 <i>Batch Insert Function</i> on page 5-24

● NJ501-□□20 or NJ101-□□20

Function		Addition/change	Unit version	Sysmac Studio version	Reference
DB Connection settings	Database type	Change	1.08	1.09	2-2-2 <i>DB Connection Settings</i> on page 2-7
		Change	1.10	1.14	
DB Connection status	SQL status*1	Change	1.08	---	4-3-4 <i>Checking the Status of each DB Connection</i> on page 4-11
	Error code*1				
	Error message*1				

*1. Error information in the SQL Server connection was changed.

A-4-3 Unit Version, DB Connection Service Version, and Unit Version Set in the Sysmac Studio Project

The following table gives the relationship between the unit versions of CPU Units, the DB Connection Service versions, and the corresponding Sysmac Studio versions.

Unit Versions and Corresponding Sysmac Studio Versions

The following table gives the relationship between the unit versions of CPU Units, the DB Connection Service versions, and the Sysmac Studio versions that can set the unit versions for cases the DB Connection Service versions are modified. Refer to the *NJ/NX series CPU Unit Software User's Manual (Cat. No. W501)* for all the combinations of the unit versions of CPU Units and the Sysmac Studio versions that can set the unit versions.

● NX701-□□20

Unit version of CPU Unit	DB Connection Service version	Sysmac Studio version that can set the unit version
1.21 or later	2.00	1.29 or higher
1.16 or later	1.03	1.21 or higher

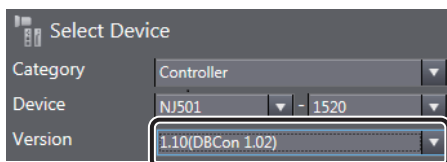
● NX102-□□20

Unit version of CPU Unit	DB Connection Service version	Sysmac Studio version that can set the unit version
1.33 or later	2.00	1.29 or higher
1.30 or later	1.04	1.24 or higher

● NJ501-□□20 or NJ101-□□20

Unit version of CPU Unit	DB Connection Service version	Sysmac Studio version that can set the unit version
1.10 or later	1.02	1.14 or higher *1*2
	1.01	1.13 or higher *1*2
1.08		1.09 or higher
1.05	1.00	1.06 or higher

- *1. When you set a unit version in Sysmac Studio version 1.14 or higher, a unit version and DB Connection Service version are displayed in the Version box because more than one DB connection version exists for the same unit version of the CPU Unit. For example, when you want to create a project for a CPU Unit with unit version 1.10 with the DB Connection Service version 1.02, select "1.10 (DBCon 1.02)" in the "Version" box.



- *2. Sysmac Studio version 1.14 or higher is required to use NJ101-□□20 Database Connection CPU Units. NJ101-□□20 cannot be used with Sysmac Studio version 1.13 or lower.

Relationship between DB Connection Service Version and Unit Version Set in the Sysmac Studio Project

The following table shows the difference in the specifications for a combination of the actual DB Connection Service version of the CPU Unit and the DB Connection Service version set in the Sysmac Studio project.

For any other cases than the one shown below, it is not necessary to consider the difference of specifications.

● Supported Database Type

- NJ501-1□□20 or NJ101-□□20

DB Connection Service version of the CPU Unit	DB Connection Service version set in the Sysmac Studio project		
	1.00*1	1.01	1.02 or higher
1.02 or higher	Oracle SQL Server	Oracle SQL Server DB2 MySQL Firebird	Oracle SQL Server DB2 MySQL Firebird PostgreSQL

DB Connection Service version of the CPU Unit	DB Connection Service version set in the Sysmac Studio project		
	1.00*1	1.01	1.02 or higher
1.01	Oracle SQL Server	Oracle SQL Server DB2 MySQL Firebird	Transfer is not possible.
1.00*1	Oracle SQL Server	Transfer is not possible.	Transfer is not possible.

*1. Only for NJ501-1□20

● Maximum Number of DB Map Variables For Which a Mapping Can Be Created

- NX701-1□20

DB Connection Service version of the CPU Unit	DB Connection Service version set in the Sysmac Studio project	
	1.03	2.00 or higher
2.00 or higher	SQL Server: 60	Same as left
1.03	Oracle: 30 DB2: 30 MySQL: 30 Firebird: 15 PostgreSQL: 30	Transfer is not possible.

- NX102-□□20

DB Connection Service version of the CPU Unit	DB Connection Service version set in the Sysmac Studio project	
	1.04	2.00 or higher
2.00 or higher	15	SQL Server: 30 Oracle: 20 DB2: 20 MySQL: 20 Firebird: 15 PostgreSQL: 20
1.04		Transfer is not possible.

- NJ501-1□20

DB Connection Service version of the CPU Unit	DB Connection Service version set in the Sysmac Studio project		
	1.00	1.01	1.02 or higher
1.02 or higher	15	SQL Server: 60 Oracle: 30 DB2: 30 MySQL: 30 Firebird: 15	SQL Server: 60 Oracle: 30 DB2: 30 MySQL: 30 Firebird: 15 PostgreSQL: 30
1.01			Transfer is not possible.
1.00		Transfer is not possible.	Transfer is not possible.

- NJ101-1□20

DB Connection Service version of the CPU Unit	DB Connection Service version set in the Sysmac Studio project	
	1.01	1.02 or higher
1.02 or higher	15	15
1.01		Transfer is not possible.

● Number of Database Connections

- NX102-□□20

DB Connection Service version of the CPU Unit	DB Connection Service version set in the Sysmac Studio project	
	1.04	2.00 or higher
2.00 or higher	1	2
1.04		Transfer is not possible.

A-4-4 DB Connection Service Version and Connection Database Types After Changing Devices

If you change the unit version of the CPU Unit by executing **Change Device** on Sysmac Studio, the version of the DB Connection Service may be downgraded. If the version of the DB Connection Service is downgraded, the types of the connected databases may change automatically.

The types of databases that can be connected after changing devices are determined as shown below according to the DB Connection Service version of the CPU Unit and the CPU Unit model after changing devices, as well as the types of connected databases before changing devices.

DB Connection Service version of the CPU Unit after changing devices	CPU Unit models after changing devices	Types of connected databases before changing devices	Types of connected databases after changing devices
1.02 or higher	NJ501-4□20	Oracle SQL Server MySQL	Same as the left
		Others	SQL Server
	Other than NJ501-4□20	All types	Same as the left
1.01	NJ501-4□20	Oracle SQL Server MySQL	Same as the left
		Others	SQL Server
	Other than NJ501-4□20	Oracle SQL Server DB2 MySQL Firebird	Same as the left
		Others	SQL Server
1.00	All models	Oracle SQL Server	Same as the left
		Others	SQL Server

A-4-5 DB Connection Service Versions and Connection Database Types/Versions

The following table specifies the relationship between the DB Connection Service versions and the connection database types/versions.

DB Connection Service version (DBcon)	SQL Server	Oracle Database	DB2 for Linux, UNIX and Windows	MySQL Community Edition	Firebird	PostgreSQL
1.00	2008, 2008R2, 2012	10g, 11g	---	---	---	---
1.01	2008, 2008R2, 2012	10g, 11g	9.5, 9.7, 10.1, 10.5	5.1, 5.5, 5.6	2.1, 2.5	---
1.02	2008, 2008R2, 2012, 2014	10g, 11g, 12c	9.5, 9.7, 10.1, 10.5	5.1, 5.5, 5.6	2.1, 2.5	9.2, 9.3, 9.4
1.03	2008, 2008R2, 2012, 2014, 2016	10g, 11g, 12c	9.5, 9.7, 10.1, 10.5, 11.1	5.1, 5.5, 5.6, 5.7	2.1, 2.5	9.2, 9.3, 9.4, 9.5, 9.6
1.04	2008, 2008R2, 2012, 2014, 2016, 2017	10g, 11g, 12c	9.5, 9.7, 10.1, 10.5, 11.1	5.1, 5.5, 5.6, 5.7	2.1, 2.5	9.2, 9.3, 9.4, 9.5, 9.6
2.0 or higher	2012, 2014, 2016, 2017	11g, 12c*1, 18c	9.7, 10.1, 10.5, 11.1	5.6, 5.7, 8.0	2.5	9.4, 9.5, 9.6, 10

*1. Includes the Release2.



Index

Index

A

- Adding a DB Connection..... 2-7
- Assumed cause..... 8-8 – 8-15

B

- Backup/Restore Function..... 5-33 – 5-36
- Batch insert..... 27

C

- CA..... 26
- Changing the DB Connection Name..... 2-8
- Checking the Status of each DB Connection..... 4-11
- Checking the Status of the DB Connection Service..... 4-8
- Clearing the Mapping of DB Map Variables..... 3-20
- Clearing the SQL Statements from the Spool Memory..... 5-9
- Column..... 26
- Communications Test..... 2-12
- Connected time..... 4-12
- Connection name..... 2-10
- Connection Settings..... 2-8
- Connection Status..... 4-12
- Correspondence of Data Types between NJ/NX-series Controllers and DB..... 3-4
- Creating a Structure Data Type..... 3-3, 3-13

D

- Data Already Spooled..... 8-43
- Database type..... 2-10
- DB..... 26
- DB Connection..... 26
- DB Connection Already Established..... 8-31
- DB Connection Disconnected Error..... 8-22
- DB Connection Disconnected Error Status..... 8-39
- DB Connection Failed..... 8-29
- DB Connection function..... 26
- DB Connection Instruction..... 26
- DB Connection Instruction Category..... 3-25
- DB Connection Instruction Execution Timeout..... 8-41
- DB Connection Instruction Set..... 3-25
- DB Connection Rejected..... 8-29
- DB Connection Service..... 26
- DB Connection Service Error Stop..... 8-41
- DB Connection Service Initializing..... 8-43
- DB Connection Service Not Started..... 8-25
- DB Connection Service Run Mode Change Failed..... 8-26
- DB Connection Service Settings..... 2-5
- DB Connection Service Shutdown..... 8-23
- DB Connection Service shutdown function..... 26, 5-26
- DB Connection Service Shutdown or Shutting Down..... 8-27
- DB Connection Service Started..... 8-23
- DB Connection Service Status..... 3-27
- DB Connection Service Stopped..... 8-23

- DB Connection Setting Error..... 8-20
- DB Connection Settings..... 2-5, 2-7
- DB Connection System..... 1-11
- DB in Process..... 8-45
- DB Map Variable..... 26, 3-17
- DB mapping..... 26, 3-2, 3-17
- DB Records Batch Insert instruction..... 26
- DB Server Certificate Error..... 8-21
- DB_AttachProcedure (Generate DB Stored Procedure Handle) instruction..... 3-26, 7-107
- DB_BatchInsert (DB Records Batch Insert) instruction..... 3-26, 7-94
- DB_Close (Close DB Connection)..... 3-25, 7-10
- DB_Connect (Establish DB Connection) instruction..... 3-25, 7-6
- DB_ControlService (Control DB Connection Service) instruction..... 3-26, 4-4, 7-60
- DB_ControlSpool (Resend/Clear Spool Data) instruction..... 3-25, 3-26, 5-9, 5-10, 7-13, 7-77
- DB_CreateMapping..... 3-20
- DB_CreateMapping (Create DB Map) instruction..... 3-2, 3-17
- DB_Delete (Delete DB Record) instruction..... 3-26, 7-45
- DB_DetachProcedure (Release DB Stored Procedure Handle) instruction..... 3-26, 7-124
- DB_ExecuteProcedure (Execute DB Stored Procedure) instruction..... 3-26, 7-112
- DB_GetConnectionStatus (Get DB Connection Status) instruction..... 3-26, 4-14, 5-5, 7-71
- DB_GetServiceStatus (Get DB Connection Service Status) instruction..... 3-26, 7-66
- DB_Insert (Insert DB Record) instruction..... 3-26, 7-17
- DB_PutLog (Record Operation Log) instruction..... 3-26, 7-84
- DB_Select (Retrieve DB Record) instruction..... 3-26, 7-39
- DB_Shutdown (Shutdown DB Connection Service) instruction..... 3-26, 7-90
- DB_Update (Update DB Record) instruction..... 3-26, 5-5, 7-21
- Debug Log..... 27, 2-6
- Dedicated area for the Spool function..... 5-6
- Disconnected time..... 4-12
- Disconnection date/time..... 4-12

E

- EM Area..... 5-7
- Encrypted communication..... 26
- Error code..... 4-13
- Error message..... 4-13
- Errors..... 8-5, 8-7
- Establishing/Closing a DB Connection..... 4-6
- Event code..... 8-8 – 8-15
- Event name..... 8-8 – 8-15
- Execution Log..... 26, 2-6
- Execution Log Save Failed..... 8-18

H

How to Prevent Losing SQL Statements at Power Interruption..... 5-28

I

Instruction Executed for Unsupported Database Type.... 8-47
 Invalid DB Connection..... 8-32
 Invalid DB Connection Name..... 8-27
 Invalid DB Map Variable..... 8-32
 Invalid Extraction Condition..... 8-38
 Invalid Number of Columns for Stored Procedure Result Set..... 8-50
 Invalid Procedure Handle..... 8-46
 Invalid Stored Procedure Argument..... 8-49
 Invalid Stored Procedure Execution..... 8-51
 Invalid Stored Procedure Name..... 8-48
 IP address..... 2-10

L

Log Code Out of Range..... 8-39
 Login timeout..... 2-12

M

Manual Resend..... 5-9
 Microsoft Excel..... 3-14

N

Number of error executions..... 4-13
 Number of normal executions..... 4-13
 Number of spool data..... 4-13

O

Operating time..... 4-9
 Operation Authority Verification..... 5-37
 Operation Log..... 26
 Operation Log Disabled..... 8-45
 Operation Mode..... 2-6, 4-2
 Operation status..... 4-9
 Oracle Data base..... 1-2

P

Possible causes..... 8-8 – 8-15
 Programming the DB Connection Service..... 3-24

Q

Query Execution..... 4-13
 Query execution timeout..... 2-12

R

Record processing..... 26
 Registration and Attributes of DB Map Variables..... 3-18
 Resend Spool Data..... 5-8

Response time..... 4-13
 Restrictions on Column Names..... 3-4
 Restrictions on DB Map Variables..... 3-19
 Restrictions on DB Mapping..... 3-20
 restrictions on structure member names in the NJ/NX-series
 Controllers..... 3-4
 Restrictions on Table's Column Names..... 3-4
 Run mode of the DB Connection Service..... 26, 4-2

S

Server Certificate..... 26
 Service Start..... 2-6
 Spool Capacity Exceeded..... 8-37
 Spool data..... 27
 Spool function..... 27
 Spool Function..... 5-5
 Spool Function Settings..... 5-8
 Spool memory..... 27
 Spool Memory Cleared..... 8-24
 Spool Memory Corrupted..... 8-17
 Spool Settings..... 2-13
 Spool usage..... 4-13
 SQL..... 26
 SQL Execution Error..... 8-35
 SQL Execution Failure Log..... 26, 2-7
 SQL Execution Failure Log Save Failed..... 8-19
 SQL Server..... 1-2
 SQL status..... 4-13
 SQL type..... 26, 3-20
 SQL Type..... 3-17
 Stored function..... 27
 Stored procedure..... 27
 Stored procedure call..... 27
 Structure data type for DB access..... 26
 Structure member name..... 3-3
 Structure member's data type..... 3-3
 Structure name..... 3-3
 System-defined Variables..... 3-26, 4-10

T

Table..... 27
 Test Mode..... 2-6, 4-2
 Too Many DB Connections..... 8-31
 Troubleshooting..... 8-8

U

Unregistered DB Map Variable..... 8-34

V

Variables Used in the DB Connection Instructions..... 7-2

OMRON AUTOMATION AMERICAS HEADQUARTERS • Chicago, IL USA • 847.843.7900 • 800.556.6766 • automation.omron.com

OMRON CANADA, INC. • HEAD OFFICE

Toronto, ON, Canada • 416.286.6465 • 866.986.6766 • automation.omron.com

OMRON ELECTRONICS DE MEXICO • HEAD OFFICE

Ciudad de México • 52.55.5901.4300 • 01.800.386.6766 • mela@omron.com

OMRON ELECTRONICS DE MEXICO • SALES OFFICE

San Pedro Garza García, N.L. • 81.12.53.7392 • 01.800.386.6766 • mela@omron.com

OMRON ELECTRONICS DE MEXICO • SALES OFFICE

Eugenio Garza Sada, León, Gto • 01.800.386.6766 • mela@omron.com

OMRON ELETRÔNICA DO BRASIL LTDA • HEAD OFFICE

São Paulo, SP, Brasil • 55 11 5171-8920 • automation.omron.com

OMRON ARGENTINA • SALES OFFICE

Buenos Aires, Argentina • +54.11.4521.8630 • +54.11.4523.8483
mela@omron.com

OTHER OMRON LATIN AMERICA SALES

+54.11.4521.8630 • +54.11.4523.8483 • mela@omron.com

Authorized Distributor:

Controllers & I/O

- Machine Automation Controllers (MAC) • Motion Controllers
- Programmable Logic Controllers (PLC) • Temperature Controllers • Remote I/O

Robotics

- Industrial Robots • Mobile Robots

Operator Interfaces

- Human Machine Interface (HMI)

Motion & Drives

- Machine Automation Controllers (MAC) • Motion Controllers • Servo Systems
- Frequency Inverters

Vision, Measurement & Identification

- Vision Sensors & Systems • Measurement Sensors • Auto Identification Systems

Sensing

- Photoelectric Sensors • Fiber-Optic Sensors • Proximity Sensors
- Rotary Encoders • Ultrasonic Sensors

Safety

- Safety Light Curtains • Safety Laser Scanners • Programmable Safety Systems
- Safety Mats and Edges • Safety Door Switches • Emergency Stop Devices
- Safety Switches & Operator Controls • Safety Monitoring/Force-guided Relays

Control Components

- Power Supplies • Timers • Counters • Programmable Relays
- Digital Panel Meters • Monitoring Products

Switches & Relays

- Limit Switches • Pushbutton Switches • Electromechanical Relays
- Solid State Relays

Software

- Programming & Configuration • Runtime