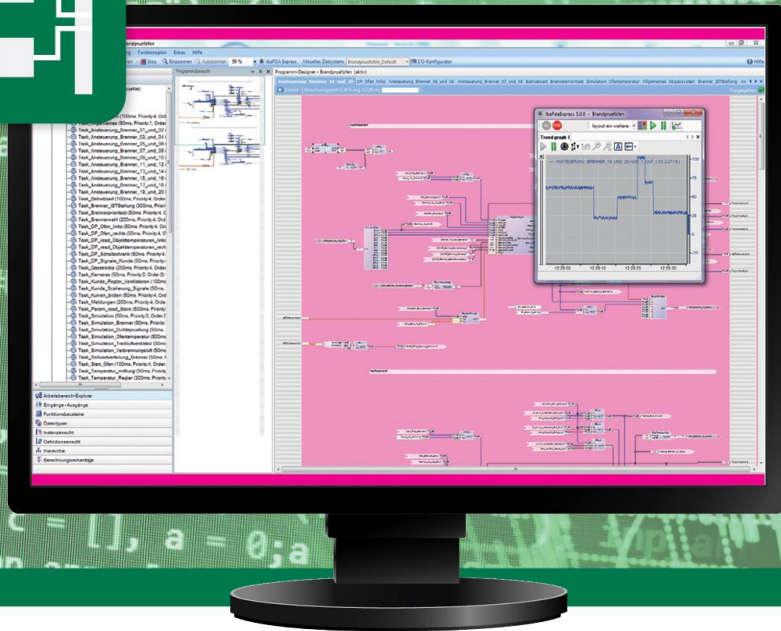
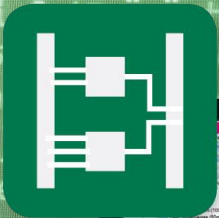




See the Big Picture



ODBC_ACCESS

Function block in ibaLogic

Manual

Issue 1.0

**Measurement Systems
for Industry and Energy**

Manufacturer

iba AG
Koenigswarterstr. 44
90762 Fuerth
Germany

Contacts

Main office	+49 911 97282-0
Fax	+49 911 97282-33
Support	+49 911 97282-14
Engineering	+49 911 97282-13
E-Mail	iba@iba-ag.com
Web	www.iba-ag.com

This manual must not be circulated or copied, or its contents utilized and disseminated, without our express written permission. Any breach or infringement of this provision will result in liability for damages.

© iba AG 2018, All Rights Reserved

The content of this publication has been checked for compliance with the described hardware and software. Nevertheless, deviations cannot be excluded completely so that the full compliance is not guaranteed. However, the information in this publication is updated regularly. Required corrections are contained in the following issues or can be downloaded from the Internet.

The current version is available for download on our web site <http://www.iba-ag.com>.

Issue	Date	Revision	Author	Version SW
1.0	18.10.2018	First issue	ST	5.4.0

Windows® is a label and registered trademark of the Microsoft Corporation. Other product and company names mentioned in this manual can be labels or registered trademarks of the corresponding owners.

Table of Contents

1	About this manual	2
1.1	Target group	2
1.2	Notations	2
1.3	Used symbols	3
2	General information	4
3	Configuration for standard SQL commands	5
3.1	Creating a database (MS SQL)	5
3.2	Creating an ODBC connection to MS SQL	6
3.3	Configuration of the ODBC_ACCESS block	9
3.4	Example for StoredProcedure in MSSQ	15
4	Configuration for calling StoredProcedure in Oracle	17
4.1	ODBC connection to Oracle	19
5	Appendix	22
5.1	ODBC Data types (MSSQL/ORACLE)	22
6	Support and contact	25

1 About this manual

This documentation describes the function and application of the function block *ODBC_ACCESS* in *ibaLogic*.

This document is a supplement to the general *ibaLogic* manual, which describes the general functions and operating options of *ibaLogic*.

1.1 Target group

This manual addresses in particular the qualified professionals who are familiar with handling electrical and electronic modules as well as communication and measurement technology. A person is regarded as professional if he/she is capable of assessing safety and recognizing possible consequences and risks on the basis of his/her specialist training, knowledge and experience and knowledge of the standard regulations.

1.2 Notations

In this manual the following notations are used:

Action	Notation
Menu command	Menu "Logic diagram"
Calling the menu command	"Step 1 – Step 2 – Step 3 – Step x" Example: Select the menu "Logic diagram - Add - New function block".
Keys	<Key name> Example: <Alt>; <F1>
Press the keys simultaneously	<Key name> + <Key name> Example: <Alt> + <Ctrl>
Buttons	<Key name> Example: <OK>; <Cancel>
File names, paths	"Filename", "Path" Example: "Test.doc"

1.3 Used symbols

If safety instructions or other notes are used in this manual, they mean:

DANGER

The non-observance of this safety information may result in an imminent risk of death or severe injury:

- ☐ From an electric shock!
 - ☐ Due to the improper handling of software products which are coupled to input and output procedures with control function!
-

WARNING

The non-observance of this safety information may result in a potential risk of death or severe injury!

CAUTION

The non-observance of this safety information may result in a potential risk of injury or material damage!



Note

A note specifies special requirements or actions to be observed.



Important note

Note if some special features must be observed, for example exceptions from the rule.



Tip

Tip or example as a helpful note or insider tip to make the work a little bit easier.



Other documentation

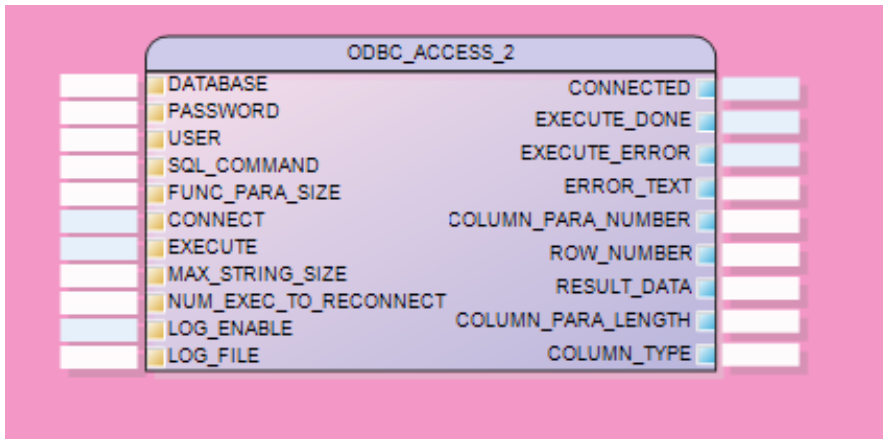
Reference to additional documentation or further reading.



Example

Configuration and application examples for a better understanding

2 General information



Using the ODBC_ACCESS block, write and read access to databases can be implemented. For this purpose, an ODBC connection to the respective database must be possible and set up.

The ODBC_ACCESS block is only executable on an ibaLogic system running on a PC. ibaPADU-S-IT-2x16 cannot use the device due to system restrictions.

Standard SQL commands are allowed. Stored Procedure calls are also possible.

The ODBC_ACCESS block is licensed and must be activated in the dongle.

Order no.	Name	Description
32.500030	ibaLogic-V5-DB ACCESS	License for ibaLogic-V5 function block for DB Access

In order to use the function block, place the SQL command on the input of the block. The result output can be placed directly on the required result structure. If several different result structures are to be expected, this can be realized using appropriate multiplexers.

Different configurations are described in the following sections:

- ☐ Configuration for standard SQL commands like SELECT/INSERT/DELETE etc.
- ☐ Configuration for calling StoredProcdure like CALL...

3 Configuration for standard SQL commands

Required steps for database access via ODBC:

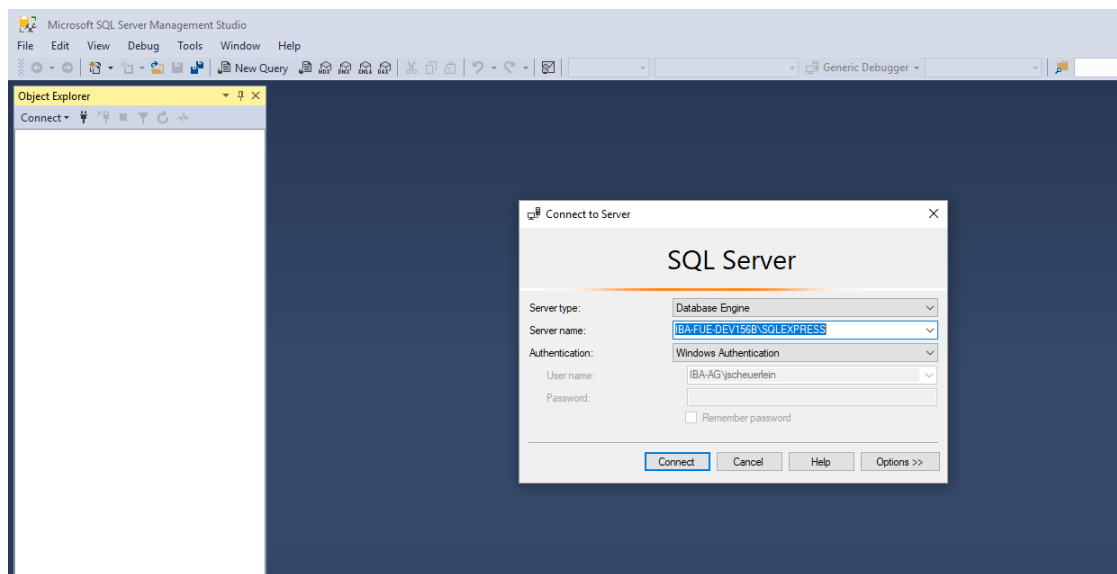
1. Create the database and table(s) for access, if they do not exist yet
2. Establish an ODBC connection to this database
3. Configure the ODBC device in ibaLogic with
 - ODBC connection name
 - SQL command
 - Logic for executing the SQL command
 - Decoding of the result into corresponding result structures

3.1 Creating a database (MS SQL)

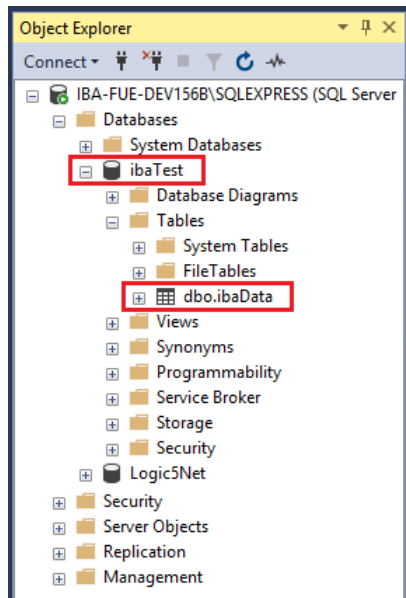
If a special database is not available, you can also use the installed ibaLogic database. This is usually an MS SQL Express database.

A download from Microsoft SQL Server Management Studio is required to access the database and its tables.

Start SQL Server Management Studio and log in with your Windows authentication.



Create a database and a new table (here: ibaTest and ibaData). The prefix dbo. is generated by the system and is not part of the table name if it is addressed from outside later.



Note: Logic5Net is the internal DB of ibaLogic.

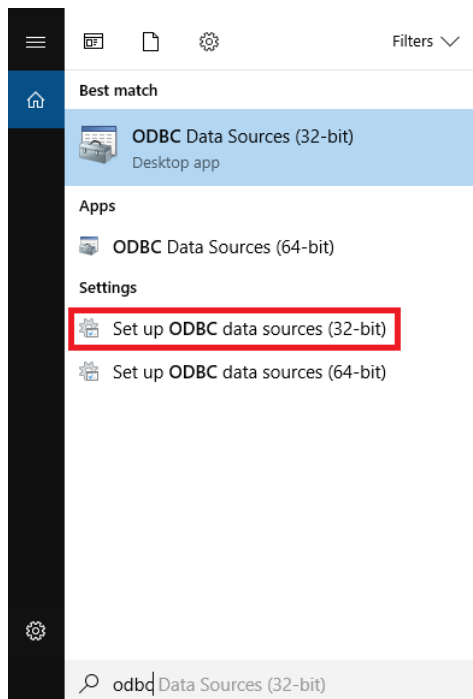
The SQL Server Management Studio can also be used to create test data or to check changes made by the ODBC_ACCESS block from ibaLogic.

3.2 Creating an ODBC connection to MS SQL

The ODBC connection to an MS SQL database is described here as an example.

Note: There are ODBC settings for 32/64 bit systems.

In the example, the ODBC settings for 32-bit are required, since this is a 32-bit application (even if it is a 64-bit system).



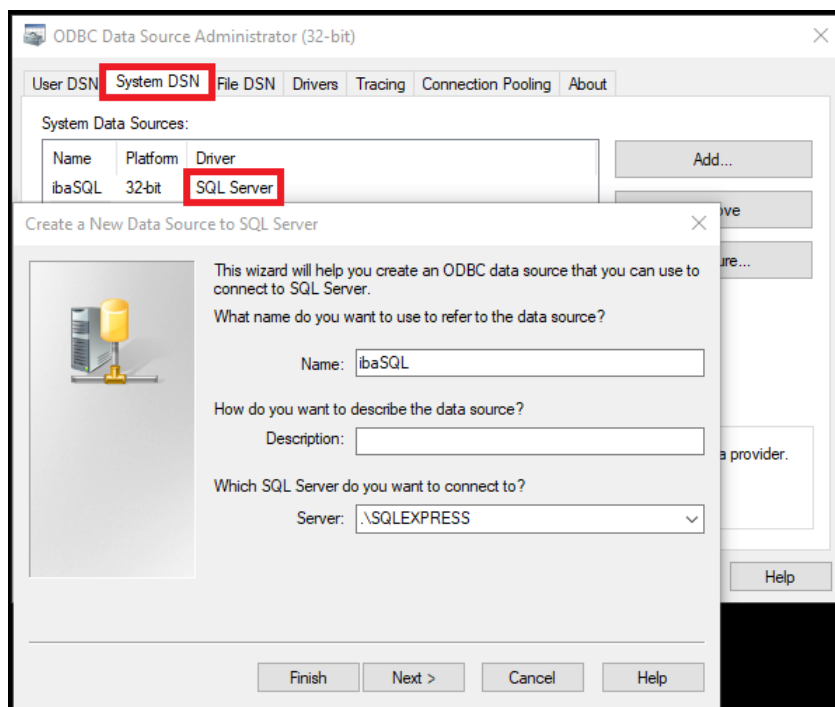
**Note**

If you can't find the appropriate ODBC settings via the search in the Windows system, you can also start them directly `C:\Windows\SysWOW64\odbcad32.exe`.

Further information can be found here:

<https://support.microsoft.com/de-de/help/942976/odbc-administrator-tool-displays-both-the-32-bit-and-the-64-bit-user-d>

1. Enter a new connection under System DSN and select the appropriate ODBC driver (here: SQL Server).
2. Give the ODBC connection a freely selectable name (here: ibaSQL)
3. Enter the name of the SQL server. You can find it in the SQL Server Management Studio.



Click <Next>.

4. Check these settings and click <Next>.

Microsoft SQL Server DSN Configuration

How should SQL Server verify the authenticity of the login ID?

☒ With Windows NT authentication using the network login ID.

☐ With SQL Server authentication using a login ID and password entered by the user.

To change the network library used to communicate with SQL Server, click Client Configuration.

Client Configuration...

☒ Connect to SQL Server to obtain default settings for the additional configuration options.

Login ID: jscheuerlein

Password:

< Back Next > Cancel Help

5. Enter the table name and click <Next>.

Microsoft SQL Server DSN Configuration

☒ Change the default database to: ibaTest

☐ Attach database filename:

☒ Use ANSI quoted identifiers.

☒ Use ANSI nulls, paddings and warnings.

☐ Use the failover SQL Server if the primary SQL Server is not available.

< Back Next > Cancel Help

6. Check these settings:

Microsoft SQL Server DSN Configuration

☐ Change the language of SQL Server system messages to: English

☐ Use strong encryption for data

☒ Perform translation for character data

☐ Use regional settings when outputting currency, numbers, dates and times.

☐ Save long running queries to the log file:

C:\Users\JSCHU~1\AppData\Local\Temp\QUEF Browse...

Long query time (milliseconds): 30000

☐ Log ODBC driver statistics to the log file:

C:\Users\JSCHU~1\AppData\Local\Temp\STAT: Browse...

< Back Finish Cancel Help

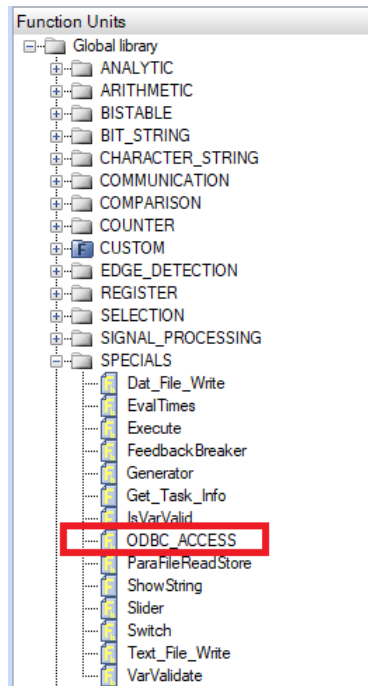
7. Complete with <Finish>.

The ODBC connection is now established.

3.3 Configuration of the ODBC_ACCESS block

The example ODBC_SQL_example.il5 is available on the DVD "Iba Software & Manuals".

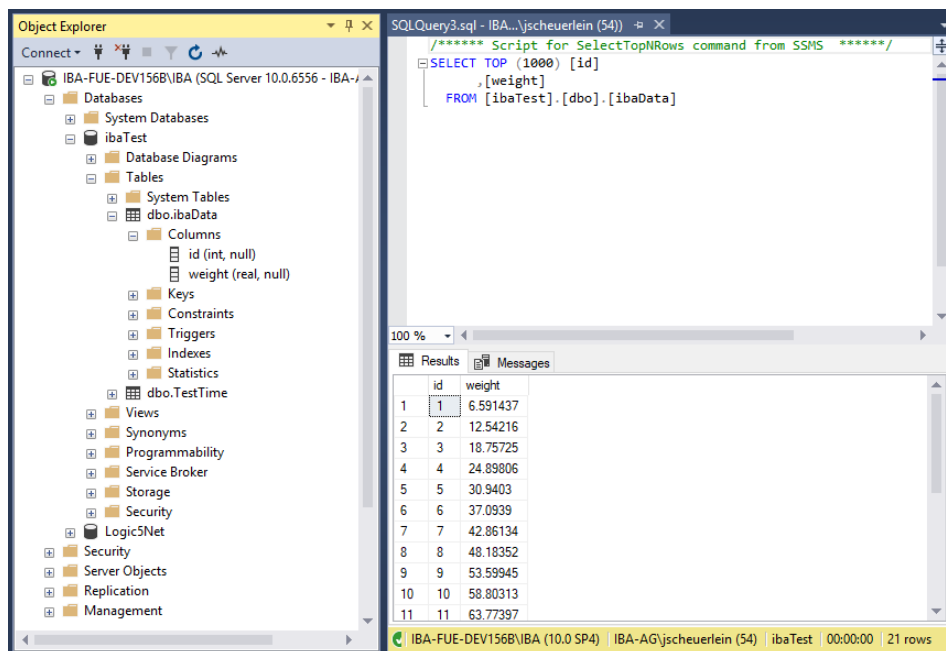
The ODBC_ACCESS block can be found here:



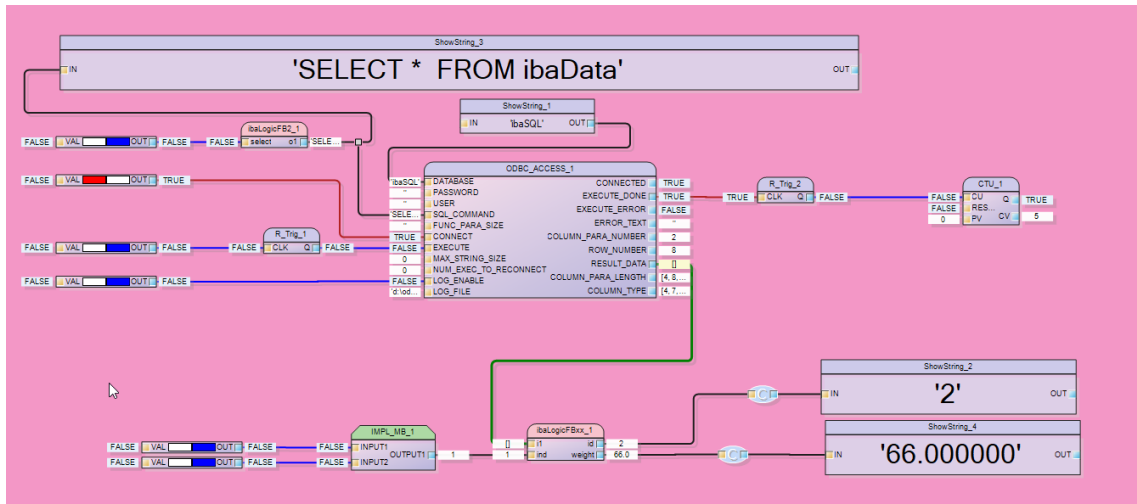
Drag the block to the layout using drag & drop.

The configuration is shown as example using a select command. The following example shows the contents of a database table to be retrieved.

The table contains ID (Int) and WEIGHT (Real) and corresponding values.



Query via ODBC_ACCESS:



Inputs

Connector	Data type	Explanation
DATABASE	String	ODBC data source name
PASSWORD / USER	String	Logon data for access-protected database
SQL_COMMAND	String	Standard SQL command (here: SELECT * from ibaData)
FUNC_PARA_SIZE	String	Data size of the parameters according to the data structure (only with StoredProcedure call), e.g.: 'INT,REAL,STRING10,REAL'
CONNECT	Bool	TRUE: Connection to the database is established
EXECUTE	Bool	Pulse controlled input for executing SQL command
MAX_STRING_SIZE	Dint	0 : 1024 bytes as maximum string length. To reduce the memory requirement for the result, you can limit the maximum string length. This must be considered in the result by a corresponding string derived type definition with this length.
NUM_EXEC_TO_DISCONNECT	Dint	Number of Execute commands after which an automatic reconnect to the database is carried out. This may be necessary if databases do not remove temporary data. 0: 50 Execute command (fixed default value)
LOG_ENABLE	Bool	Enable to write a log file.
LOG_FILE	String	The log file name consists of path and file name, e.g. 'D:\ODBC_ACCESS.log' All Execute commands and error messages are written to a text file with date/time. (No automatic clean-up mechanism)

Outputs

Connector	Data type	Explanation
CONNECTED	Bool	TRUE, when the database is connected.
EXECUTE_DONE	Bool	TRUE if the SQL command was executed. Note: For each execution, the output generates a low edge, so that the executions can be counted with a downstream counter module or subsequent actions can be triggered.
EXECUTE_ERROR	Bool	TRUE when an error occurs.
ERROR_TEXT	String	Plain text of the error message
COLUMN_PARAMETER_NUMBER	Dint	Number of columns of the result. (Here: 2, since there are two values ID and WEIGHT in the database)
ROW_NUMBER	Dint	Number of datasets of the result. (Here: 8, since there are 8 records in the database and all have been retrieved)
COLUMN_PARAMETER_LENGTH	ICPBUF_INT	Array with display of the data length in BYTE for each result cell. (Here: 4 bytes for the INT data type in the database and 8 bytes for the REAL data type of the database)
COLUMN_TYPE	ICPBUF_INT	Array displaying the SQL data type for each result cell. (Here: 4 for the database type INTEGER and 7 for the database type REAL) 0 = unknown type 1 = CHAR 2 = NUMERIC 3 = DECIMAL 4 = INTEGER 5 = SMALLINT 6 = FLOAT 7 = REAL 8 = DOUBLE 9 = DATETIME 12 = VARCHAR
RESULT_DATA	Any	Result of the returned datasets. A suitable structure must be created for subsequent processing.

The structure created must match the receiving data.

**Note**

If you have different queries that return different types of data, you have to multiplex the received data. This is applied in the second example "Configuring StoredProcedure Calls in Oracle" and is described there.

In our example we need a structure with two members ID and WEIGHT. This corresponds to one dataset. The number of datasets returned must then be written to a suitable array of datasets.

Therefore, we create a dataset structure and use it in an array.

First the dataset structure must be defined. Please note that the data types of MS-SQL and ibaLogic sometimes have different BYTE lengths.

In our case, an MS-SQL INTEGER corresponds to an ibaLogic DINT and an MS-SQL REAL to an ibaLogic LREAL. Therefore, we create the following dataset structure.

The 'Edit Data Type' dialog box shows the configuration for 'Struct_ibaData'. The 'General' tab is active, with 'Name' set to 'Struct_ibaData' and 'Description' empty. The 'Structure properties' section shows 'Count' set to 2. The 'Members' table lists two members:

Num	Type	Name	Description	Default
1	DINT	id		
2	LREAL	weight		

The dialog includes 'OK', 'Apply', and 'Cancel' buttons at the bottom.

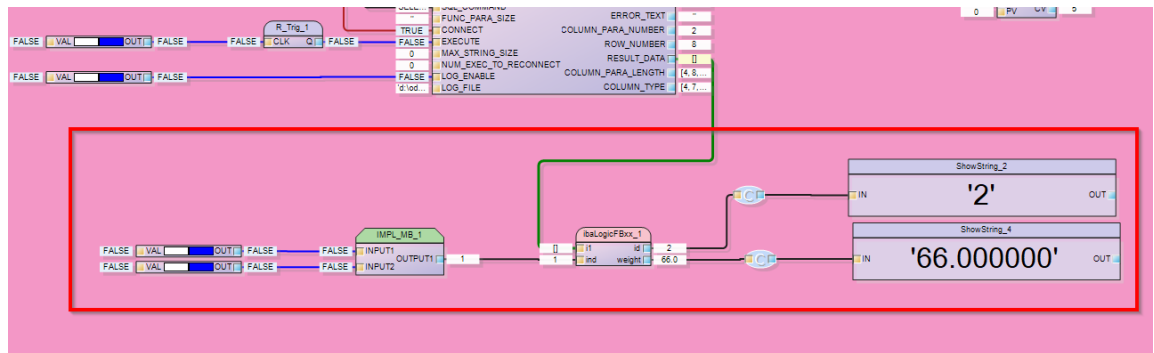
This is used in an array, which should be able to contain the maximum number of expected datasets. (here: 16)

The 'Edit Data Type' dialog box shows the configuration for 'DataSets'. The 'General' tab is active, with 'Name' set to 'DataSets' and 'Description' empty. The 'Array properties' section shows 'Type' set to 'Struct_ibaData' (highlighted with a red box) and 'Count' set to 1. The 'Elements' table lists one element:

Num	Lower Boundary	Upper Boundary
1	0	15

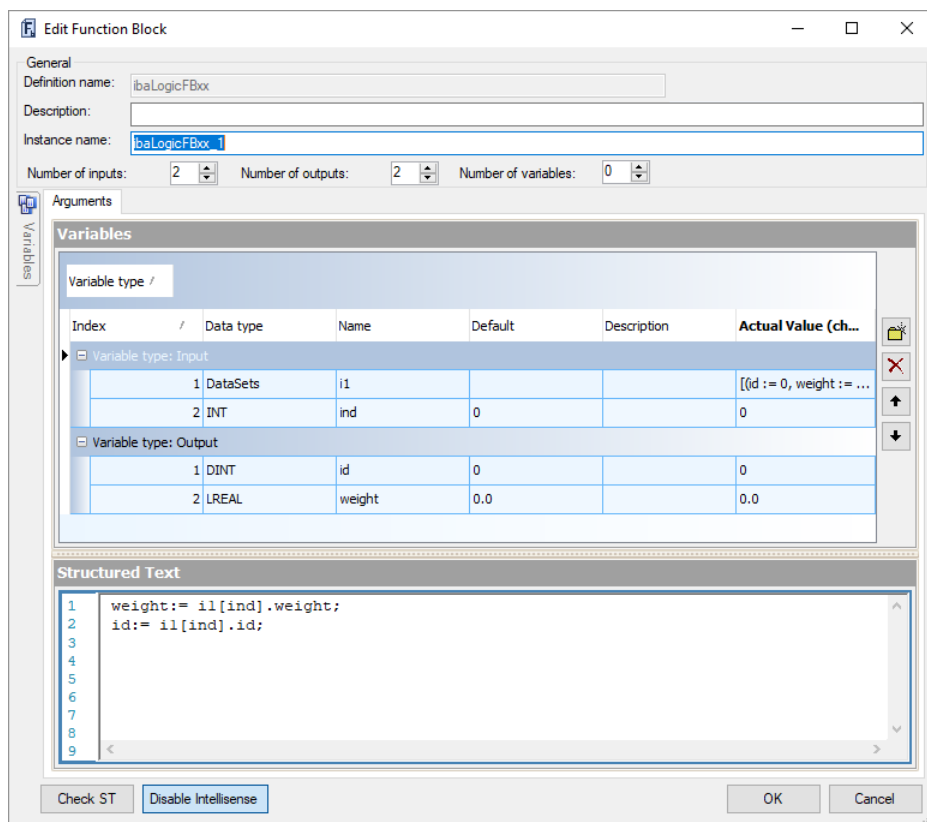
The dialog includes 'OK', 'Apply', and 'Cancel' buttons at the bottom.

Now we can read the datasets for further processing. Below is an access example:

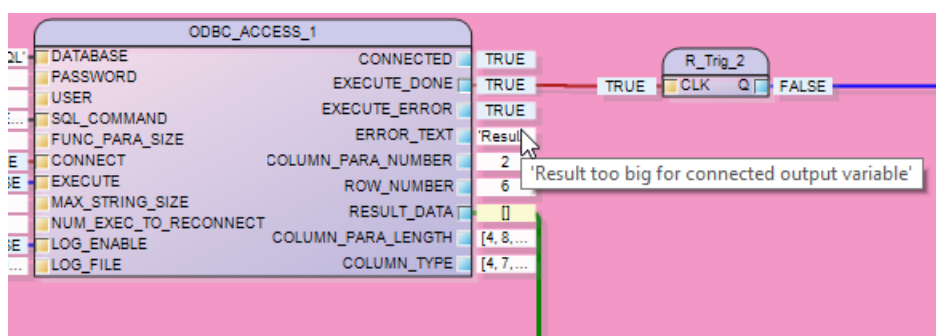


The IMPL_MB macro can be used to select an index = dataset number. The block ibaLogicFBxx then outputs the values of the respective dataset.

The content of the ibaLogicFBxx shows how to address the dataset.



If the dataset array is too small to contain all received data, you will get an error message.



Example of a dataset structure that only allows 6 datasets, but actually more have been retrieved:

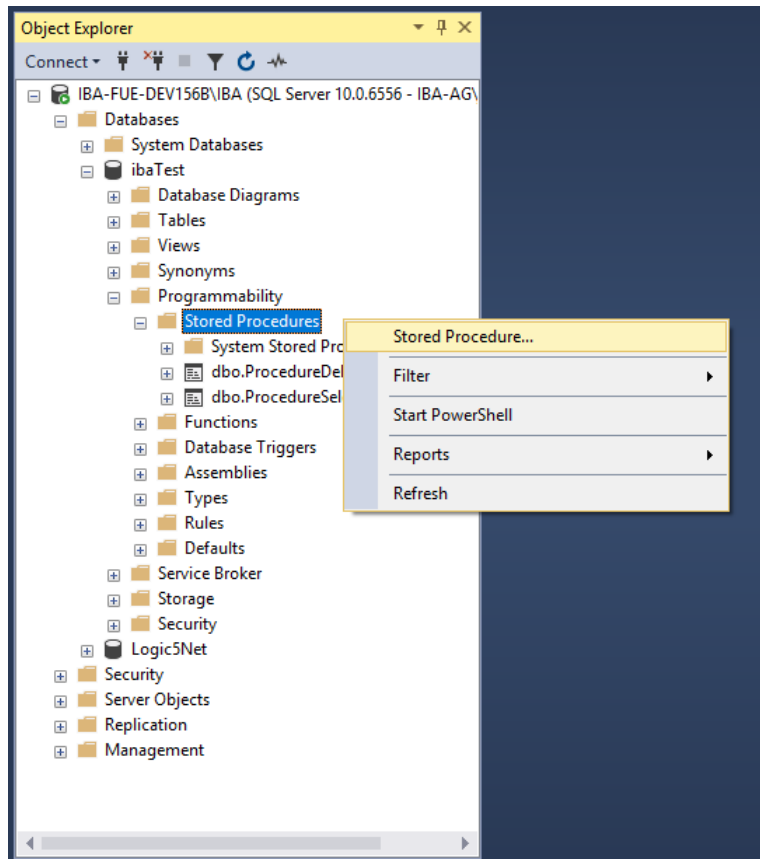
The 6 possible datasets are transferred (row number = 6) and an error message indicates that there are actually more datasets available. The user must then enlarge the structure or specifically request fewer data records.

3.4 Example for StoredProcedure in MSSQ

The example for StoredProcedure in MSSQL can be imported into ibaLogic. The import file ODBC_StoredProcedure_MSSQL.il5 is located on the DVD "iba Software & Manuals".

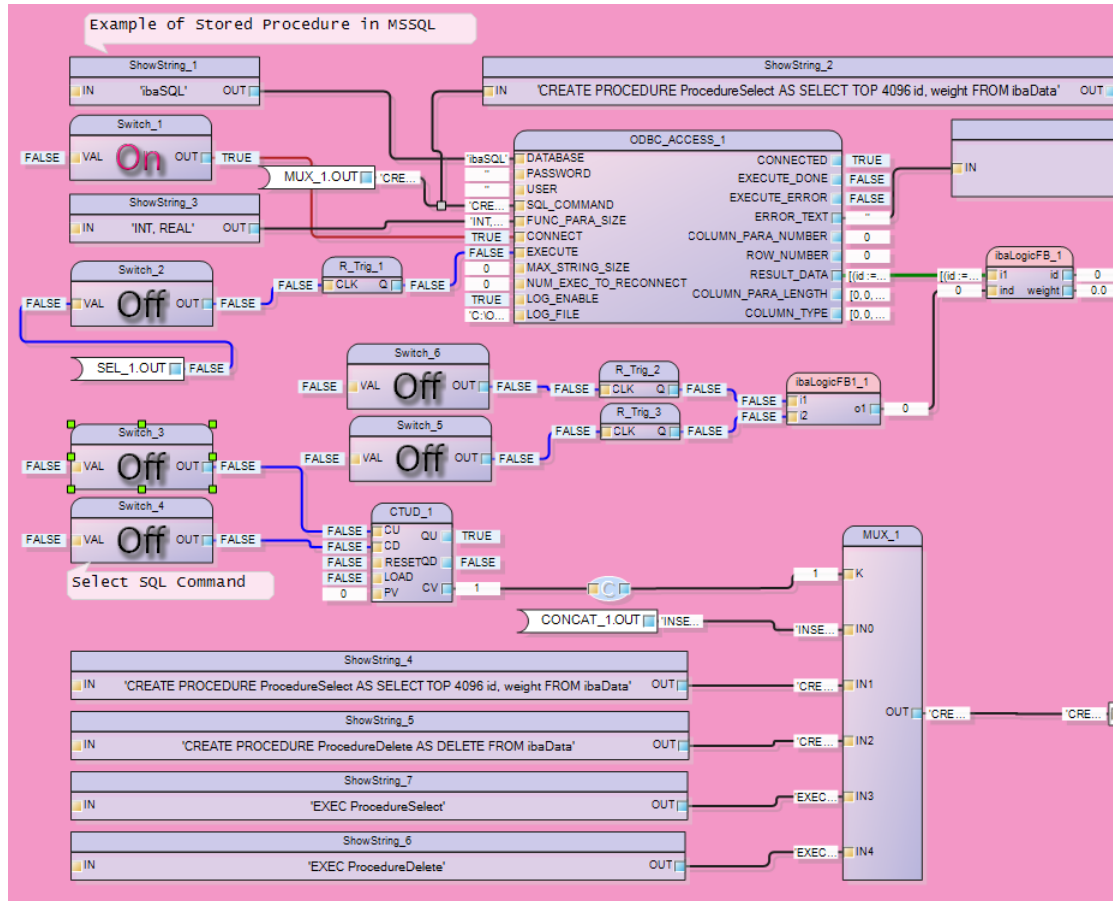
A StoredProcedure can be created in the Microsoft SQL Server Management Studio itself (see figure below) or by using the ODBC_ACCESS block with the command "CREATE PROCEDURE procedurename AS ...".

With the example program you can create a SELECT and a DELETE procedure and then execute it with the command "EXEC procedurename".



For easy handling, the already configured database connection ibaSQL with the table ibaTest and the columns ID (Int) and WEIGHT (Real) included in it is used.

Example



When starting the calculation, an INSERT command is present at the input of the ODBC_ACCESS block. To fill the database with values first, the switch Switch_7 can be activated. Test data is written to the columns every second.

The corresponding SQL commands can be selected via the buttons Switch_3 and Switch_4.

Press the button Switch_3 to select the command "CREATE PROCEDURE ProcedureSelect AS SELECT TOP 4096 id, weight FROM ibaData" or "CREATE PROCEDURE ProcedureDelete AS DELETE FROM ibaData". The command is executed by pressing the button Switch_2.

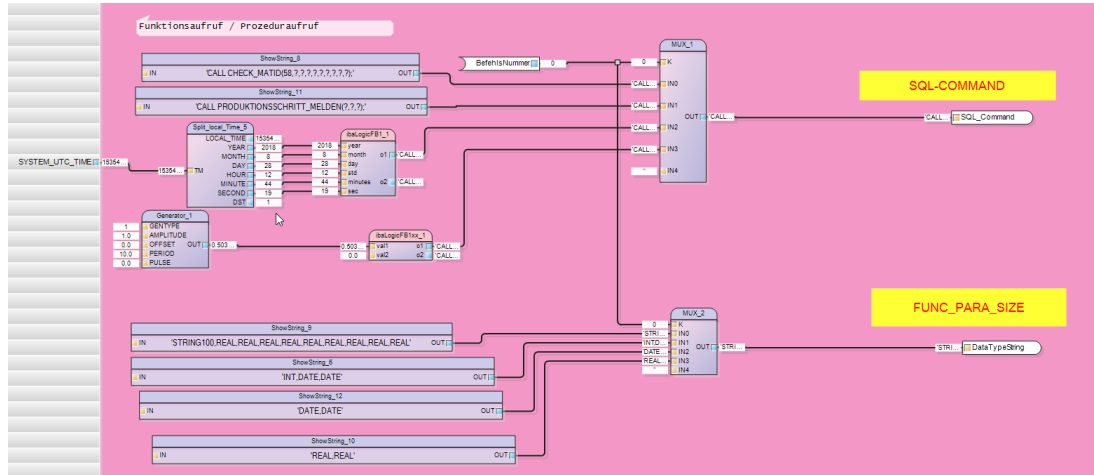
In Microsoft SQL Server Management Studio, you can check whether the procedures have been created and execute them there. To execute the procedures from ibaLogic, follow the same process as when creating the procedures and select the corresponding commands "EXEC ProcedureSelect" or "EXEC ProcedureDelete" and press button Switch_2 to trigger the execution.

At output RESULT_DATA the received data are displayed in an array, which you can view one by one by pressing the switches Switch_5 and Switch_6.

4 Configuration for calling StoredProcedure in Oracle

A configuration example "ODBC_StoredProcedure_Oracle.il5" is available on the DVD "iba Software & Manuals". Only the specific features related to the StoredProcedure are described here.

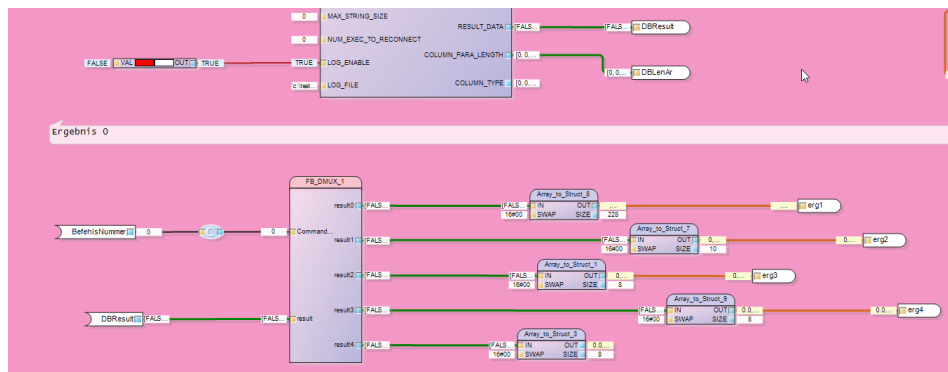
A StoredProcedure consists of the call CALL procedurename (parameter).



You can transfer values and get values back. The return values are marked with ? in the request. To ensure that the data transfer works, the data types must also be transferred. This is done by a plain text string at input FUNC_PARA_SIZE.

In this example, different procedures are called. With different parameters and thus with different results.

It is necessary to estimate the maximum size of the received data and to create a corresponding byte array at the output. Depending on the requirements, a different structure is then assigned to this array according to the expected result. The ARRAY_TO_STRUCT block is mainly used for this purpose.



The command number determines the query and also switches the result via the FB-Mux to the correct output. There, a corresponding structure is supplied by an ARRAY_TO_STRUCT block. This structure can then be used for further processing.

The multiplexer looks as follows:

The screenshot shows the 'General' tab of a configuration dialog for a multiplexer named 'FB_DMUX'. The 'Instance name' is 'FB_DMUX_1'. It has 2 inputs, 5 outputs, and 0 variables. The 'Variables' tab is active, showing a table of variables. The 'Structured Text' tab is also visible, containing a SQL-like script.

General

Definition name: FB_DMUX

Description:

Instance name: FB_DMUX_1

Number of inputs: 2 Number of outputs: 5 Number of variables: 0

Variables

Index	Data type	Name	Default	Description	Actual Value (c...
Variable type: Input					
1	INT	CommandNr	0		0
2	ODBC_SQL_Access	result			[FALSE, FALSE, FA...
Variable type: Output					
1	ODBC_SQL_Access	result0			[FALSE, FALSE, FA...
2	ODBC_SQL_Access	result1			[FALSE, FALSE, FA...
3	ODBC_SQL_Access	result2			[FALSE, FALSE, FA...

Structured Text

```
1 CASE CommandNr OF
2 0:result0:=result;
3 1:result1:=result;
4 2:result2:=result;
5 3:result3:=result;
6 4:result4:=result;
7 END_CASE;
```

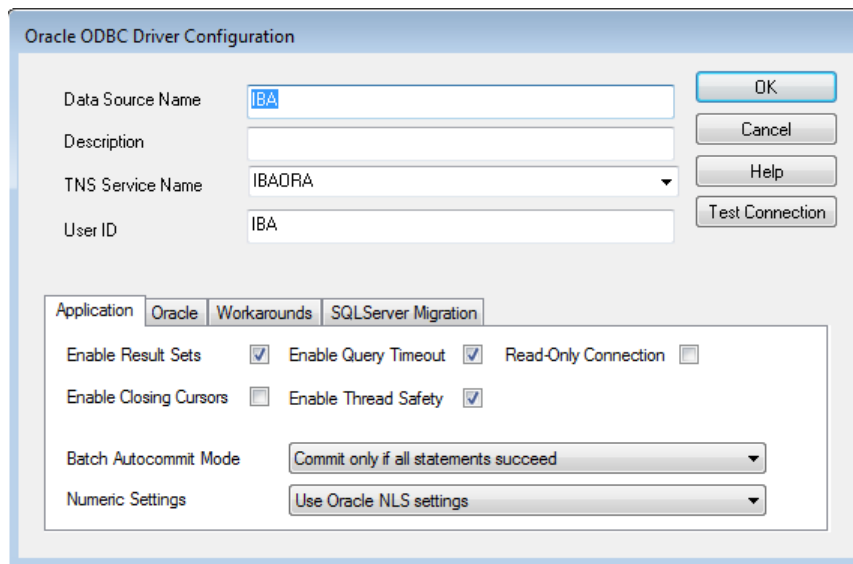
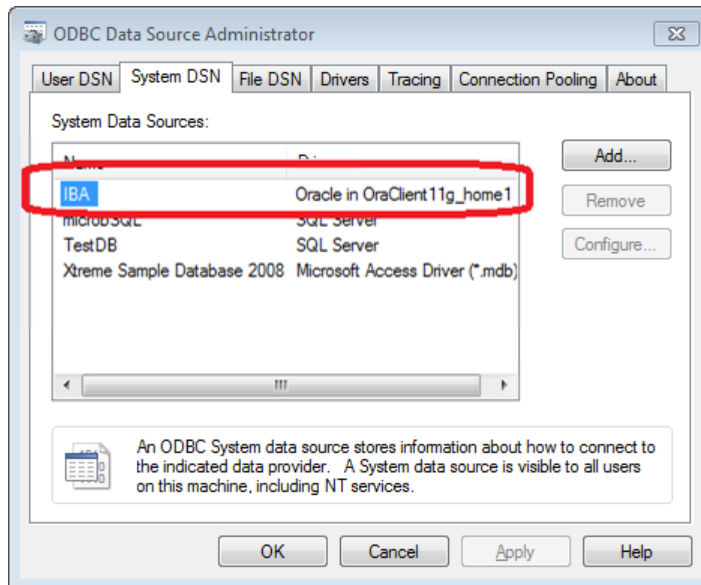
Check ST Disable Intellisense OK Cancel

Note the following regarding the structures:

- ☐ A DATE value arrives as UDINT (UTC time).
- ☐ A string value must exist in the structure according to its size. A STRING100 must be defined as data type "string of size 100" (see String derived type).
- ☐ The MaxStringSize input has priority. If it is set to 50, all strings not greater than 50 are returned. If it is set to 0, the ibaLogic string size 1024 is used.

4.1 ODBC connection to Oracle

The following settings have been made in the test for the ODBC connection to Oracle. An Oracle client must also be installed on the ibaLogic computer to have the ODBC driver available.



Oracle ODBC Driver Configuration

Data Source Name: IBA

Description:

TNS Service Name: IBADORA

User ID: IBA

OK Cancel Help Test Connection

Application Oracle Workarounds SQLServer Migration

Fetch Buffer Size: 64000

Enable LOBs: ☒

Enable Statement Caching: ☐

Cache Buffer Size: 20

Failover

Enable Failover: ☒

Retry: 10

Delay: 10

Oracle ODBC Driver Configuration

Data Source Name: IBA

Description:

TNS Service Name: IBADORA

User ID: IBA

OK Cancel Help Test Connection

Application Oracle Workarounds SQLServer Migration

Bind TIMESTAMP as DATE: ☒

Force SQL_WCHAR Support: ☒

Disable Microsoft Transaction Server: ☒

Set Metadata Id Default to SQL_TRUE: ☐

Disable SQLDescribeParam: ☐

Bind NUMBER As FLOAT: ☐

Disable RULE Hint: ☒

Oracle ODBC Driver Configuration

Data Source Name: IBA

Description:

TNS Service Name: IBADORA

User ID: IBA

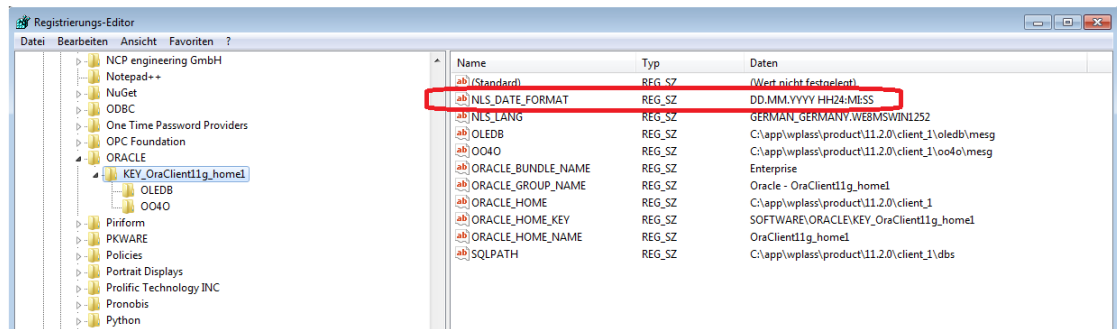
OK Cancel Help Test Connection

Application Oracle Workarounds SQLServer Migration

Enable EXEC Syntax: ☐

Schema:

Please note that the DATE format DD.MM.YYYY HH:MM:SS is expected. This format must be set, e.g. in the registry
HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_OraClient11g_home1.



Error messages:

INVALID NUMBER: This error can occur if a point is specified for real values, but the system expects a comma. This is due to the NLS settings of Oracle.

A possible remedy is an entry in the REGEDIT under HKEY_LOCAL_MACHINE—ORACLE—KEY_OraClient11g_home1 (or similar).

5 Appendix

5.1 ODBC Data types (MSSQL/ORACLE)

MS SQL DATA TYPES					
Name	MSSQL Data types	Code	Length	ibaLogic type	Comment
bit	BIT	-7	1	BYTE	Zero = 0
tinyint	TINYINT	-6	1	USINT	Zero = 0
smallint	SMALLINT	5	2	INT	Zero = 0
int	INT	4	4	DINT	Zero = 0
bigint	BIGINT	-5	19	STRING(19)	Zero = 'NULL'
smallmoney	SMALLMONEY	3	8	LREAL	Zero = NaN.0
money	MONEY	3	8	LREAL	Zero = NaN.0
numeric	NUMERIC(18,0)	2	8	LREAL	Zero = NaN.0
decimal	DECIMAL(18,0)	3	8	LREAL	Zero = NaN.0
real	REAL	7	8	LREAL	Zero = NaN.0
float	FLOAT	6	8	LREAL	Zero = NaN.0
smalldatetime	SMALLDATETIME	11	16	STRUCT(INT,INT,INT,I NT,INT,INT,DINT)	Zero = 0, 'return structure 'Year/Month/day/hour/minute/s econd/fraction'
datetime	DATETIME	11	16	STRUCT(INT,INT,INT,I NT,INT,INT,DINT)	Zero = 0, 'return structure 'Year/Month/day/hour/minute/s econd/fraction'
datetime2	DATETIME2	-9	27	STRING(27)	Zero = 'NULL'
datetimeoffset	DATETIMEOFFSET	-9	34	STRING(34)	Zero = 'NULL'
date	DATE	-9	10	STRING(10)	Zero = 'NULL'
time	TIME7	-9	16	STRING(16)	Zero = 'NULL'
char	CHAR(10)	1	10	STRING (10)	Zero = 'NULL'
varchar	VARCHAR(len)	12	x	STRING(x)	Zero = 'NULL'
varchar(max)	VARCHAR(MAX)	-10	1024	STRING	Zero = 'NULL', SELECT with CAST and CONVERT ¹
text	TEXT	-8	1024	STRING	Zero = 'NULL', SELECT with CAST and CONVERT ¹
nchar	NCHAR(10)	-8	10	STRING(10)	Zero = 'NULL'
nvarchar	NVARCHAR(50)	-9	50	STRING(50)	Zero = 'NULL'
nvarchar(max)	NVARCHAR(MAX)	-10	1024	STRING	Zero = ''
ntext	NTEXT	-10	1024	STRING	Zero = ''
binary	BINARY(50)	-2	50	BYTE(50)	Zero = 0
varbinary	VARBINARY(50)	-3	50	BYTE(50)	Zero = 0
varbinary(max)	VARBINARY(MAX)	x	x	not supported	no SELECT
image	IMAGE	x	x	not supported	no SELECT
sqlvariant	SQL_VARIANT	-150	1024	STRING	Zero = 'NULL'

¹ SELECT CAST(column name AS CHAR(1024)) FROM table name
SELECT CONVERT(CHAR(1024), column name) FROM table name
The target size of 1024 is selected randomly. Any other size is possible as long as it fits into the target data type.

MS SQL DATA TYPES					
Name	MSSQL Data types	Code	Length	ibaLogic type	Comment
xml	XML	-10	128	not supported	
uniqueidentifier	UNIQUEIDENTIFIER	-11	16	not supported	
timestamp	TIMESTAMP	-2	8	not supported	is supplied by DB
geography				not supported	
geometry				not supported	
hierarchyid				not supported	
cursor				not supported	
table				not supported	

ORACLE DATA TYPES					
Name	Oracle Data type	Code	Length	ibaLogic type	Comment
bfile	BFILE	?	?	not supported	no SELECT
binary_double	BINARY_DOUBLE	8	8	LREAL	Zero = NaN.0
binary_float	BINARY_FLOAT	7	8	LREAL	Zero = NaN.0
blob	BLOB	12	1024	STRING	Zero = 'NULL'
character	CHAR(1 BYTE)	1	1	STRING(1)	NULL = 'N'
clob	CLOB	12	1024	STRING	Zero = 'NULL'
date	DATE	12	19	STRING TO_CHAR	Zero = 'NULL'
real	FLOAT	6	8	LREAL	Zero = NaN.0
float	FLOAT	6	8	LREAL	Zero = NaN.0
double_precision	FLOAT	6	8	LREAL	Zero = NaN.0
interval_day	INTERVAL DAY(2) TO SECOND(6) !!TO_CHAR!	12	20	STRING(20)	Zero = 'NULL'
interval_year	INTERVAL YEAR(2) TO MONTH !!TO_CHAR	12	7	STRING(7)	Zero = 'NULL'
nchar	NCHAR(1 CHAR)	-8	1	STRING(1)	Zero = 'N'
national_char	NCHAR(1 CHAR)	-8	1	STRING(1)	Zero = 'N'
number	NUMBER	6	8	LREAL	Zero = NaN.0
int	NUMBER(38,0)	3	8	LREAL	Zero = NaN.0
integer	NUMBER(38,0)	3	8	LREAL	Zero = NaN.0
numeric	NUMBER(38,0)	3	8	LREAL	Zero = NaN.0
dec	NUMBER(38,0)	3	8	LREAL	Zero = NaN.0
decimal	NUMBER(38,0)	3	8	LREAL	Zero = NaN.0
nchar_varying	NVARCHAR2(20 CHAR)	-9	20	STRING(20)	Zero = 'NULL'
national_char_varying	NVARCHAR2(20 CHAR)	-9	20	STRING(20)	Zero = 'NULL'
nvarchar2	NVARCHAR2(20 CHAR)	-9	20	STRING(20)	Zero = 'NULL'

ORACLE DATA TYPES					
Name	Oracle Data type	Code	Length	ibaLogic type	Comment
raw	RAW	12	1024	not supported	no SELECT
rowid	ROWID	12	18	not supported	no SELECT
timestamp	TIMESTAMP(6)	12	19	STRING TO_CHAR	Zero = 'NULL'
urowid	UROWID	12	18	not supported	no SELECT
character_varying	VARCHAR2(20 BYTE)	12	20	STRING(20)	Zero = 'NULL'
varchar2 (20)	VARCHAR2(20 BYTE)	12	20	STRING(20)	Zero = 'NULL'
char_varying	VARCHAR2(20 BYTE)	12	20	STRING(20)	Zero = 'NULL'
long_raw				not supported	
nclob				not supported	
smallint				not supported	

6 Support and contact

Support

Phone: +49 911 97282-14
Fax: +49 911 97282-33
Email: support@iba-ag.com



Note

If you require support, indicate the serial number (iba-S/N) of the product.

Contact

Headquarters

iba AG
Koenigswarterstr. 44
90762 Fuerth
Germany

Phone: +49 911 97282-0
Fax: +49 911 97282-33
Email: iba@iba-ag.com
Contact: Mr Harald Opel

Regional and worldwide

For contact data of your regional iba office or representative please refer to our web site

www.iba-ag.com.