mapp Database: Anbindung an Datenbank

mapp Database stellt eine Verbindung mit einer Datenbank her und kann Informationen abfragen, aktualisieren, löschen und vieles mehr.

Auf unserem B&R Tutorial Portal sind Tutorials zum Thema mapp Database (GER/ENG) zu finden. (Zugangsdaten)

1 Konzept

Allgemeines

- Als Endnutzer möchte ich das Datenbanksystem verwenden, welches wir seit langer Zeit in unserer Fabrik in Verwendung haben
- Als Applikationist möchte ich, dass ich Informationen in einer Datenbank anfordern oder abspeichern kann
- Als Applikationist möchte ich ich mich nicht im Detail mit der Datenbank-Sprache SQL auseinandersetzen müssen
- Als Applikationist mit mehr Datenbankwissen möchte ich alle Möglichkeiten der Datenbank, sowie von SQL nutzen können

Daten visuell darstellen

- Als Endnutzer möchte ich unterschiedliche Abfragen bequem und einfach über meine Visualisierung starten können sowie tabellarisch oder grafisch darstellen

1.1 Grundprinzip

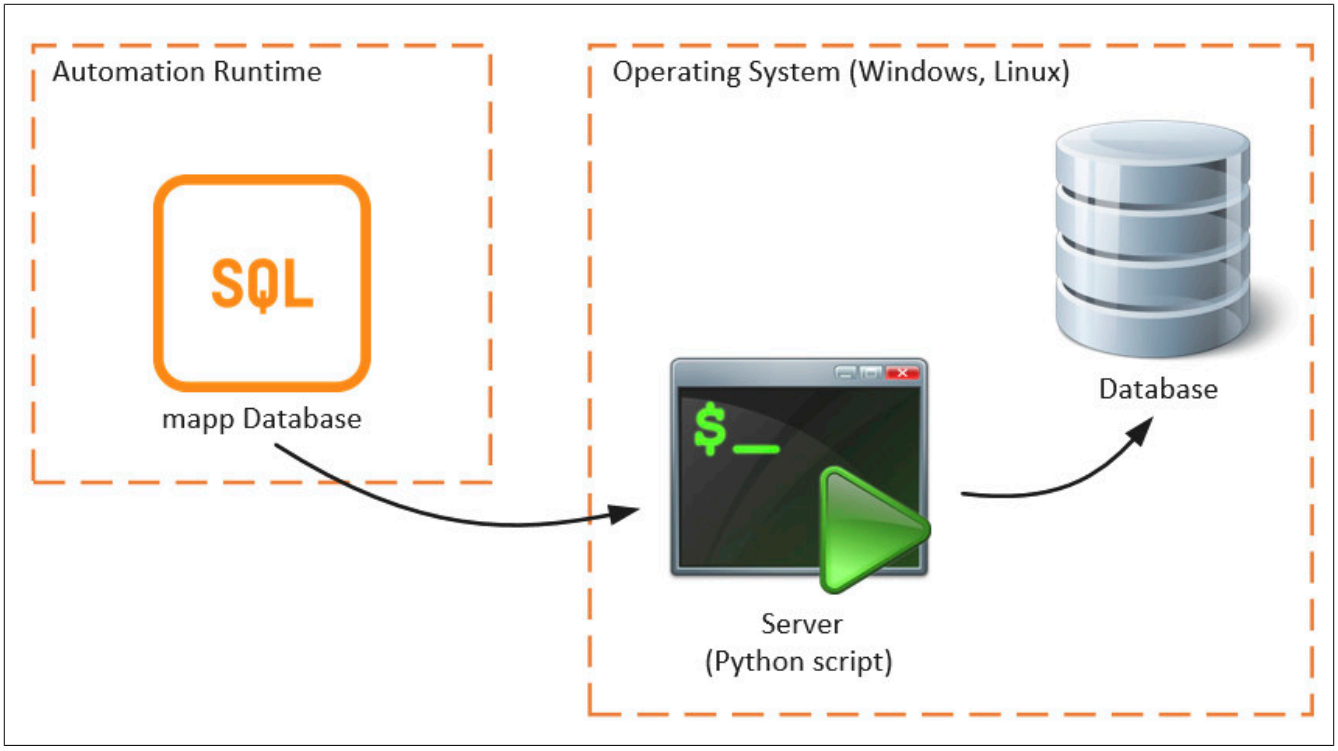
Mithilfe von mapp Database kann eine Verbindung zu einer Datenbank hergestellt werden. In einer Datenbank können große Datenmengen effizient gespeichert werden. Welche Datenbanken in Kombination mit mapp Database verwendet werden können, ist im Abschnitt Datenbank herunterladen und konfigurieren erklärt.

Für allgemeine Informationen zu Datenbanken siehe hier.

Mit mapp Database können in einer Datenbank mithilfe von Abfragen, sogenannten Queries, ein Tabelle erstellt, Einträge hinzugefügt, gelöscht oder angepasst werden. Der Anwender kann die Anfragen über eine vereinfachte Konfigruationsschnittstelle erstellen, oder direkt mit SQL-Abfragen arbeiten. Mehr Information zu dem Thema kann in der MpDatabaseCore Konfiguration im Abschnitt Abfrage-Typen gefunden werden.

Die Kommunikation zwischen der Steuerung und der Datenbank läuft über einen zwischengelagerten HTTP-Server ab. Die Steuerung schickt die Anfragen an einen HTTP-Server. Der HTTP-Server wird über ein Python-Script verwaltet. Dieses Script verarbeitet die von der Steuerung empfangenen Daten weiter und übermittelt sie als SQL-Anfrage an die Datenbank. Die Datenbank schickt die entsprechenden Daten zurück, bzw. führt die gewünschte Aktion aus. Der Vorteil dieser Löung liegt darin, dass es für den PC auf welchem das Script läuft auch die unterschiedlichsten Datenbank-Konnektoren gibt, welche für gängige Betriebssysteme verfügbar sind. Um eine spezielle Datenbank zu verwenden muss dadurch nur der entsprechende Konnektor installiert werden und möglicherweise das Skript angepasst werden.

Wie das Skript angepasst und die Kommunikation Schritt für Schritt aufgebaut wird, ist im Abschnitt Verbindung mit Datenbank herstellen erklärt.



1.2 Verbindung mit Datenbank herstellen

Mithilfe von mapp Database ist es möglich eine Verbindung zu einer Datenbank herzustellen. Doch bevor mapp Database implementiert werden kann, sind einige Schritte notwendig. In diesem Abschnitt wird Schritt für Schritt erklärt wie die Verbindung mit einer Datenbank realisiert werden kann.

1.2.1 Datenbank herunterladen und konfigurieren

Um mit mapp Database arbeiten zu können, muss ein lauffähiges Datenbanksystem am Betriebssystem installiert sein.Folgende Datenbanken wurden in Kombination mit mapp Database getestet:

- MySQL (getestet mit Version 8.0)
- Microsoft SQL Server (getestet mit Version 2017)
  Beid der Verwendung vom Microsoft SQL Server muss auch der Microsoft SQL Server Native Client installiert werden. Dieser kann hier heruntergeladen werden.
- MariaDB (getestet mit Version 10.0.34)
- PostgreSQL (getestet mit Version 10.5)

Es ist auch möglich andere Datenbanken zu verwenden. Dabei muss ein passender Konnektor heruntergeladen und das Skript dementsprechend angepasst werden. Für mehr Information siehe Abschnitt Server installieren.

Wie die Datenbank konfiguriert werden müssen, ist in der Dokumentation in der jeweiligen Datenbank zu finden. Folgende Information werden in den nächsten Schritten von der Datenbank benötigt:

- IP-Adresse
- Port-Nummer
- Benutzername
- Passwort

1.2.2 Server installieren

Wie im Abschnitt Grundprinzip erkärt, läuft die Kommunikation zwischen mapp Database und der Datenbank über einen HTTP-Server. Dieser wird durch ein Skript gestartet. Das Skript wird mit Python ausgeführt. Python kann hier heruntergeladen werden.

mapp Database unterstützt die Python Version 2.7 / 32-Bit Version sowie die Python-Version 3.8.2 / 32 Bit-Version. Mehr Information dazu kann im Abschnitt Definieren des Python-Skripts gefunden werden.

Damit die Kommunikation zwischen HTTP-Server und Datenbank funktioniert, muss ein Python Konnektor installiert werden. Welcher dabei verwendet wird ist in der Dokumentation der jeweiligen Datenbank zu finden.

Unabhängig davon welches Datenbank-System verwendet wird, muss zusätzlich der **MySQL Konnektor** installiert werden. Grund dafür ist, dass die Datentypen in mapp Database auf MySQL basieren. Mithilfe des MySQL Konnektors werden Datentypen von anderen Datenbank-Systemen automatisch korrekt von mapp Database interpretiert.

Der MySQL-Konnektor kann zum Beispiel hier gefunden werden.

Je nach verwendeter Datenbank-Version kann sich die Python Konnektor-Version ändern. Dies sollte beim Verwenden einer neuen Datenbank-Version immer überprüft werden!

Abhängig davon welches Datenbank-System verwendet wird, müssen folgende Python Konnektoren verwendet werden:

Datenbank-System	Konnektoren
MySQL, MariaDB	• MySQL-Konnektor (Version: 8.0 / 32-Bit Version)
MS SQL	• MySQL-Konnektor (Version: 8.0 / 32 Bit-Version) • MS SQL pyodbc Konnektor (Version: 4.0 / 32-Bit Version)
PostgreSQL	• MySQL-Konnektor (Version: 8.0 / 32-Bit Version) • PostgreSQL psycopg Konnektor (Version: 2.7)

Im Abschnitt Rahmenbedingungen wird unter "Software-Versionen" nocheinmal zusammenfassend aufgelistet, welches Datenbank-System welche Software benötigt.

Nach der Installation der Python-Version und zu der Datenbank passenden Konnektoren muss ein Python-Skript definiert werden. Im Anschluss wird das Skript gestartet.

1.2.2.1 Definieren des Python-Skripts

Durch das Python-Skript wird mithilfe des installierten MySQL-Konnektors eine Verbindung mit einer Datenbank hergestellt.

Abhängig davon welche Python-Version installiert wurde, muss das dazugehörtige Python-Skript verwendet werden.

Nach herunterladen des benötigten Python-Skripts muss das Skript gestartet werden. Dies wird im Abschnitt Starten des Skripts erklärt.

Python Version 2.7

Wird die Python Version 2.7 verwendet, so können folgende Python-Skripts heruntergeladen werden:

Python-Skript	
Download	**Version**
mappDatabaseConnector	5.4.0
mappDatabaseConnector	5.5.0
mappDatabaseConnector	5.7.0
mappDatabaseConnector	5.8.0
mappDatabaseConnector	5.11.0
mappDatabaseConnector	5.12.0

Python Version 3.8

Wird die Python Vesion 3.8 verwendet, so können folgende Python-Skripts heruntergeladen werden:

Python-Skript	
Download	**Version**
mappDatabaseConnector	5.12.0

Features

Abhängig von der verwendeten Python-Skript Version werden folgende Features in mapp Database unterstützt:

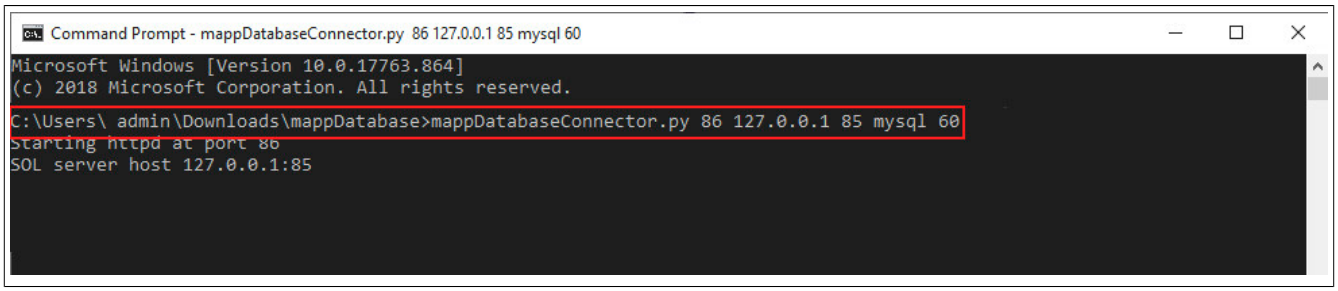
Feature	Python Version 2.7 Skript-Version	Python Version 3.8 Skript-Version
Unterstützung des Datenbanksystems PostgreSQL Kommunikation zum Database-Widget	5.8.0	5.12
Erneuter Verbindungsaufbau zwischen Datenbank und Zielsystem nach beispielsweise Neustart des Zielsystems	5.11.0	
Überwachung der verwendeten Skript-, und mapp Services-Version	5.12.0	

1.2.2.2 Starten des Skripts

Das Skript wird mithilfe einer Kommando-Konsole (In Windows "cmd" in Linux Debian "Terminal") gestartet. In der Konsole muss anschließend folgendes eingegeben werden:

- Pfad wo sich das Skript befindet
- Port-Nummer auf dem das Skript läuft. Es kann ein belibig freier Port angegeben werden. Die Port-Nummer muss mit der Port-Nummer in der MpDatabaseCore Konfiguration übereinstimmen.
- IP-Adresse des PCs auf dem die Datenbank läuft. In den meisten Fällen wird dies der gleich PC sein, auf welchem auch das Skript ausgeführt wird, daher kann 127.0.0.1 verwendet werden.
- Port-Nummer auf dem die Datenbank läuft. Die Port-Nummer wird beim konfigurieren der Datenbank angegeben.
- Der nächste Parameter bestimmt den Datenbank-Typ. Dabei können folgende Kürzel verwendet werden.
  mysql: Verwendung für MySQL-Datenbanksystem.
  mssql: Verwendung für MS SQL-Datenbanksystem.
  postgres: Verwendung für PostgreSQL-Datenbanksystem.
- Der letzte Parameter legt die Timeout-Zeit fest. Für mehr Information siehe weiter unten im Abschnitt "Verbindungsaufbau zur Datenbank nach Neustart des Zielsystems". Der Parameter kann nur genutzt werden, wenn die Skript-Version 5.11 verwendet wird. Die Timeout-Zeit sollte zwischen >= 10 Sekunden und < 10 Minuten (600 Sekunden) liegen. Wird eine Timeout-Zeit von unter 10 Sekunden definiert, so wird an der Konsole die Meldung "Timeout not valid. Please introduce a timeout highter than 10 seconds" angezeigt. Das Skript wird dabei nicht gestartet. Es muss eine Timeout-Zeit von >=10 Sekunden definiert werden. Wird die Timeout-Zeit >= 10 Minuten angegeben, wie beispielsweise 5000 Sekunden, so erscheint die Warnung "Warning: In case the PLC got restarted while a connection between mapp Database and the script was active a timeout of 5000 seconds will pass before connection is established again". Das Skript wird aber trotz der höheren Timeout-Zeit gestartet!

Dies kann in der Konsole zum Beispiel folgendermaßen aussehen:



Im Beispiel verwendet der HTTP-Server den Port 86 und die Datenbank den Port 85. Als Timeout-Zeit wurden 60 Sekunden festgelegt.

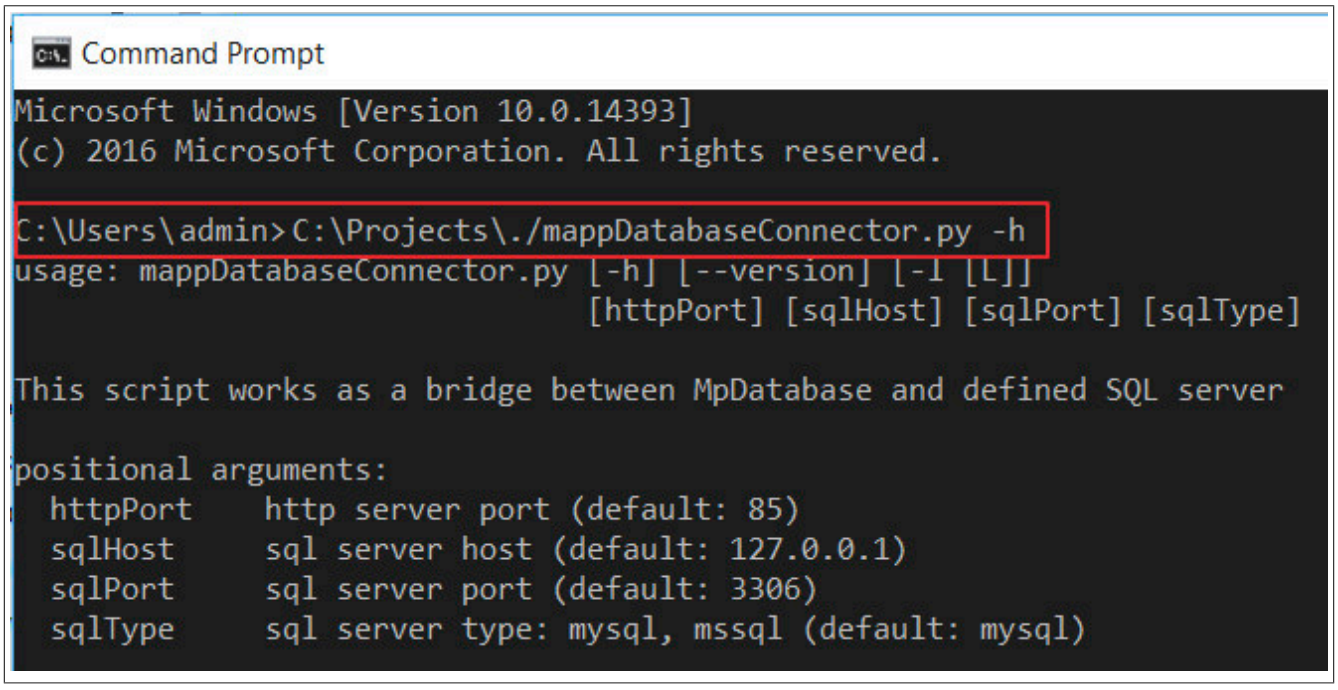
Wird zum Beispiel das Datenbank-System Microsoft SQL Server ("mssql") verwendet, so kann dies folgendermaßen aussehen:

```
Path\of\Scipt\./mappDatabaseConnector.py 86 127.0.0.1 85 mssql
```

> **Will man sich mit einem anderen Datenbank-System verbinden, so kann das Skript beliebig angepasst und geändert werden.**
>
> **Da Python ein Open-Source Programm ist, kann es vom B&R Support nicht unterstützt werden!**

Über dieses Kommando-Fenster kann zum Beispiel auch herausgefunden werden, ob Probleme beim Verbindungsaufbau aufgetreten sind. Für mehr Information dazu siehe FAQ.

Weitere Information und Beispiele können auch über das Kommando-Fenster angezeigt werden. Dafür muss "./mappDatabaseConnector.py -h" angegeben werden:



> **Sind am verwendeten System die Python-Versionen 2.7 und 3.8 installiert, so gilt es zu beachten, dass ein Skript standardmäßig mit der neuesten Python-Version, also 3.8, gestartet wird.**
> **Soll ein Skript mit der Python-Version 2.7 gestartet werden, so muss beim Aufrufen des Skripts das Kürzel "py -2" verwendet werden. Dies kann folgendermaßen aussehen:**
>
> ```
> py -2 mappDatabaseConnector.py 86 127.0.0.1 3306 mysql 60
> ```

Verbindungsaufbau zur Datenbank nach Neustart des Zielsystems

Nach einem Neustart des Zielsystems muss die Verbindung zwischen mapp Database und der Datenbank neu aufgebaut werden. Um einen problemlosen Verbindungs-Aufbau zu gewährleisten, wird beim erstmaligen Starten des Skripts eine Timeout-Zeit festgelegt. Fährt die Steuerung nach beispielsweise einem Stromausfall hoch und mapp Database wird aktiviert, so kann eine erneute Verbindung mit dem Skript nach Ablauf der Timeout-Zeit gewährleistet werden.

Vorteil davon ist, dass das Skript nach einem Neustart der Steuerung nicht auch neu gestartet werden muss. Nach Ablauf der Timeout-Zeit ist sichergestellt, dass das Skript eine erneute Verbindung mit mapp Database zulässt. Das Kommando "Connect = `TRUE`" an MpDatabaseCore kann gestartet werden. Abfragen können im Anschluss wieder gestartet werden.

Wird das Kommando "Connect = `TRUE`" vor Ablauf der Timeout-Zeit gestartet bleibt der Ausgang "CommandBusy" solange `TRUE` bis das die Timeout-Zeit abgelaufen ist. Erst danach wird "CommandBusy = `FALSE`" und "CommandDone = `TRUE`". Die Verbindung zur Datenbank ist wieder hergestellt.

Dieses Verhalten wird ab der Skript-Version 5.11 gewährleistet.

1.2.3 mapp Database implementieren

Im letzen Schritt muss mapp Database implementiert werden.

In der MpDatabaseCore Konfiguration wir die IP-Adresse, Portnummer etc. der Datenbank festgelegt.

Die Verbindung wird mithilfe des Funktionsbausteins MpDatabaseCore realisiert. Mithilfe von Abfragen, welche ebenfalls in der MpDatabaseCore Konfiguration definiert werden, können Informationen von der Datenbank auf Prozessvariablen abgespeichert werden. Die Abfragen werden mithilfe von MpDatabaseQuery durchgeführt.

Mithilfe des Database-Widgets kann der Inhalt einer Datenbank in einer mapp View Visualisierung angezeigt werden. Für mehr Information dazu siehe hier.

Wie die Konfiguration bearbeitet und wie der Funktionsbaustein MpDatabaseCore implementiert wird, ist im Getting Started Verbindung mit Datenbank herstellen erklärt.

Wie Abfragen in der MpDatabaseCore Konfiguration definiert und mit MpDatabaseQuery ausgeführt werden können, wird im Anwendungsfall Werte von Datenbank abfragen erklärt.

1.3 Best Practise

In diesem Abschnitt wird erklärt worauf man bei der Implementierung von mapp Database achten sollte.

Ergebnisse einer Abfrage limitieren

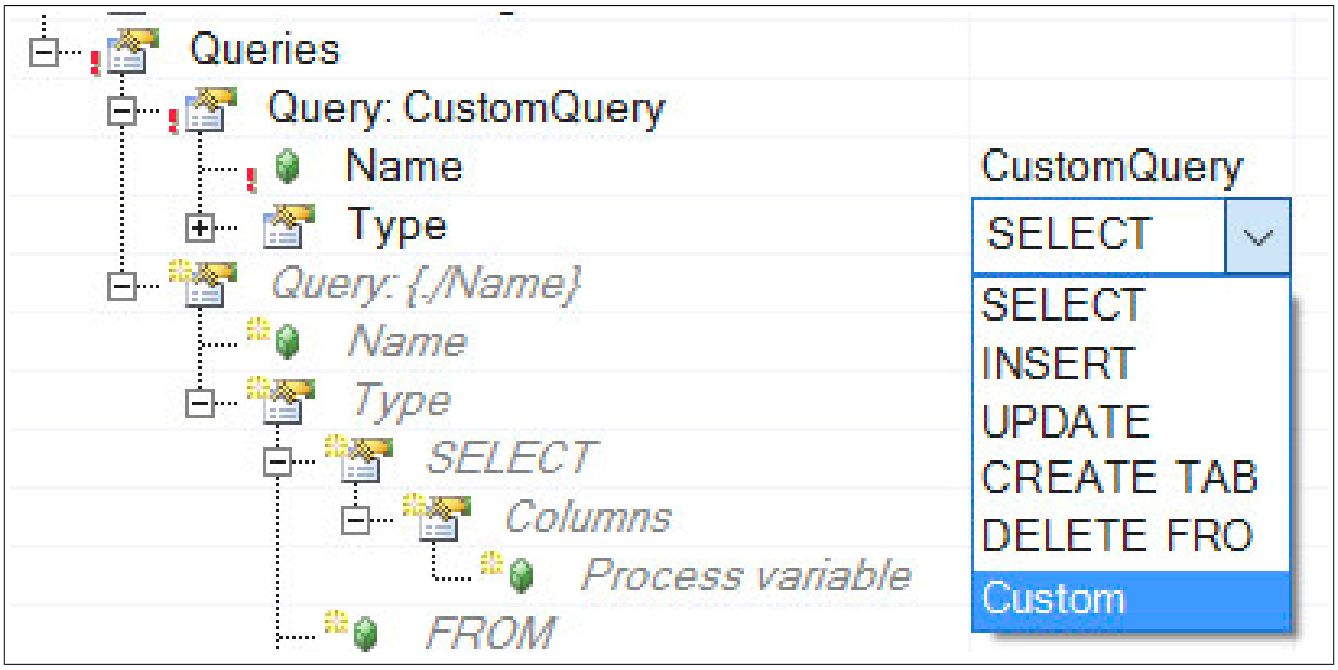
Mithilfe eines Schlüsselworts kann die maximale Anzahl an Ergebnissen einer Abfrage festgelegt werden. Somit hört das Datenbanksystem nach x Werten auf nach weiteren zu suchen. Welches Schlüsselwort dabei verwendet werden muss, ist in der Dokumentation der jeweiligen Datenbank zu finden.

Bei der Verwendung einer MySQL Datenbank wird "LIMIT" verwendet. In einer MSSQL wird anstelle "LIMIT" "TOP" angegeben.

Beispiel:

In einer MySQL Datenbank befinden sich 30 000 Einträge. Eine Abfrage trifft auf 10 000 Einträge zu, dass bedeutet es würden 10 000 Einträge an mapp Database zurückgeliefert werden. Um das Ergebnis der Abfrage zu limitieren, kann das "LIMIT" Schlüsselwort verwendet werden. Somit kann das Ergebnis, auf zum Beispiel 100 Einträge, limitiert werden. Dadurch werden an mapp Database nur die ersten 100 Einträge auf denen die Abfrage zutrifft, geschickt.

Um das "LIMIT" Schlüsselwort zu verwenden, muss in der MpDatabaseCore Konfiguration der Abfrage-Typ "Custom" verwendet werden:



Die Abfrage kann dabei folgendermaßen aussehen:

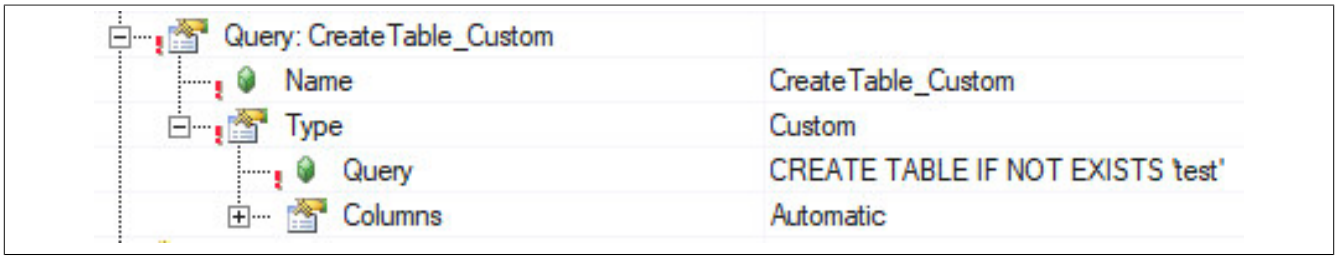
```
SELECT * from testtable WHERE id = 1 LIMIT 100
```

In MSSQL muss anstelle von "LIMIT" "TOP" definiert werden:

```
SELECT TOP 100 * FROM testtable WHERE id = 1
```

Erzeugen von Tabellen

Mithilfe des Abfrage-Typen "CREATE TABLE" können Tabellen erstellt werden. Wird die Abfrage mehr als einmal ausgeführt erscheint automatisch ein Fehler, da die Tabelle ja bereits beim erstmaligen Aufrufen erstellt wurde. Um dies zum umgehen, kann auch eine benutzerdefinierte Abrage (Abrage-Typ = "Custom") erstellt werden, wobei "CREATE TABLE" mit beispielsweise dem Schlüsselwort "IF NOT EXIST" oder "REPLACE" kombiniert werden kann.



Datenbehandlung in der Datenbank

In einer beispielsweisen MySQL Datenbank können innerhalb der Datenbank Berechnungen durchgeführt werden. Dabei können Summen, Mittlerwerte und vieles mehr kalkuliert werden. Solche Berechnungen sollten aus Performance-Gründen immer innerhalb der Datenbank und nicht am Zielsystem durchgeführt werden. Das Zielsystem wird dadurch entlastet und Berechnungen können schnell und einfach direkt in der dafür konstruierten Datenbank durchgeführt werden.

Darstellen von Sonderzeichen

Bei der Verwendung von Strings muss unbedingt auf die vewendeten Zeichen geachtet werden. Es muss klar sein ob ASCII-Zeichen, erweiterte ASCII-Zeichen etc. benötigt werden. Nur so kann sichergestellt werden, dass die verwendeten Zeichen korrekt in der Datenbank abgespeichert bzw. auch korrekt angezeigt werden.

Im Zweifelsfall kann mit WSTRING gearbeitet werden. Wie WSTRING-Variablen verwendet werden, ist im Abschnitt Rahmenbedingungen zu finden.

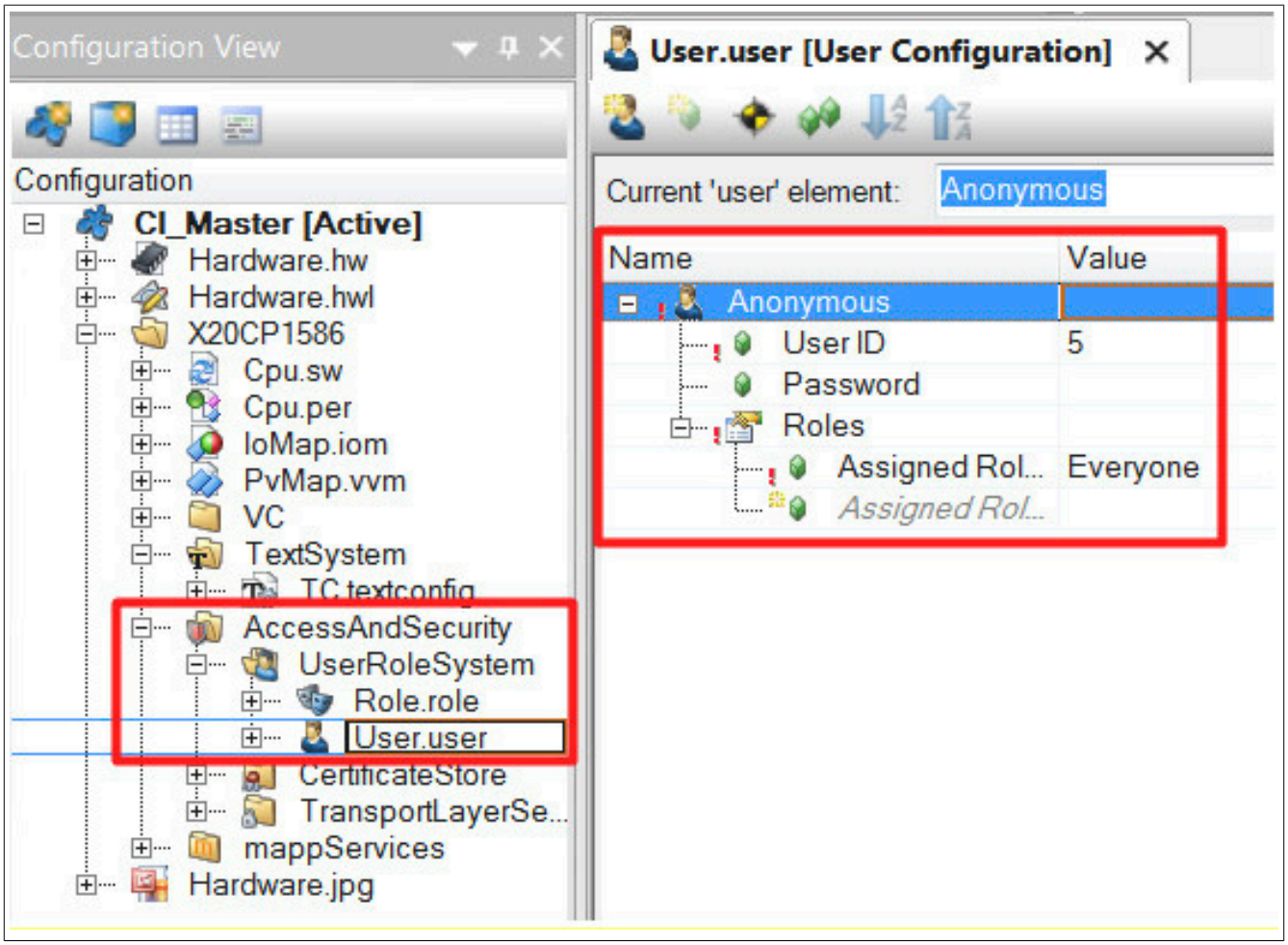
Verwendung von Datentypen

Bei der Verwendung von Datentypen muss auf die richtige Angabe der Größe sowie auf das verwendete Vorzeichen geachtet werden. Welcher Datenbank-Datentyp mit welchen Automation Studio Datentypen verwendet werden soll, ist hier im Abschnitt "Angabe von Variablen" zu finden.

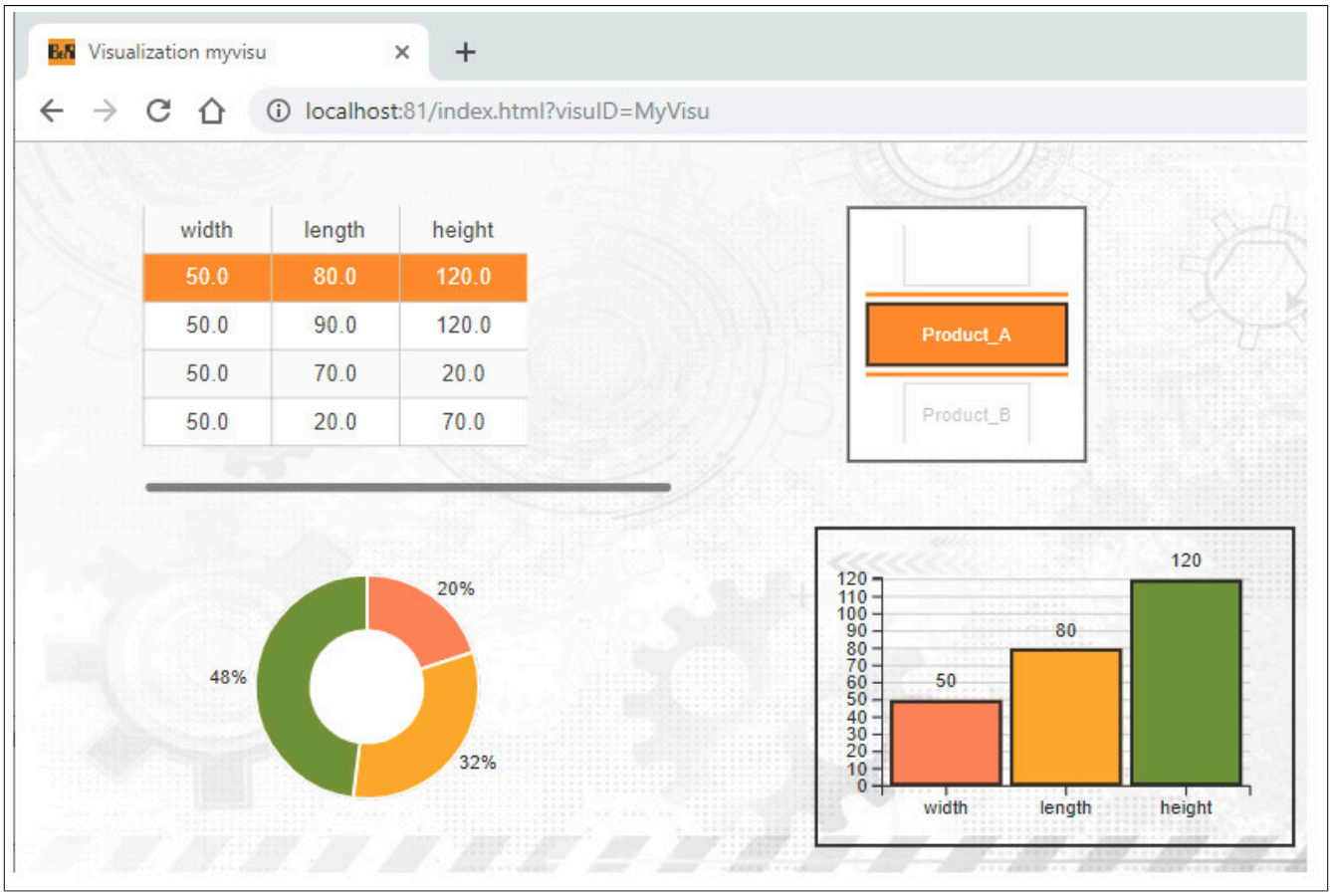
1.4 Anbindung an mapp View

Bei der Verwendung von mapp Database in Kombination mit dem Database-Widget, kann die Datenbank-Information schnell und einfach in einer mapp View Visualisierung dargestellt werden. Das Widget stellt mehrere Aktionen und Ereignisse zur Verfügung um mit Datenbank zu interagieren.

Es gilt zu beachten, dass bei der Verwendung von mapp View der Benutzer "Anonymous" mit der Rolle "Everyone" benötigt wird. Diese sind standardmäßig im Benutzer-Rollen-System von Automation Studio enthalten und sollten nicht verändert werden.



Auf unserem B&R Tutorial Portal sind Tutorials zum Thema mapp Database und dem Database-Widget (GER/ENG) zu finden. (Zugangsdaten)



1.5 Rahmenbedingungen

In diesem Abschnitt werden Rahmenbedingungen von mapp Database aufgelistet.

Software-Versionen

In folgenden Tabellen wird aufgelistet, welche Software-Versionen für welches Datenbank-System verwendet werden müssen.

mapp Database untersützt die Python Version 2.7 und 3.8. Für mehr Information dazu siehe hier.

MySQL:

Software	Version	Download
MySQL Server	getestet mit Version 8.0	Link
Python	2.7.15 / 32-Bit Version | 3.8.2 / 32-Bit Version	Link
MySQL-Konnektor	8.0 / 32-Bit Version	Link

Maria DB:

Software	Version	Download
MariaDB Server	getestet mit Version 10.0.34	Link
Python	2.7.15 / 32-Bit Version | 3.8.2 / 32-Bit Version	Link
MySQL-Konnektor	8.0 / 32-Bit Version	Link

MS SQL:

Software	Version	Download
MS SQL Server	getestet mit Version 2017	Link
Python	2.7.15 / 32-Bit Version | 3.8.2 / 32-Bit Version	Link
MySQL-Konnektor	8.0 / 32-Bit Version	Link
MS SQL pyodbc Konnektor	4.0 / 32-Bit Version	Link

PostgreSQL:

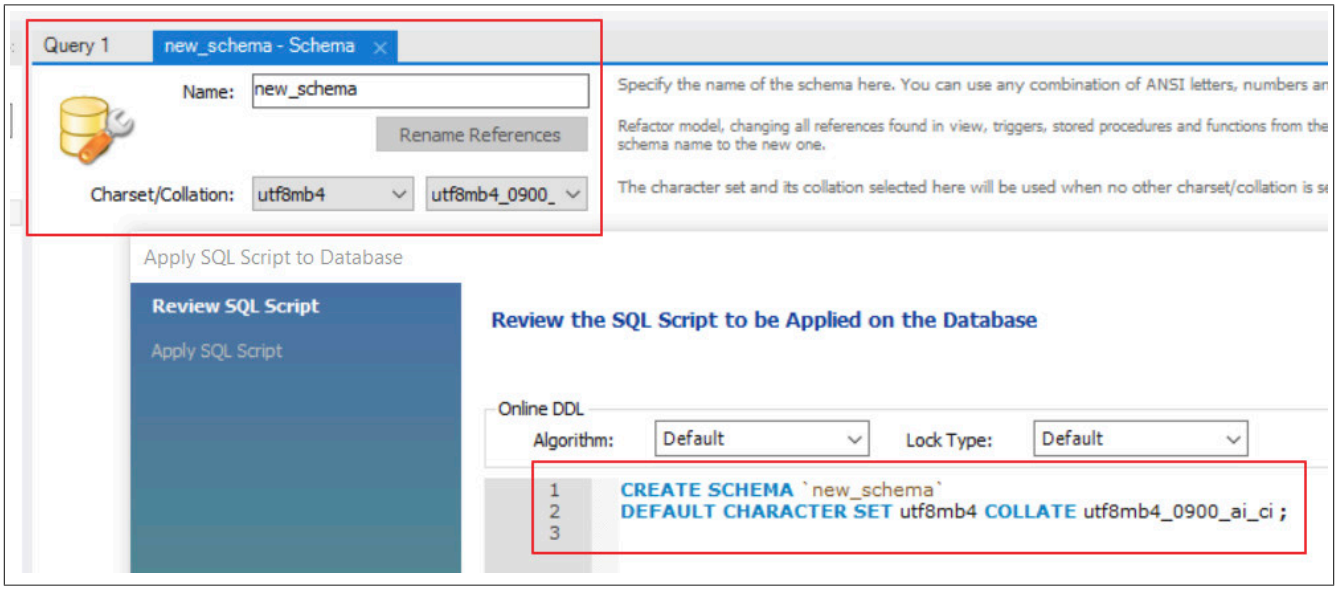
Software	Version	Download
PostgreSQL	10.5	Link
Python	2.7.15 / 32-Bit Version | 3.8.2 / 32-Bit Version	Link
MySQL-Konnektor	8.0 / 32-Bit Version	Link
PostgreSQL psycopg Konnektor	2.7	Link

Unterstützung des Datentypen WSTRING

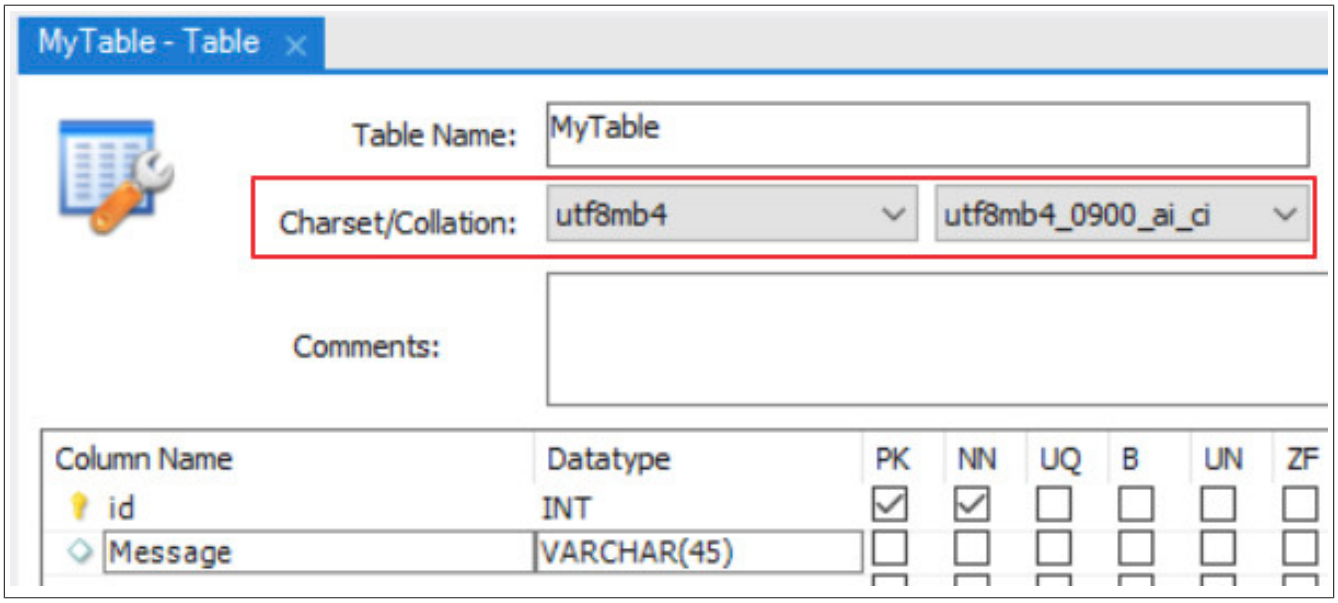
Abhängig davon welches Datenbank-System verwendet wird, muss bei der Verwendung von WSTRING Datentypen folgende Punkte beachtet werden.

MySQL, MariaDB:

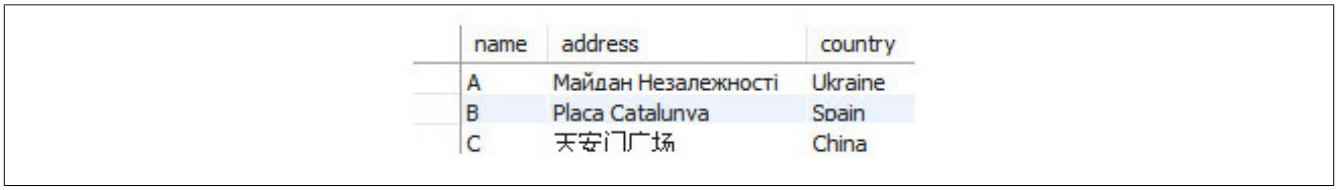
In einer MySQL oder MariaDB Datenbank muss die richtige Kodierung eingestellt sein.
mapp Database verwendet eine UTF8-Kodierung. Für zum Beispiel eine MySQL oder MariaDB Datenbank muss die Kodierung UTF8MB4 verwendet werden. Die Kodierung wird dabei bei der Erstellung der Datenbank angegeben:



Beim Erstellen einer Tabelle muss ebenfalls auf die richtige Kodierung geachtet werden.



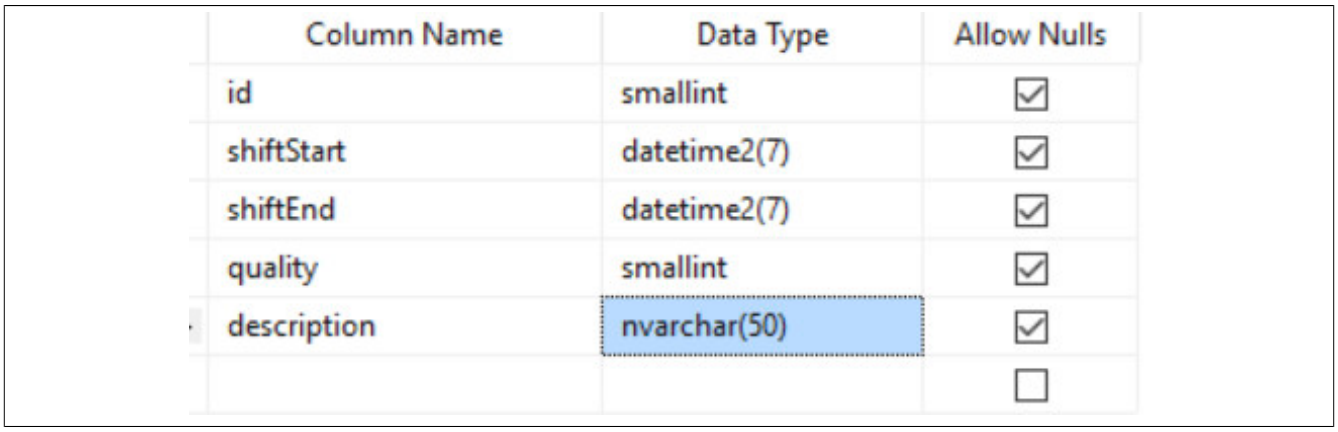
Mit der richtigen Kodierungs-Einstellung in der Datenbank und der Tabelle, können verschiedenste Zeichen in der Datenbank verwendet werden:



	name	address	country
	A	Майдан Незалежності	Ukraine
	B	Placa Catalunya	Spain
	C	天安门广场	China

MS SQL:

mapp Database verwendet eine UTF8-Kodierung. Damit der WSTRING Datentyp bei einer MS SQL Datenbank richtig interpretiert wird, muss bei der Angabe der Spalten der Typ NVARCHAR oder NCHAR verwendet werden:



2 Konfiguration

2.1 MpDatabaseCore Konfiguration

Die hier in der Konfiguration existierenden Passwörter können verschlüsselt werden. Für mehr Information siehe Abschnitt "Passwortverschlüsselung in der Konfiguration" unter Rahmenbedingungen.

Wie die Konfiguration zur Laufzeit angepasst werden kann, wird hier im Abschnitt "Konfiguration zur Laufzeit ändern" erklärt.

Allgemein

In den allgemeinen Einstellungen können folgende Parameter konfiguriert werden:

- **Database name:** Unter "Database name" wird der Name der gewünschten Datenbank angegeben.
- **User name:** Angabe des Benutzernames um sich mit der Datenbank zu verbinden.
- **Password:** Angabe des Passworts um sich mit der Datenbank zu verbinden.
- **IP address:** Definiert die IP Adresse um sich mit der Datenbank verbinden zu können.
- **Port number:** Definiert die Portnummer um sich mit der Datenbank verbinden zu können. Es wird die Port-Nummer des HTTP-Servers angegeben über den die Kommunikation läuft. Für mehr Information siehe hier.
- **DBMS:** Unter "DBMS" wird das Datenbanksystem ausgewählt. Es kann "Open" verwendet werden. Das bedeutet es kann sich mit jedem Datenbanksystem verbunden werden. Für mehr Information siehe Abschnitt Grundprinzip.

Database settings

Einige Datenbanken unterstützen 8 Byte Integer. Diese werden von Automatio Studio aber nicht untersützt. Aus diesem Grund kann, falls ein 8 Byte Integer vorkommt, dieser umgewandelt werden in UDINT, DINT oder LREAL.

Soll keine umwandlung stattfinden, so kann "No conversion" ausgewählt werden. Der Benutzer muss in dem Fall sicherstellen, dass die Werte richtig konvertiert werden. Dabei muss ein Cast durchgeführt werden. Für mehr Information dazu siehe hier.

Automatisches Mapping

Unter "Automatic mapping" kann definiert werden, wie Struktur-Variablen in Automation Studio auf die Spalten in der Datenbank-Tabelle gemappt werden. Es stehen zwei Optionen zur Auswahl:

- **Strict:** Bei "Strict" muss die Reihenfolge und der Name der Variablen innerhalb Struktur exakt mit dem Namen und der Reihenfolge der Spalten innerhalb der Tabelle übereinstimmen. Die Struktur darf keine zusätzlichen Variablen enthalten. Stimmt die Struktur nicht exakt mit der Tabelle überein, wird ein Fehler an MpDatabaseQuery angezeigt. Abhängig von der verwendeten Datenbank, kann in der Kommando-Konsole zusätzliche Information gefunden werden. Für mehr Information dazu, siehe Abschnitt FAQ.
- **Flexible:** Bei "Fexible" kann die Reihenfolge der Variablen innerhalb der Struktur variieren. Es muss lediglich der Name der Variable mit dem Spalten-Namen in der Tabelle übereinstimmen. Stimmt die Spalten-Anzahl und Variablen-Anzahl nicht überein oder wurden nicht alle Spalten abgedeckt, so ist das Verhalten vom verwendeten Datenbank-System abhängig. Für mehr Information siehe hier im Abschnitt "Angabe von Variablen".

Im Abschnitt Abfrage-Typen wird erklärt, bei welchen Abfrage-Typen die Automatische Mapping Funktion verwendet werden kann.

Abfrage

Unter "Queries" können Abfragen erstellt werden. Dabei muss unter "Name" ein eindeutiger Name für die Abfrage angegeben werden. Soll die Anfrage über MpDatabaseQuery ausgeführt werden, so muss am Eingangsparameter "Name" der Name angegeben werden, welcher hier definiert wird.

Über "Type" wird der Abfrage-Typ bestimmt. Folgende Typen sind möglich:

- SELECT
- INSERT
- UPDATE
- CREATE TABLE
- DELETE FROM
- Custom

Für mehr Information zu den einzelnen Abfrage-Typen siehe hier.

Inhalte welche durch SELECT-Abfragen von einer Datenbank angefrage werden, können mithilfe des Database-Widgets in einer mapp View Visualisierung angezeigt werden. Für mehr Information dazu siehe hier im Abschnitt "SELECT".

2.1.1 Abfrage-Typen

> **Abfragen vom Typen SELECT können mithilfe des Database-Widgets ausgeführt und angezeigt werden. Für mehr Information siehe Abschnitt "SELECT".**

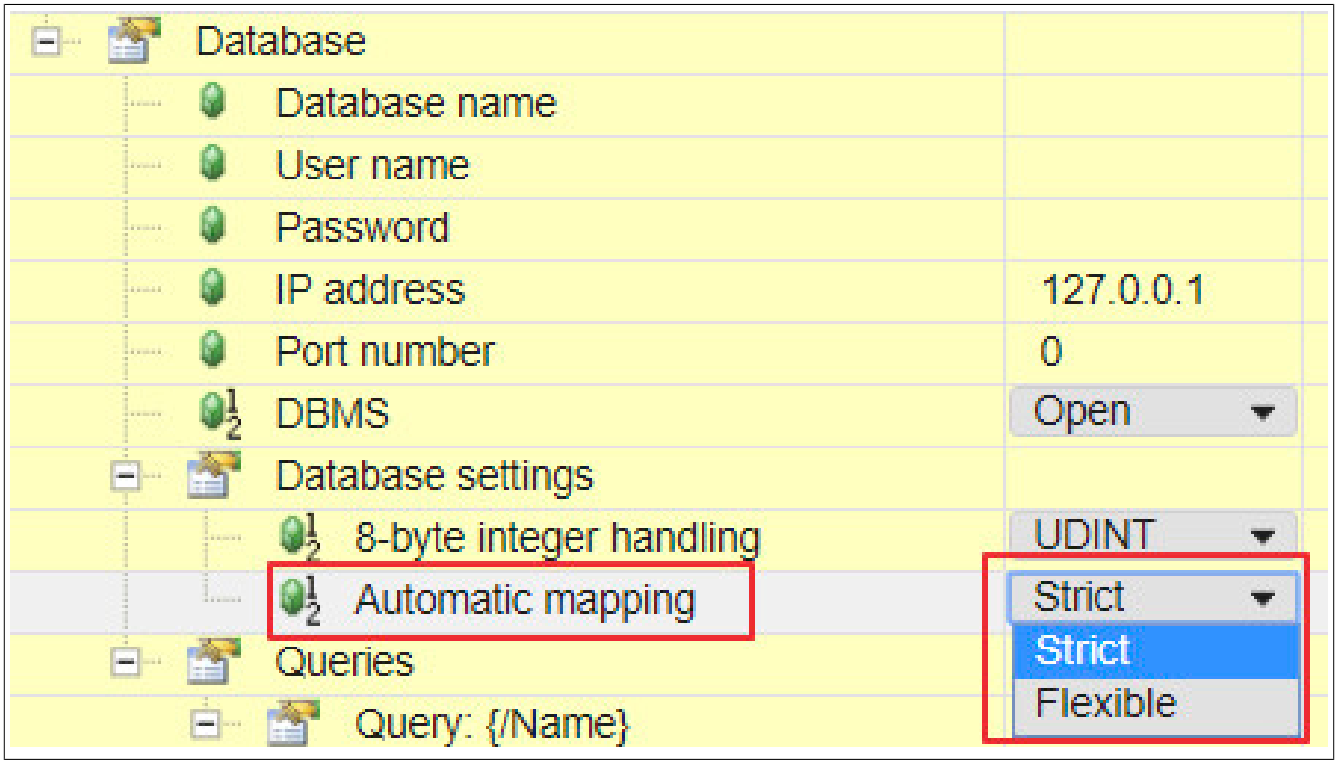
Beispiele für die verschiedenen Abfrage-Typen können hier gefunden werden.

Angabe von Variablen

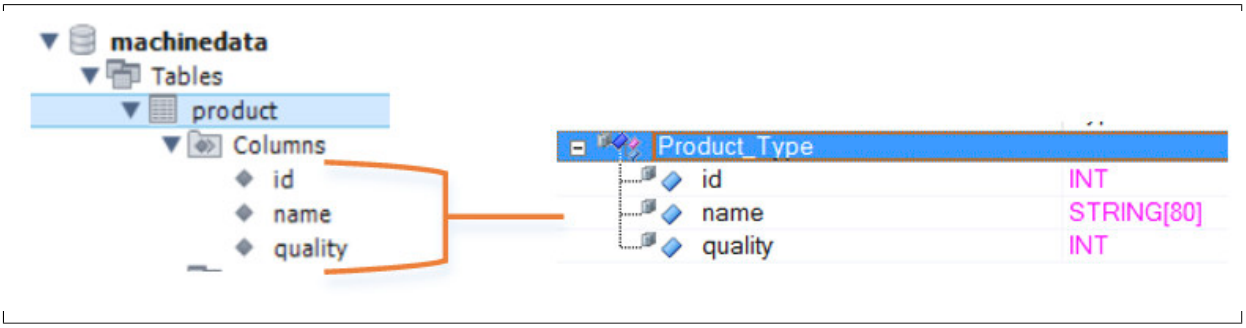
Die Variablen in den verschiedenen Abfrage-Typen können auf zwei verschiedene Weisen angegeben werden.

Automatic

Die Option "Automatic" ist abhängig davon, welche Einstellung unter "Automatic mapping" vorgenommen wurde:



- **Strict:** Durch "Automatic" und "Strict" verbindet sich die Variable automatisch mit den Spalten der Tabelle. Es muss eine Variable angegeben werden, welche die exakte Struktur und Variablen-Namen besitzen wie in der Datenbank-Tabelle. Im Bild unten existiert die Tabelle "product", welche die Spalten "id", "name" und "quality" besitzt. In Automation Studio wird eine Variable erstellt, wie zum Beispiel "SetData" welche vom Typen "Product_Type" ist:



  Stimmen die Variablen-Namen nicht mit den Spalten-Namen überein oder wurden zu wenig Variablen angegeben, so wird der Fehler "Fehler in der Abfrage" am Funktionsbaustein MpDatabaseQuery ausgelöst. Im Logger wird zusätzlich der Fehler "Falsche PV-Angabe" angezeigt.
- **Fexible:** Bei "Automatic" und "Fexible" kann die Reihenfolge der Variablen innerhalb der Struktur variieren. Es muss lediglich der Name der Variable mit dem Spalten-Namen in der Tabelle übereinstimmen. Stimmt die Spalten-Anzahl ("SA") und Variablen-Anzahl ("VA") nicht überein oder wurden nicht alle Spalten abgedeckt, so ist das Verhalten vom verwendeten Datenbank-System und vom Abfrage-Typen abhängig.

Mapping-Fall	SELECT	INSERT INTO
VA und SA sind identisch Spalten-Namen sind abgedeckt Variablen-Namen und Spalten-Namen in gleicher Reihenfolge	✓	✓
VA und SA sind unterschiedlich Spalten-Namen sind abgedeckt Variablen-Namen und Spalten-Namen in gleicher Reihenfolge	✓	✗
VA und SA sind unterschiedlich Spalten-Namen sind nicht abgedeckt. Variablen-Namen und Spalten-Namen in geleicher Reihenfolge	✓	⚠
VA und SA sind identisch Spalten-Namen sind abgedeckt Variablen-Namen und Spalten-Namen nicht in gleicher Reihenfolge	✓	✓
VA und SA sind unterschiedlich Spalten-Namen sind abgedeckt Variablen-Namen und Spalten-Namen nicht in gleicher Reihenfolge	✓	✗
VA und SA sind unterschiedlich Spalten-Namen sind nicht abgedeckt Variablen-Namen und Spalten-Namen sind nicht in gleicher Reihenfolge	✓	⚠

- ✓: Abfrage kann ausgeführt werden.
- ✗: Abfrage kann nicht ausgeführt werden. Es wird ein Fehler ausgelöst. Für mehr Information siehe "Meldungen in der Kommando-Konsole" im Abschnitt FAQ.
- ⚠: Verhalten hängt von der Einstellung der verwendeten Datenbank ab. Dies kann in der Dokumentation der jeweiligen Datenbank gefunden werden. Da nicht alle Spalten-Namen abgedeckt sind, wird entweder ein in der Datenbank festgelegter Standardwert eingefügt oder ein Fehler ausgelöst. Für mehr Information siehe "Meldungen in der Kommando-Konsole" im Abschnitt FAQ.

Single Columns

Bei "Single columns" werden die gewünschten Spalten einzeln angegeben. Der gewünschte Spalten-Name wird an "Column" definiert. Damit die Information abgespeichert, gelöscht, aktualisiert etc. werden kann, muss eine PV angegeben werden. Die Namen der PV und der Spalten-Name in der Datenbank müssen dabei nicht übereinstimmen!

Angabe von PV

Wird eine Variable von einem einfachen Datentypen angegeben, so erfolgt dies folgendermaßen: "::Taskname:PV-Name"

Für folgende IEC-Datentypen sollte folgender SQL-, MSSQL-, oder PostgreSQL-Datentyp verwendet werden:

IEC-Datentyp	MySQL-/MariaDB-Datentyp
BOOL	TINYINT(1)
SINT	TINYINT(4)
INT	SMALLINT(6)
DINT	INT(11)
USINT	TINYINT(3) (unsigned)
UINT	SMALLINT(5) (unsigned)
UDINT	INT(10) (unsigned)
STRING	VARCHAR(81)
WSTRING	VARCHAR
TIME	TIME
LREAL	DOUBLE
REAL	FLOAT
DATE	DATE
DATE_AND_TIME	DATETIME
DTStructure	DATETIME(6)

Bei der Verwendung vom Datentypen WSTRING sollten die Punkte beachtet werden, welche im Abschnitt Rahmenbedingungen erklärt werden!

Negative TIME-Werte können in MS SQL und PostgreSQL nicht verwendet werden!

Beim MSSQL-Datentypen BIGINT handelt es sich um einen 8-Byte Datentypen. Dieser kann von Automation Studio nicht automatisch verarbeitet werden. Es muss eine Konvertierung nach LREAL, DINT oder UDINT vorgenommen werden. Dies kann in der MpDatabaseCore Konfiguration unter den allgemeinen Einstellungen ("Database setting") vorgenommen werden.

Angabe von Array-PVs

Wird eine Array-PV angegeben, so erfolgt die Angabe folgendermaßen: "::Taskname:MyArrayPV[]"

Die Klammern [] müssen dabei selbst hinzugefügt werden.

Angabe von PV-Snippets

In Abfragen gibt es auch die Möglichkeit PVs in Form von Snippets anzugeben. Das ist speziell in Kombination mit "Custom" Anfragen relevant. Diese werden in der Konfiguration mit "{&pv[Taskname:PVName]}" angegeben. Das Snippet muss immer in geschwungenen Klammern ("{}") angegeben werden. Das "&pv" steht dafür, dass die Angabe einer Prozessvariable erfolgt. In den eckigen Klammern ("[]") erfolgt die eigentliche Angabe der PV.

Für Beispiele wie Variablen in Abfragen angegeben werden können, siehe Abschnitt Abfrage-Beispiele.

SELECT

Mithilfe von "SELECT" kann die Information von einer Tabelle/Spalte in eine PV kopiert werden. Dabei wird unter "FROM" definiert von welcher Tabelle Information abgefragt werden soll. Von welcher Spalte die Information kommen soll, wird unter "Columns" bestimmt. Dabei gibt es zwei Abfrage Möglichkeiten "Automatic" und "Single columns". Diese sind oben unter "Angabe von Variablen" beschrieben.

Zusätzlich dazu können noch Filterkriterien unter "WHERE" angegeben werden. Über "Column" wird der Name der gewünschten Spalte angegeben. Unter "Value" kann ein zu vergleichender Wert bestimmt werden. Wie der Wert mit dem Spaltennamen verglichen werden soll, wird über "Operator" definiert.

Wird als Vergleich ("Operator") "LIKE" verwendet, so kann nach ähnlichen Einträgen gesucht werden. Dabei muss das Platzhaltersymbol "*" verwendet werden. Wird als Wert ("Value"), zum Beispiel "*Test*" verwendet, können zum Beispiel nach den Einträgen "Test01", "MyTest" oder "MyTest01" gesucht werden.

In manchen Datenbanken wird auch das Platzhaltersymbol (Wildcard) "%" verwendet.

Es können verschiedene Filteroptionen angegeben werden. Diese werden über "Connect" miteinander verbunden.

Über "Order by" kann der abgefragte Inhalt abhängig von einer gewünschten Spalte auf-, oder absteigend sortiert werden. Dabei muss der Spaltenname angegeben werden, welcher in der Datenbank-Tabelle verwendet wird.

Wie "SELECT" verwendet werden kann, ist auch im Anwendungsfall Produktionsdaten in Datenbank einfügen und abfragen erklärt.

Anbindung zu mapp View (Database-Widget)

Die in der Konfiguration angegebenen SELECT-Abfragen können mithilfe des Database-Widgets ausgeführt und angezeigt werden. Wie das Widget dabei projektiert werden muss, ist im Abschnitt Beispielhafte Projektierung bzw. Arbeiten mit dem Database-Widget erklärt. Dabei wird über den Parameter "Destination" in der Konfiguration bestimmt, ob der Inhalt der Abfrage nur am Widget oder auch an der angegeben Prozess Variable angezeigt werden soll.

Ist "Destination = Widget" so kann die Abfrage vom Widget ausgeführt und der Inhalt der Abfrage wird auch nur am Database-Widgets angezeigt. Bei "Destination = Process variable" kann die Abfrage vom Widget ausgeführt und der zurückgelieferte Inhalt der Abfrage wird in der angegebenen Prozess Variable sowie am Database-Widgets angezeigt. Für weitere Information siehe hier.

INSERT

Mithilfe von "INSERT" kann in einer Datenbank Information hinzugefügt werden. Dabei muss unter "INTO" die gewünschte Datenbank-Tabelle angegeben werden. "Columns" bestimmt wie die Information hinzugefügt wird. Dies wird oben im Abschnitt "Angabe von Variablen" erklärt.

Wie "INSERT" verwendet werden kann, ist auch im Anwendungsfall Produktionsdaten in Datenbank einfügen und abfragen erklärt.

UPDATE

Durch "UPDATE" können Informationen in der Tabelle aktualisiert werden. Dabei wird der gewünschte Tabellen-Name unter "Table Name" angegeben. Anschließend kann über "Columns" bestimmt werden wie die Information aktualisiert werden soll. Dabei gibt es zwei Möglichkeiten "Automatic" und "Single columns". Diese sind oben unter "Angabe von Variablen" beschrieben.

Zusätzlich dazu können noch Filterkriterien unter "WHERE" angegeben werden. Über "Column" wird der Name der gewünschten Spalte angegeben. Unter "Value" kann ein zu vergleichender Wert bestimmt werden. Wie der Wert mit dem Spaltennamen verglichen werden soll, wird über "Operator" definiert.

Wird als Vergleich ("Operator") "LIKE" verwendet, so kann nach ähnlichen Einträgen gesucht werden. Dabei muss das Platzhaltersymbol "*" verwendet werden. Wird als Wert ("Value"), zum Beispiel "*Test*" verwendet, können zum Beispiel nach den Einträgen "Test01", "MyTest" oder "MyTest01" gesucht werden.

In manchen Datenbanken wird auch das Platzhaltersymbol (Wildcard) "%" verwendet.

Es können verschiedene Filteroptionen angegeben werden. Diese werden über "Connect" miteinander verbunden.

CREATE TABLE

Mithilfe von "CREATE TABLE" kann eine neue Tabelle in der Datenbank erstellt werden. Der Name der Tabelle wird dabei über "Table Name" definiert.

Über "Columns" wird definiert wie die Spalten in der Tabelle erstellt werden sollen. Dabei gibt es zwei Möglichkeiten "Automatic" und "Single columns". Diese sind oben unter "Angabe von Variablen" beschrieben.

Die Datentypen der angegebenen Struktur/Variable bestimmt dabei die Datentypen in der Datenbank-Tabelle!

DELETE FROM

Durch "DELETE FROM" kann in einer Tabelle, welche über "Table Name" angegeben werden muss, eine oder mehrere Spalten gelöscht werden.

Welche Einträge gelöscht werden sollen, wird im Abschnitt "WHERE" definiert. Dabei können verschiedene Filteroptionen definiert werden.

Über "Column" wird der Name der gewünschten Spalte angegeben. Unter "Value" kann ein zu vergleichender Wert bestimmt werden. Wie der Wert mit dem Spaltennamen verglichen werden soll, wird über "Operator" bestimmt.

Wird als Vergleich ("Operator") "LIKE" verwendet, so kann nach ähnlichen Einträgen gesucht werden. Dabei muss das Platzhaltersymbol "*" verwendet werden. Wird als Wert ("Value"), zum Beispiel "*Test*" verwendet, können zum Beispiel nach den Einträgen "Test01", "MyTest" oder "MyTest01" gesucht werden.

In manchen Datenbanken wird auch das Platzhaltersymbol (Wildcard) "%" verwendet.

Es können verschiedene Filteroptionen angegeben werden. Diese werden über "Connect" miteinander verbunden.

Custom

Unter "Custom" kann eine selbst definierte SQL-Abfrage erstellt werden. Der SQL-Code wird dabei unter "Query" eingefügt. Mehr Information zum Thema SQL-Abfragen erstellen, kann zum Beispiel hier gefunden werden.

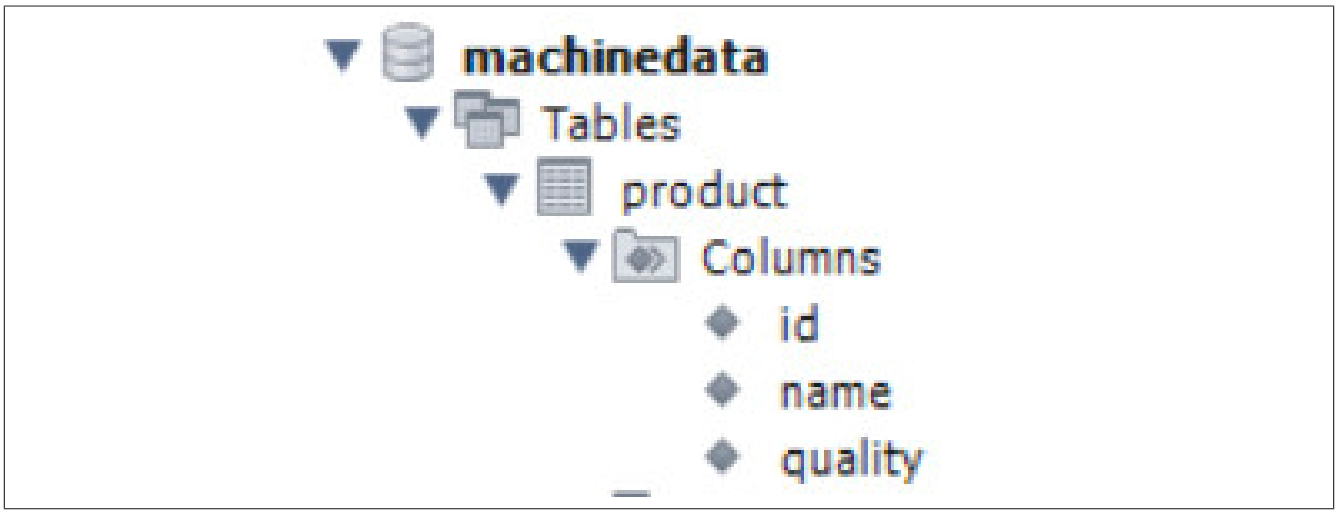
Wie eine benutzerdefinierte Abfrage erstellt werden kann, ist im Anwendungsfall Benutzerdefinierte Abfrage erstellen erklärt.

Die Gültigkeit der Abfrage kann nicht in mapp Database geprüft werden. Falls bei der Ausführung der Abfrage Fehler entstehen, kann zusätzliche Information im Kommando-Fenster des Skripts gefunden werden. Für mehr Information siehe hier.

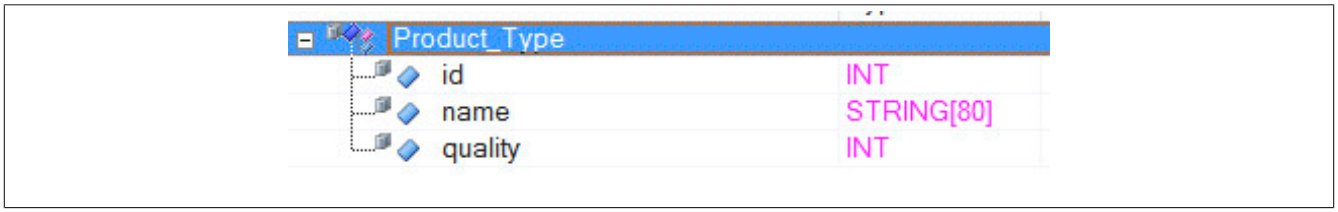
2.1.1.1 Abfrage-Beispiele

In diesem Abschnitt werden Beispiele für verschiedenste Abfrage-Typen aufgelistet.

Für die Beispiele wurde das Datenbanksystem MySQL verwendet. Es existiert die Datenbank "machinedata" welche die Tabelle "product" besitzt. In der Tabelle existieren die Spalten "id", "name" und "quality". Diese Datenbank wird in den unten folgenden Beispielen verwendet.



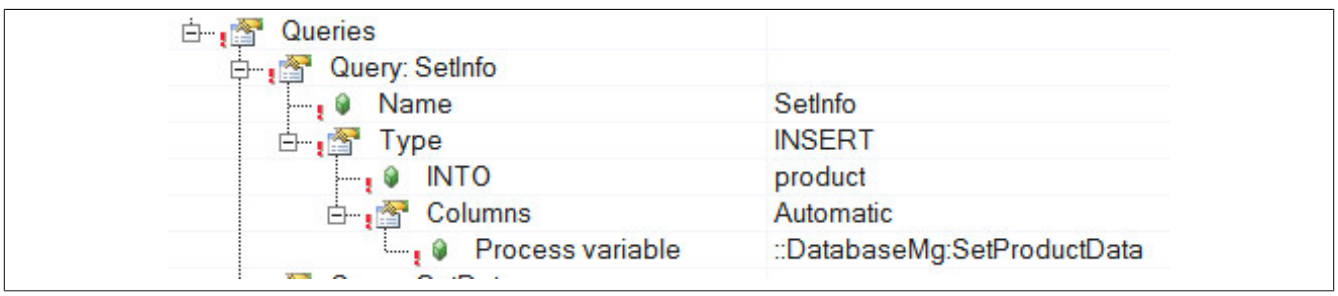
In Automation Studio existiert der benutzerdefinierte Datentyp "Product_Type", welcher die gleiche Struktur wie die Tabelle besitzt:



Neue Daten in der Tabelle hinzufügen ("INSERT")

Um Daten in der Tabelle hinzuzufügen, wird der Abrage Typ "INSERT" verwendet. Das Einfügen der Daten in der Tabelle kann dabei automatisch für alle Spalten erfolgen oder einzeln. Es existiert die Variable "SetProductData" welche vom Datentypen "Product_Type" ist.

Automatisches einfügen



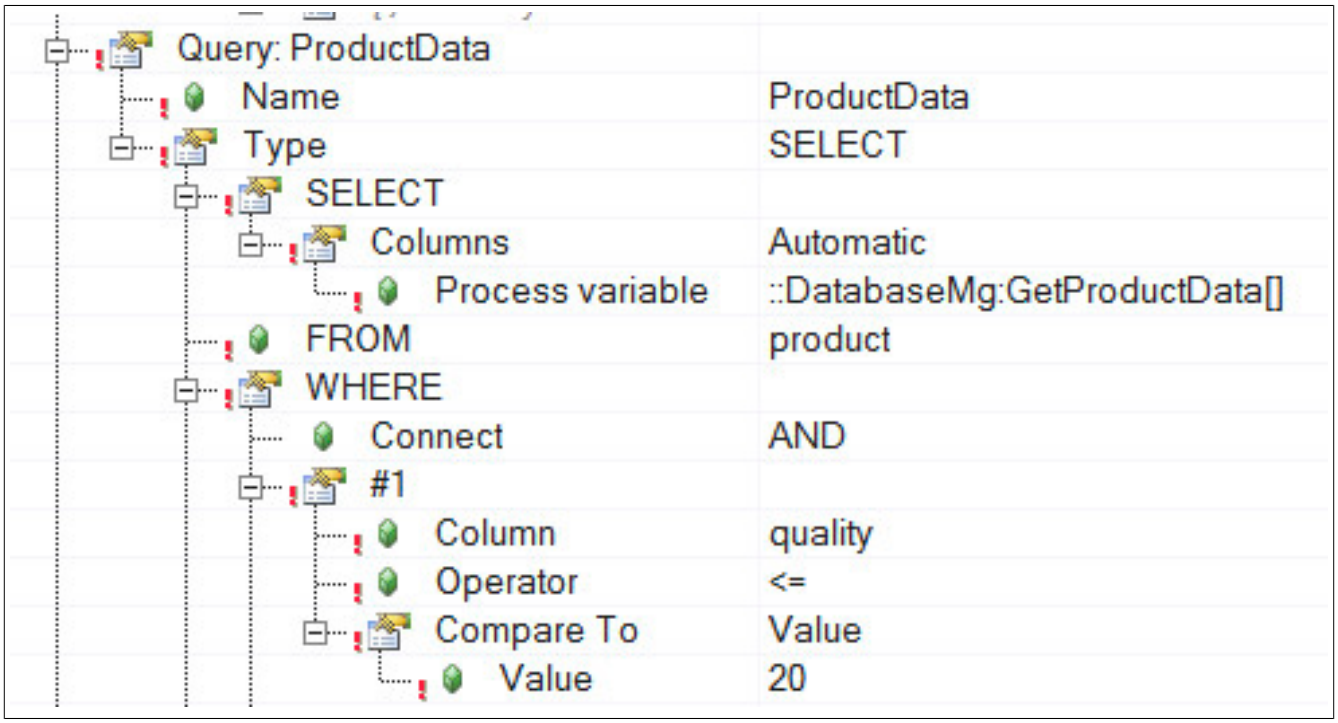
Einzelnes einfügen

Daten aus der Tabelle abfragen ("SELECT")

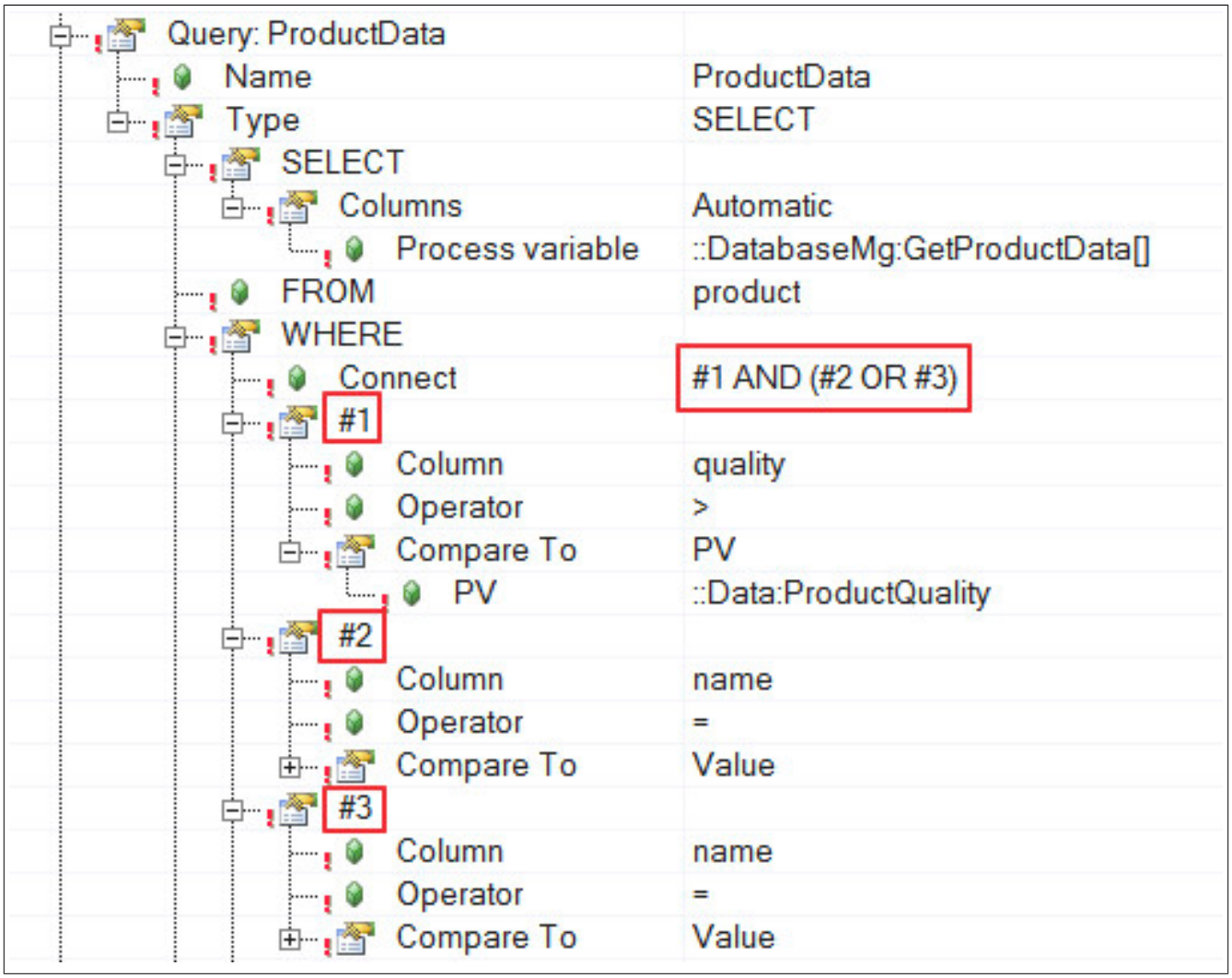
Um Daten aus der Tabelle abzufragen und zu analysieren, wird der Abfrage-Typ "SELECT" verwendet.

Es soll die Qualität der Produkte geprüft werden. Dafür sollen alle schlechten Produkte (Qualität <= 20) angezeigt werden.

Dies kann folgendermaßen abgefragt werden:



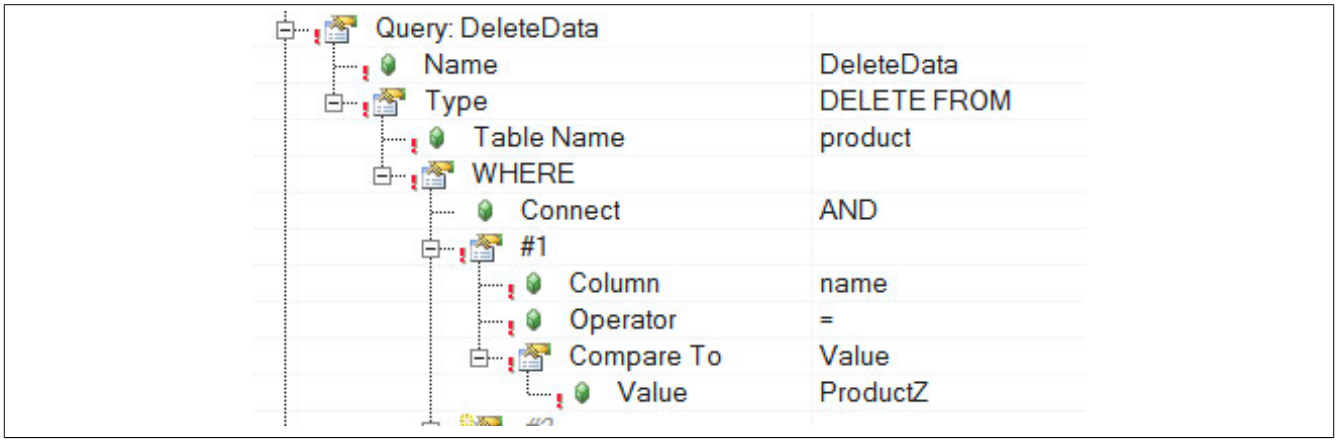
Es können auch zum Beispiel mehrere Abfragen miteinander verbunden werden. In diesem Beispiel wird nach der Qualität (#1) und nach dem Produktnamen "RED" (#2) oder "BLUE" (#3) abgefragt:



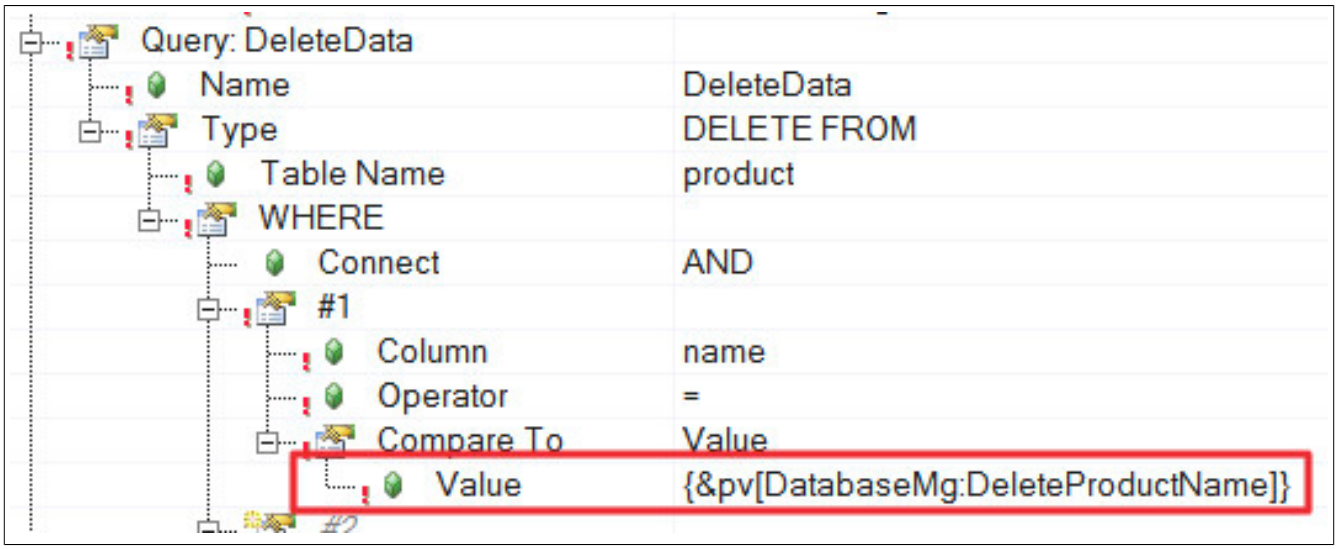
Daten aus der Tabelle löschen ("DELETE")

Um Daten aus der Tabelle zu löschen, wird der Abfrage-Typ "DELETE" verwendet.

Da das Produkt mit dem Namen "ProductZ" nicht mehr produziert wird, soll es aus der Datenbank entfernt werden. Dies kann folgendermaßen funktionieren.



Um verschiedene Produkt-Namen zu verwenden, kann ein PV-Snippet verwendet werden. Dadurch kann sich der Produkt-Name zur Laufzeit ändern:



Daten in der Tabelle aktualisieren ("UPDATE")

Um Daten aus der Tabelle zu aktualisieren, wird der Abfrage-Typ "UPDATE" verwendet.

Da es Anpassungen bei der Produzierung des Produkts "ProductA" gab, ist die Qualität des Produkts gestiegen. Aus diesem Grund soll die Information in der Datenbank angepasst werden. Dies kann folgendermaßen funktionieren:

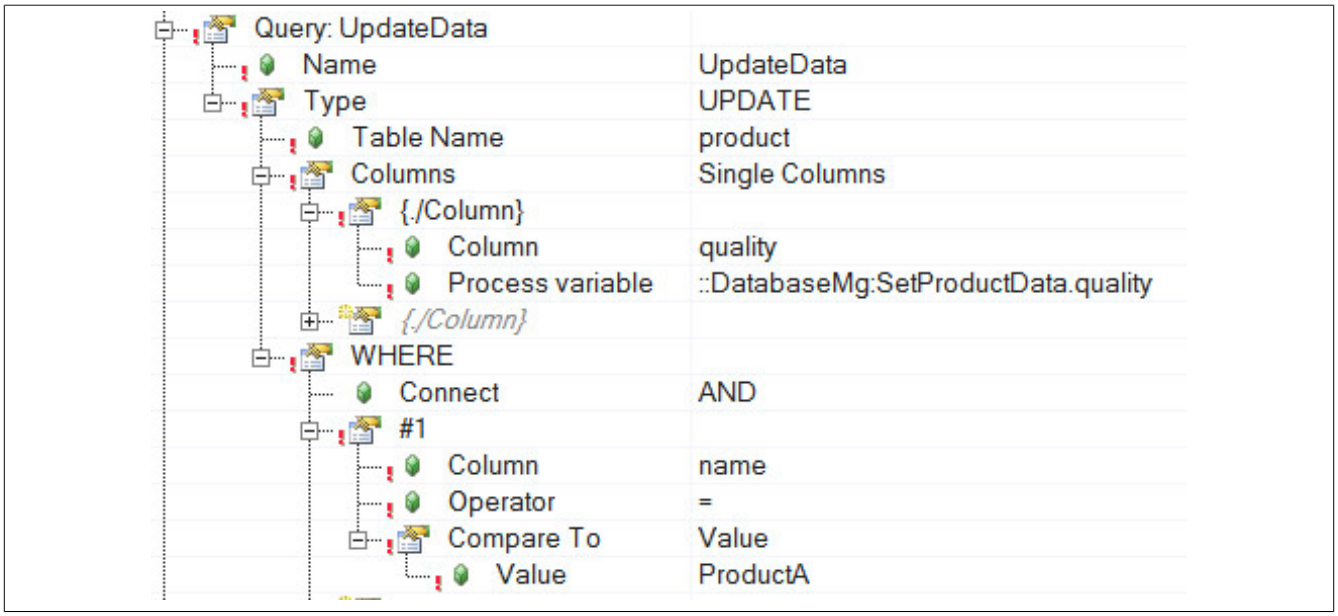
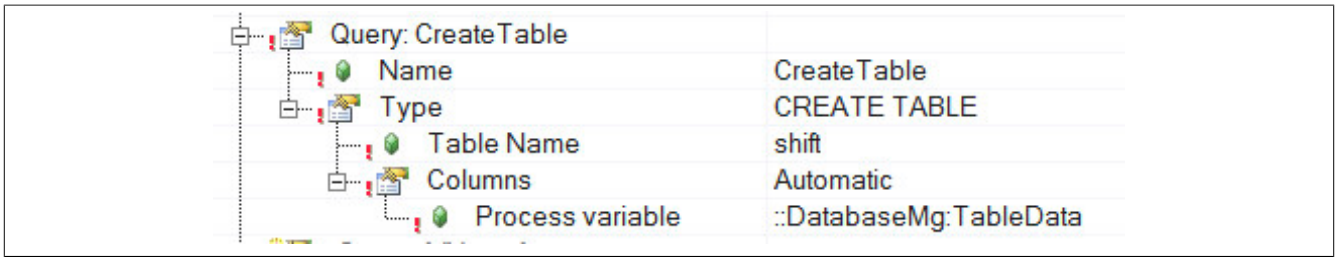


Tabelle erstellen ("CREATE")

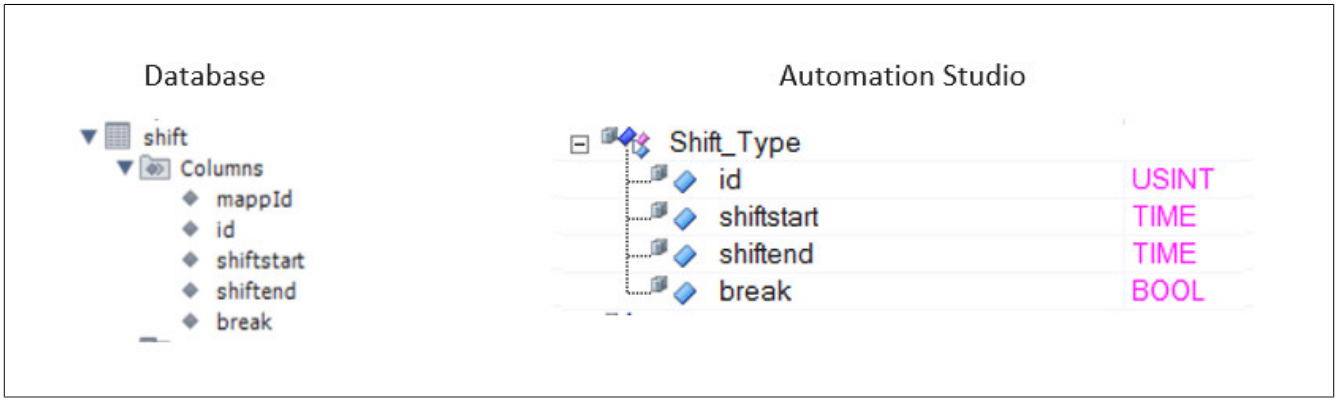
Um eine neue Tabelle in der Datenbank zu erstellen, wird der Abfrage-Typ "CREATE" verwendet.

In der Datenbank wird eine neue Tabelle "shifts" erstellt, welche Informationen über die Schicht enthalten soll.



In dem Beispiel wurde eine PV mit dem Namen "TableData" angegeben, welche vom Datentypen "Shift_Type" ist. Es wird in der Datenbank eine Tabelle erstellt, welche als Spalten genau die Variablen verwendet welche im Datentyp "Shift_Type" zu finden sind. Zusätzlich wird noch eine Spalte "mappId" angelegt. Wenn mit Datenbanken gearbeitet wird, wird empfohlen die erste Spalte als "Primary Key" zu markieren. Mehr Information dazu kann hier gefunden werden. Da der "Primary Key" in der Konfiguration von mapp Database nicht festgelegt werden kann, wird dieser automatisch in der Datenbank erstellt.

Die weiteren Spalten haben dabei den gleichen Datentypen wie die Variablen in Automation Studio.



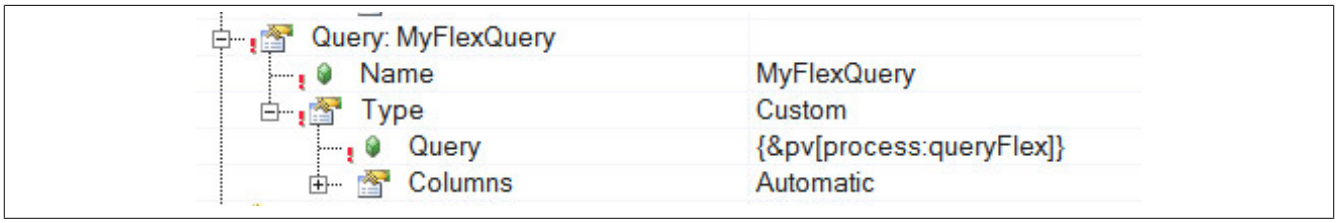
Benutzerdefinierte Abfrage erstellen ("Custom")

Durch verwenden des Abfrage-Typen "Custom" kann eine benutzerdefinierte Abfrage erstellt werden.

> **Die Gültigkeit der Abfrage kann nicht in mapp Database geprüft werden. Falls bei der Ausführung der Abfrage Fehler entstehen, kann zusätzliche Information im Kommando-Fenster des Skripts gefunden werden. Für mehr Information siehe hier.**

Über "Custom" können alle beliebigen Abfragen erstellt werden. Dabei können auch PV-Snippets verwendet werden:

Oder es besteht zum Beispiel auch die Möglichkeit, die Abfrage selbst zur Laufzeit zu ändern. Dabei muss die gesamte Abfrage in der Konfiguration ein PV-Snippet sein. Im folgendem Beispiel existiert im Task "process" die Variable "queryFlex". Über diese Variable kann der Inhalt der Abfrage bestimmt werden:



3 Anwendungsfälle

3.1 Anwendungsfall 1: Produktionsdaten in Datenbank einfügen und abfragen

Anforderung

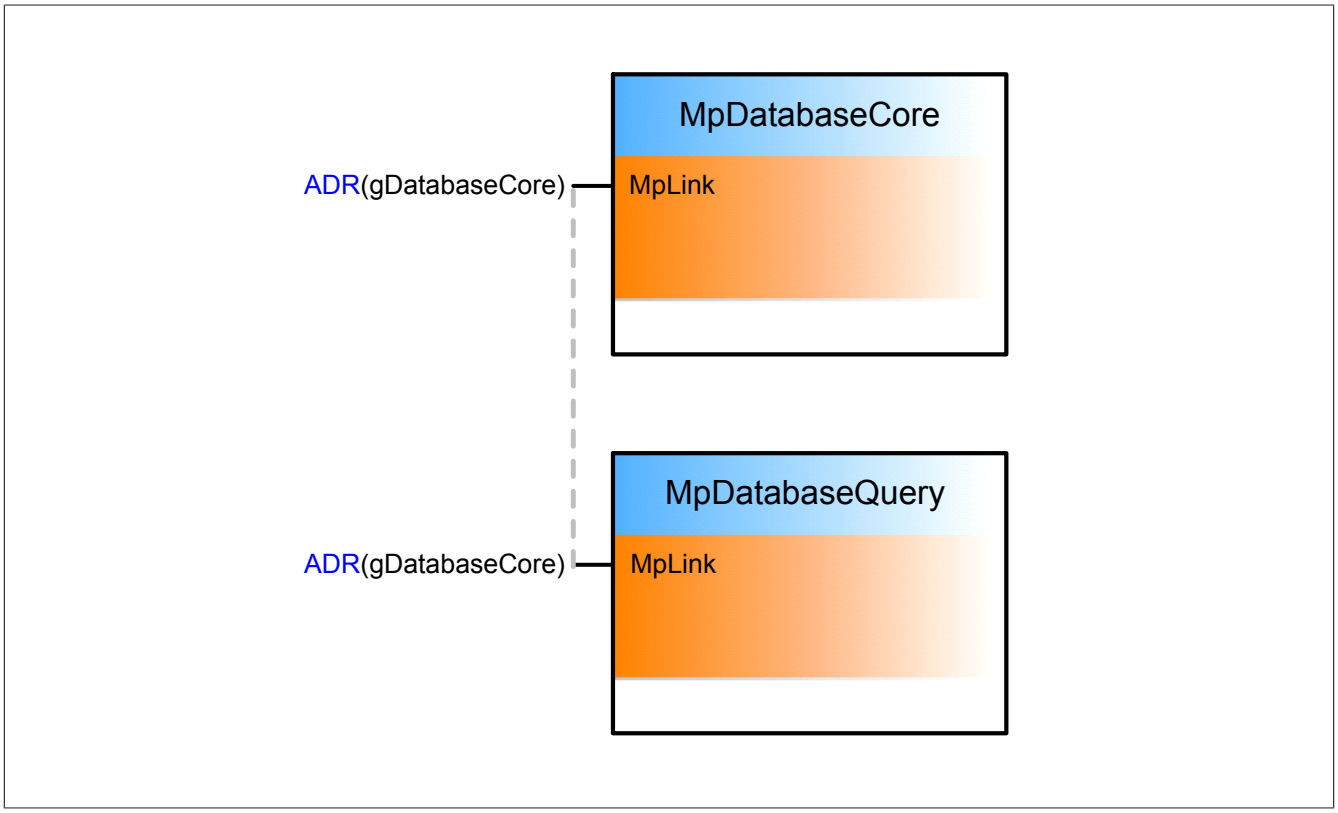
Es sollen statistische Informationen über produzierte Produkte in einer MySQL Datenbank erfasst werden. Die Daten in der Datenbank sollen anschließend weiterverwendet werden können, um verschiedene Analysen und Berechnungen durchzuführen.

Lösung

Komponentenliste

- ***MpDatabaseCore*** (Eigener MpLink): Stellt Verbindung zu Datenbank her
- ***MpDatabaseQuery*** (Eigener MpLink): Startet angegebene Abfragen

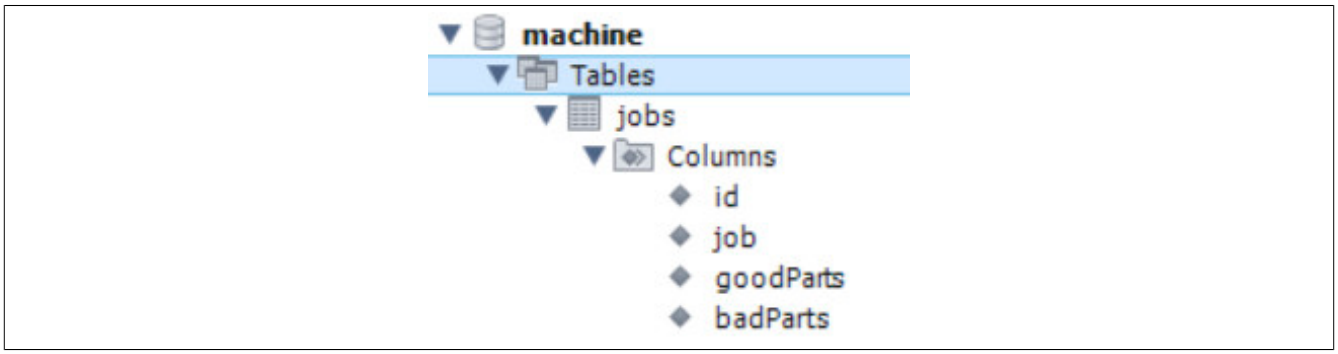
Verbindungsschema



Datenbank

Es wird das Datenbanksystem MySQL verwendet. Es wird die Datenbank "machine" angelegt.

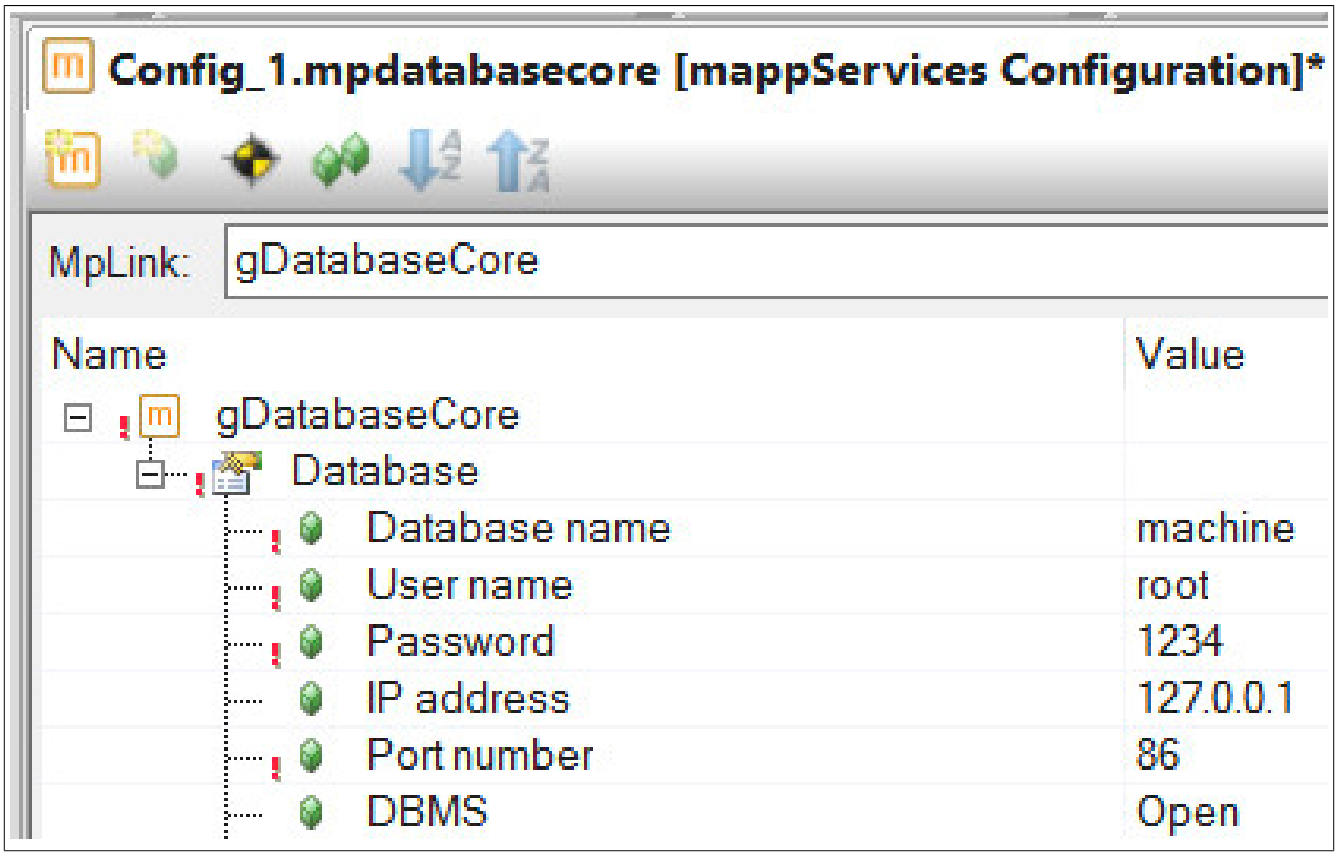
In der Datenbank gibt es eine Tabelle "jobs" welche die Spalten "id", "job", "goodParts" und "badParts" besitzt. In der Tabelle sollen Produktinformationen abgespeichert und angefordert werden können.



Konfiguration

Um eine Verbindung mit einer Datenbank herzustellen, wird die Konfiguration MpDatabaseCore eingefügt. Es müssen die Verbindungs-Parameter angegeben werden.

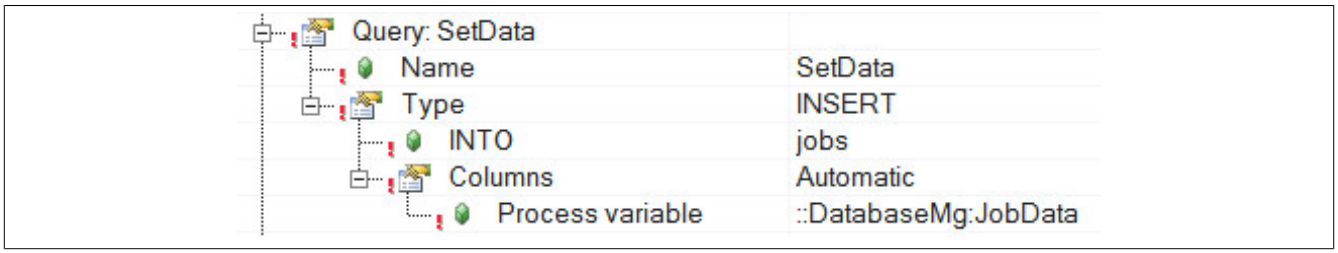
Es soll eine Verbindung zur Datenbank "machine" hergestellt werden. Beim konfigurieren der Datenbank wurden Benutzername, Passwort, IP Adresse und Port Nummer festgelegt. Diese müssen in der Konfiguration angegeben werden.



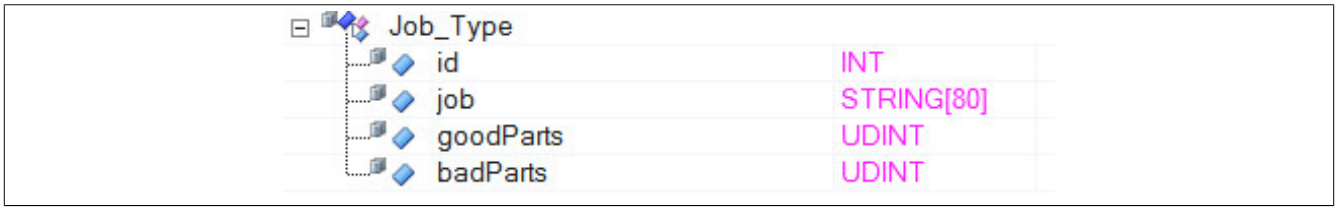
Anschließend müssen die Abfragen ("Queries") konfiguriert werden. Eine Abfrage wird dazu verwendet Information in die Tabelle "jobs" zu speichern, eine wird dazu verwendet die gesamte Information der Tabelle auszulesen.

Abfrage um Daten in der Tabelle hinzuzufügen

Es wird die Abfrage "SetData" vom Typen "INSERT" erstellt. Mithilfe von "INSERT" können Daten in die angegebene Datenbank-Tabelle hinzugefügt werden.



Die angegebene Variable "JobData" ist vom Datentypen "Job_Type". Dieser Datentyp enthält die gleichen Variablen wie die in der Datenbank-Tabelle. Wichtig hierbei ist, dass die Datentypen auch identisch sein müssen.



Abfrage um Daten in der Tabelle anzufordern

Es wird die Abfrage "GetData" vom Typen "SELECT" erstellt. Mithilfe von "SELECT" können Daten aus der Datenbank-Tabelle angefragt werden.

Die angegebene Variable "GetJobData" ist ein Array vom Datentypen "Job_Type". In dieser Variable werden die angeforderten Produktionsinformationen angezeigt.

Mehr Information zu den einzelnen Abfrage-Typen kann hier gefunden werden.

Skript

Bevor sich mapp Database mit der Datenbank verbinden kann, muss ein Skript aktiviert werden. Dieses ist dafür verantwortlich, dass Daten zwischen der Datenbank und mapp Database ausgetauscht werden können. Das Skript kann im Abschnitt Server installieren heruntergeladen werden. Dort kann auch zusätzliche Information zu dem Skript gefunden werden.

Verwendung der mapp Komponenten

Es werden die Funktionsbausteine **MpDatabaseCore** und **MpDatabaseQuery**, eingefügt. Die Funktionsbausteine werden wie im Punkt Verbindungschema miteinander verbunden.

Anschließend werden die Funktionsbausteine parametriert.

Die konfigurierten Abfragen können mit **MpDatabaseQuery** gestartet werden. Dabei muss der Abfrage-Name, welcher in der Konfiguration definiert wurde, am Eingangsparameter "Name" von **MpDatabaseQuery** angegeben werden.

Durch "Connect = `TRUE`" an **MpDatabaseCore** wird die Verbindung zur Datenbank hergestellt. Mithilfe von "Execute = `TRUE`" an **MpDatabaseQuery** wird die Abfrage gestartet.

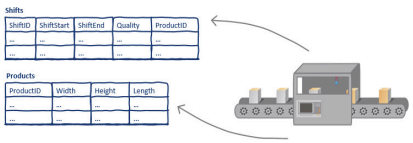
Zuerst kann mit der Abfrage "SetData" die Produktionformation in der Datenbank-Tabelle abgespeichert werden. Anschließend kann mit der Abfrage "GetData" die Information aus der Tabelle angefordert werden. Die Information landet in der Prozessvariable "GetJobData". Anschließend können weitere Analysen mit den erhaltenen Daten durchgeführt werden.

3.2 Anwendungsfall 2: Benutzerdefinierte Abfrage erstellen

Anforderung

Eine Maschine speichert Produktionsdaten in einer MySQL Datenbank ab. Die Produktionsdaten teilen sich in zwei Bereiche auf, den Schicht-Daten und den Produkt-Daten.

Diese Daten werden in jeweils 2 unterschiedlichen Tabellen in der MySQL Datenbank abgespeichert.



In den Schichtdaten wird die Start-, und Endzeit der Schicht sowie die Durchnittsqualität der produzierten Produkte abgespeichert. Welche Art von Produkt erstellt wird, wird über die Produkt-ID festgelegt.

In den Produktdaten ist pro Produkt-ID hinterlägt welche Länge, Höhe und Breite das Produkt besitzt.

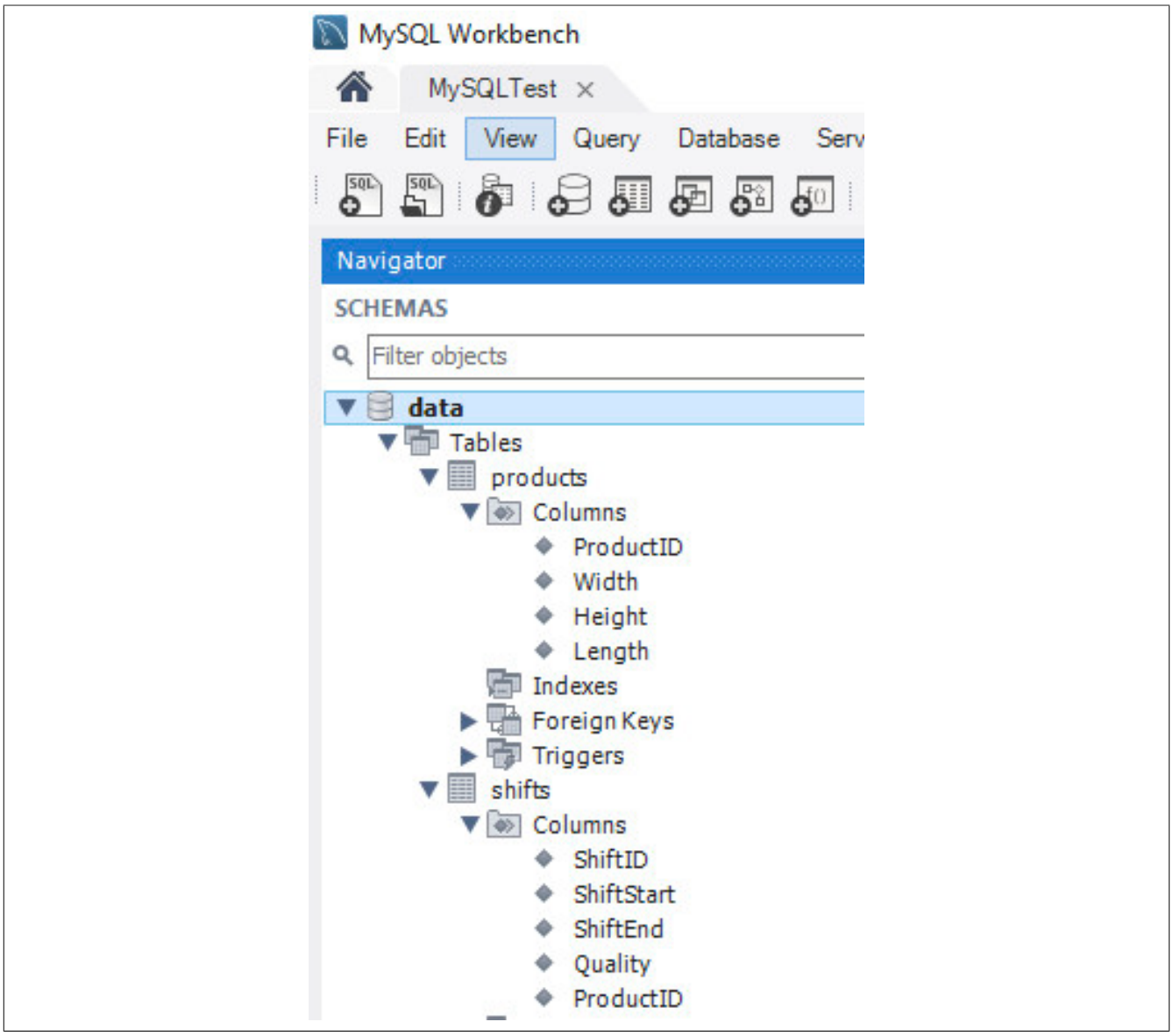
Aufgrund von Qualitätsprüfungen soll es möglich sein, über eine Abfrage Information aus den Schicht-Daten sowie aus den Produkt-Daten zu bekommen. Es soll, abhängig von der Produkt-ID, die Qualität und die Länge des Produkts sowie der Startzeitpunkt der Schicht herausgefunden werden. Dadurch soll überprüft werden, ob die Länge des Produkts die Qualität beeinflusst.

Datenbank

Es wird das Datenbanksystem MySQL verwendet. Es existieren die Tabellen "shifts" und "products".

Die Tabelle "shifts" enthält den Start-, und Endzeitpunkt der Schicht sowie die Durchschnitts-Qualität und ID des Produktes welches in dieser Zeit produziert wurde.

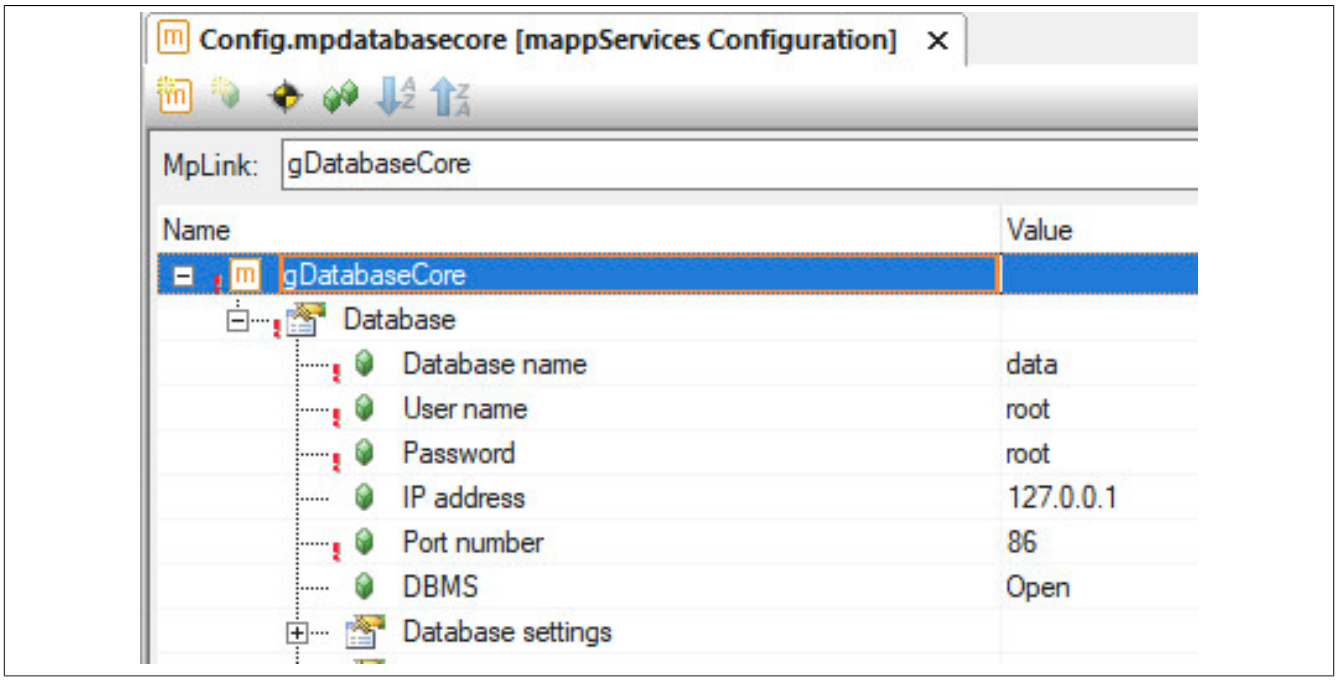
Die Tabelle "product" enthält die Produktinformation. Diese besteht aus der ID des Produkts sowie die Höhe, Länge und Breite davon.



Konfiguration

Um eine Verbindung mit einer Datenbank herzustellen, wird die MpDatabaseCore Konfiguration eingefügt. Es müssen die Verbindungs-Parameter angegeben werden.

Es soll eine Verbindung zur Datenbank "data" hergestellt werden. Beim konfigurieren der Datenbank wurden Benutzername, Passwort, IP Adresse und Port Nummer festgelegt. Diese müssen in der Konfiguration angegeben werden.



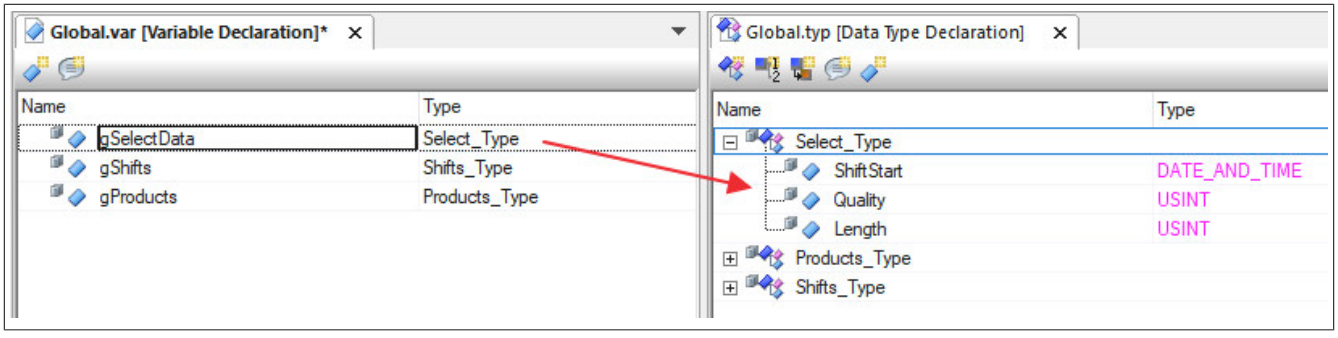
Die Abfragen werden im Abschnitt "Queries" definiert.

Ziel ist es, abhängig von der Produkt-ID, Information aus der "shifts", sowie "products" Tabelle anzufordern. Mithillfe vom Abfrage-Typ SELECT kann Information angefragt werden, da aber die Information aus zwei unterschiedlichen Tabellen kommt, ist eine komplexere Datenbankabfrage nötig.

Aus diesem Grund wird eine benutzerdefinierte Abfrage ("User defined") mit dem Namen "SelectData" erstellt. Um das gewünschte Ziel zu erreichen, kann das SQL-Schlüsselwort "INNER JOIN" verwendet werden. Dadurch kann in Abhängigkeit der Produkt-ID Information aus verschiedenen Tabellen angefordert werden. Für mehr Information dazu siehe hier.

Die gewünschte benutzerdefinierte Abfrage wird unter "Query" definiert. Die angeforderte Information wird an der Prozessvariable "gSelectData" gespeichert.

Die Prozessvariable ist vom benutzerdefinierten Datentypen "Select_Type". In dieser Struktur können die Qualität, Länge des Produkts sowie Startzeitpunkt der Schicht abgespeichert werden.



Mehr Information zu den einzelnen Abfrage-Typen kann hier gefunden werden.

Skript

Bevor sich mapp Database mit der Datenbank verbinden kann, muss ein Skript aktiviert werden. Dieses ist dafür verantwortlich, dass Daten zwischen der Datenbank und mapp Database ausgetauscht werden können. Das Skript kann im Abschnitt Server installieren heruntergeladen werden. Dort kann auch zusätzliche Information zu dem Skript gefunden werden.

Auf unserem B&R Tutorial Portal wird über das Tutorial mapp Database Part 1 (GER/ENG) erklärt, wie das Skript funktioniert und gestartet werden kann. (Zugangsdaten)

Verwendung der mapp Komponenten

Es werden die Funktionsbausteine **MpDatabaseCore** und **MpDatabaseQuery**, eingefügt und parametriert.

Wie die Funktionsbausteine parametriert werden, ist in den B&R Tutorials mapp Database Part 2 und mapp Database Part 3 erklärt.

Die konfigurierte Abfrage kann mithilfe von**MpDatabaseQuery** ausgeführt werden. Dabei muss der Abfrage-Name, welcher in der Konfiguration definiert wurde, am Eingangsparameter "Name" von **MpDatabaseQuery** angegeben werden.

Durch "Connect = `TRUE`" an **MpDatabaseCore** wird die Verbindung zur Datenbank hergestellt. Mithilfe von "Execute = `TRUE`" an **MpDatabaseQuery** wird die Abfrage gestartet.

Die angeforderten Daten werden, wie in der MpDatabaseCore Konfiguration festgelegt, an der Prozessvariable "gSelectData" angezeigt und können überprüft werden.

4 Bibliotheken

4.1 Datentypen

4.1.1 MpDatabaseCoreInfoType

Mit diesem Datentyp wird für den Funktionsbaustein zusätzliche Information bereitgestellt.

Parameter	Datentyp	Beschreibung
Diag	`MpDatabaseDiagType`	Eine Diagnosestruktur für die StatusID.

4.1.2 MpDatabaseDiagType

Dieser Datentyp wird in der Struktur für zusätzliche Informationen als Unterstruktur für die Diagnose verwendet und liefert weitere Infos über die StatusID.

Parameter	Datentyp	Beschreibung
StatusID	`MpDatabaseStatusIDType`	StatusID Diagnose Struktur.

4.1.3 MpDatabaseQueryInfoType

Mit diesem Datentyp wird für den Funktionsbaustein zusätzliche Information bereitgestellt.

Parameter	Datentyp	Beschreibung
ArraySize	`UDINT`	Gibt an wieviele Einträge in das Array geschrieben werden können.
Rows	`MpDatabaseQueryRowsInfoType`	Dieser Parameter gibt Auskunft über die gestartete Abfrage an.
Diag	`MpDatabaseDiagType`	Eine Diagnosestruktur für die StatusID.

4.1.4 MpDatabaseQueryRowsInfoType

Dieser Datentyp gibt Auskunft über die gestartete Abfrage an.

Parameter	Datentyp	Beschreibung
Total	`UDINT`	Anzahl der Einträge, welche durch die Abfrage an die Datenbank zurückgeschickt wurden.
Read	`UDINT`	Gibt an wie viele Einträge bereits gelesen wurden. Die erhaltene Information aus der Datenbank wird an der PV angezeigt, welche in der MpDatabaseCore Konfiguration definierten Abfrage angegeben wurde.
Remaining	`UDINT`	Gibt an, wie viele Einträge noch nicht gelesen wurden.

4.1.5 MpDatabaseStatusIDType

Dieser Datentyp wird in der Struktur für zusätzliche Informationen als Unterstruktur für die Diagnose verwendet und liefert weitere Infos über die StatusID.

Parameter	Datentyp	Beschreibung
ID	MpDatabaseErrorEnum	Fehlerkode des Funktionsbausteins.
Severity	MpComSeveritiesEnum	Beschreibt welche Art Info die StatusID liefert (Erfolg, Information, Warnung, Fehler).
Code	UINT	Code der Status ID. Diese Fehlernummer kann in der Hilfe für weitere Informationen gesucht werden.

4.2 Funktionsbausteine

Name	Beschreibung
MpDatabaseCore	Mithilfe von MpDatabaseCore kann man sich mit einer in der MpDatabaseCore Konfiguration definierten Datenbank verbinden.
MpDatabaseQuery	Mithilfe von MpDatabaseQuery kann eine in der MpDatabaseCore Konfiguration definierte Abfrage gestartet werden.

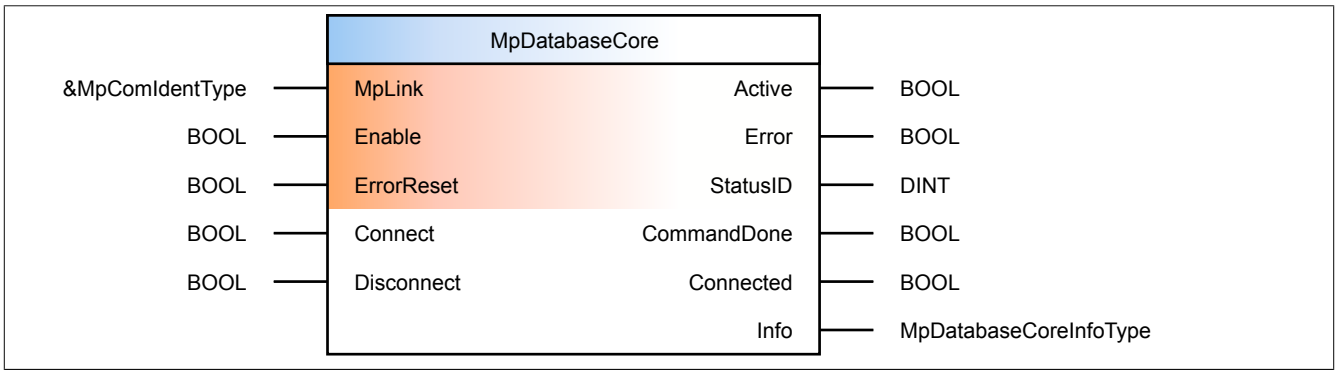
4.2.1 MpDatabaseCore

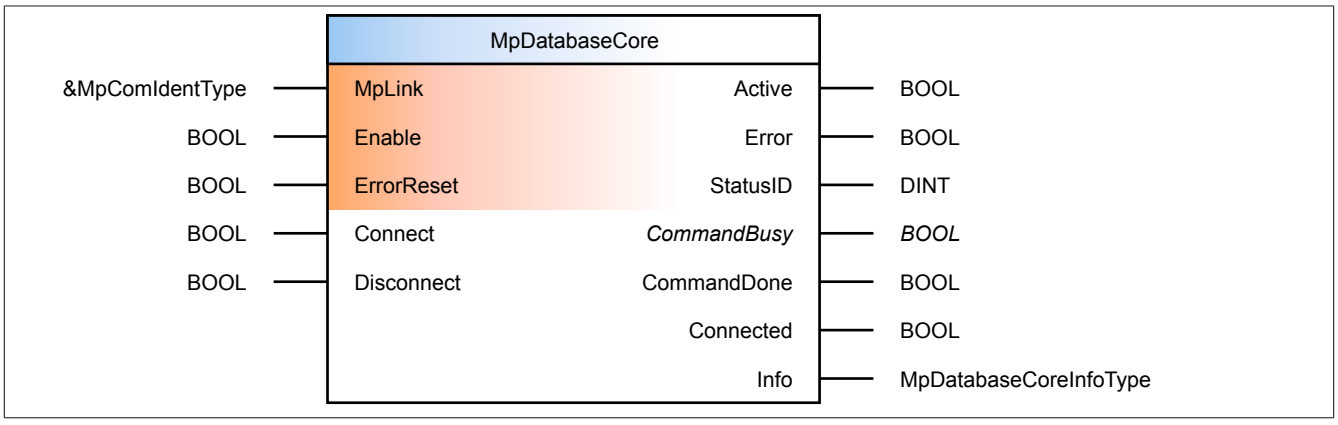
Mithilfe von MpDatabaseCore kann man sich mit einer in der MpDatabaseCore Konfiguration definierten Datenbank verbinden.

Auf unserem B&R Tutorial Portal sind Tutorials zum Thema mapp Database (GER/ENG) zu finden. (Zugangsdaten)

Funktionsbaustein

Optionale Parameter





Schnittstelle

I/O	Parameter	Datentyp	Beschreibung
IN	MpLink	Zeiger auf MpComIdentType	Verbindung zur mapp Konfiguration (MpLink einer MpDatabaseCore Konfiguration)
IN	Enable	BOOL	FB ist aktiv solange der Eingang gesetzt ist.
IN	ErrorReset	BOOL	Dient zum Zurücksetzen von Funktionsbaustein Fehlern.
IN	Connect	BOOL	Kommando, um sich mit einer Datenbank zu verbinden.
IN	Disconnect	BOOL	Kommando, um die Verbindung zu einer Datenbank zu trennen.
OUT	Active	BOOL	FB ist aktiv.
OUT	Error	BOOL	Fehler bei Abarbeitung aufgetreten.
OUT	StatusID	DINT	Statusinformation.

I/O	Parameter	Datentyp	Beschreibung
OUT	*CommandBusy*	BOOL	Funktionsblock führt aktuell einen Befehl aus.
OUT	CommandDone	BOOL	Abarbeitung erfolgreich. FB ist fertig.
OUT	Connected	BOOL	Gibt an ob eine Verbindung zu einer Datenbank aufgebaut ist.
OUT	Info	MpDatabaseCoreInfoType	Zusätzliche Komponenteninformationen.

mapp Konzept

Im Abschnitt mapp Komponenten wird erklärt wie die mapp Komponenten aufgebaut sind. Außerdem wird auf die richtige Verwendung von mapp Komponenten hingewiesen (z.B. beim Download).

4.2.1.1 Beschreibung

Für diesen Funktionsbaustein wird der MpLink einer MpDatabaseCore Konfiguration verwendet.

Über das Kommando "Connect = TRUE" verbindet man sich mit einer in der MpDatabaseCore Konfiguration definierten Datenbank. Der Ausgangsparameter "Connected = TRUE" signalisiert, dass sich der Funktionsbaustein erfolgreich mit der Datenbank verbunden hat.

Um die Verbindung mit der Datenbank zu trennen, wird das Kommando "Disconnect" verwendet.

Über die Info-Stuktur des Funktionsbausteins kann der aktuelle Status des Funktionsbausteins abgelesen werden.

Wie der Funktionsbaustein Schritt für Schritt parametriert wird, ist im Getting Started Verbindung mit Datenbank herstellen erklärt.

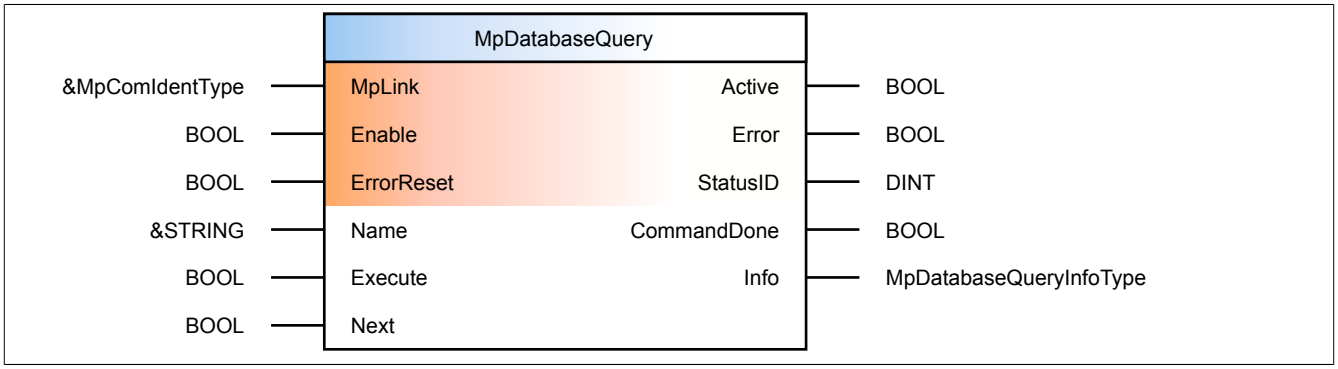
4.2.2 MpDatabaseQuery

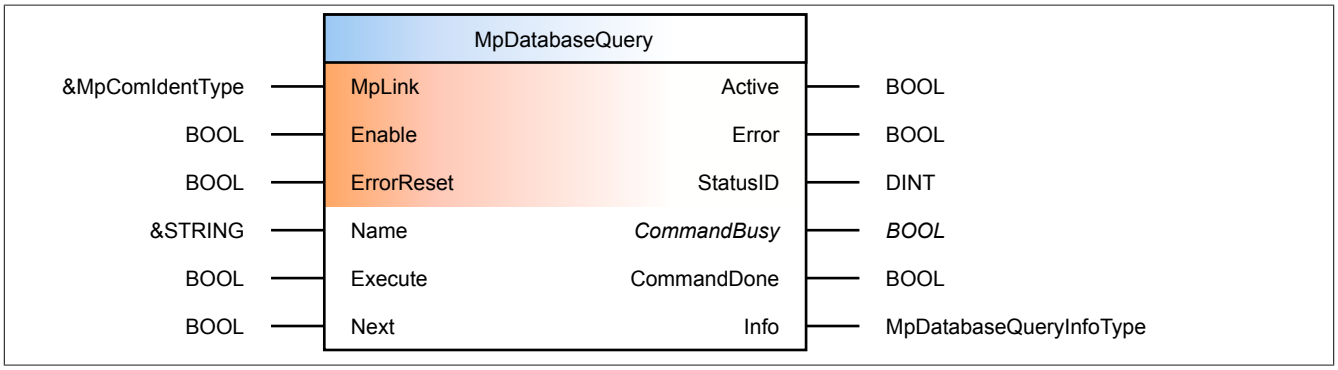
Mithilfe von MpDatabaseQuery kann eine in der MpDatabaseCore Konfiguration definierte Abfrage gestartet werden.

Auf unserem B&R Tutorial Portal sind Tutorials zum Thema mapp Database (GER/ENG) zu finden. (Zugangsdaten)

Funktionsbaustein

Optionale Parameter





Schnittstelle

I/O	Parameter	Datentyp	Beschreibung
IN	MpLink	Zeiger auf MpComIdentType	Verbindung zur mapp Konfiguration (MpLink einer
IN	Enable	BOOL	FB ist aktiv solange der Eingang gesetzt ist.
IN	ErrorReset	BOOL	Dient zum Zurücksetzen von Funktionsbaustein Fehlern.

I/O	Parameter	Datentyp	Beschreibung
IN	Name	`Zeiger auf STRING[50]`	Name der Abfrage. Die Abfrage wird in der MpDatabaseCore Konfiguration definiert.
IN	Execute	`BOOL`	Kommando um Abfrage zu starten.
IN	Next	`BOOL`	Befehl um weitere Einträge der Abfrage anzuzeigen.
OUT	Active	`BOOL`	FB ist aktiv.
OUT	Error	`BOOL`	Fehler bei Abarbeitung aufgetreten.
OUT	StatusID	`DINT`	Statusinformation.
OUT	*CommandBusy*	`BOOL`	Funktionsblock führt aktuell einen Befehl aus.
OUT	CommandDone	`BOOL`	Abarbeitung erfolgreich. FB ist fertig.
OUT	Info	`MpDatabaseQueryInfoType`	Zusätzliche Komponenteninformationen.

mapp Konzept

Im Abschnitt mapp Komponenten wird erklärt wie die mapp Komponenten aufgebaut sind. Außerdem wird auf die richtige Verwendung von mapp Komponenten hingewiesen (z.B. beim Download).

4.2.2.1 Beschreibung

Für diesen Funktionsbaustein wird derselbe MpLink verwendet, wie für MpDatabaseCore.

Starten der Abfrage

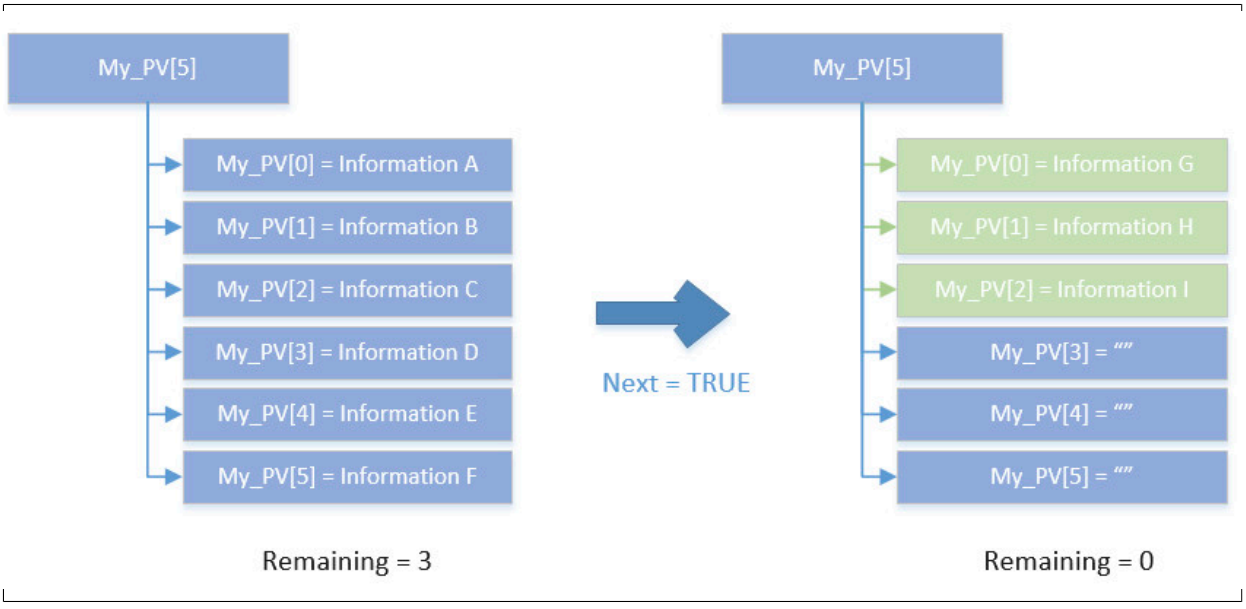
Am Eingangsparameter "Name" wird definiert welche Abfrage gestartet werden soll. Dabei muss ein Name angegeben werden welcher in der MpDatabaseCore Konfiguration im Abschnitt "Queries" definiert wurde.

Über das Kommando "Execute = `TRUE`" wird die Abfrage gestartet. Die Information wird an der Prozessvariable angzeigt welche in der MpDatabaseCore Konfiguration definierten Abfrage angegeben wurde.

Mithilfe von "Next" kann weitere Information angefordert werden. Dazu werden die Parameter in der Info-Struktur benötigt.

In der Info-Struktur können folgende Informationen entnommen werden:

- **ArraySize:** Gibt an wie viel die Platz die Prozess Variable, welche in der MpDatabaseCore Konfiguration im Abschnitt "Queries" definiert wurde, bietet. Ist die Prozess Variable ein STRING Array von 100 Elementen, so wird 100 angezeigt.
- **Rows:** In der MpDatabaseQueryRowsInfoType kann weitere Information zu der durchgeführten Abfrage gefunden werden. Dazu zählt wieviele Einträge insgesamt in der Datenbank gefunden wurde ("Total"), wieviele Einträge bereits über die PV dargestellt wurden ("Read") und wie viele Einträge noch gelesen werden können ("Remaining"). Wurde als Prozessvariable ein STRING Array von 6 Elementen verwendet, aber die Anfrage liefert 9 Einträge aus der Datenbank zurück, so wird "Remaining=3" angezeigt. Um die letzten 3 Einträge zu bekommen, muss der Befehl "Next = `TRUE`" von MpDatabaseQuery verwendet werden. Die letzten 3 Einträge werden auf die Variable geschrieben. Dabei wird die Information auf die ersten Elemente der Prozessvariable geschrieben:



4.3 Statusnummern

4.3.1 1083194019: Tabelle wurde erweitert

Beschreibung:

Die Tabelle wurde um eine Spalte erweitert.

4.3.2 1083194020: Tabelle wurde nicht bearbeitet

Beschreibung:

Die Tabelle wurde nicht bearbeitet.

4.3.3 1083194013: Buffer voll

Beschreibung:

Daten wurde in die PV kopiert, aber es ist noch mehr Information in der Datenbank vorhanden. Buffer der PV ist voll.

Ursache / Abhilfe:

- Weitere Information durch das Kommando "Next" an MpDatabaseQuery anfordern

4.3.4 -1064239091: Ungültige Konfiguration

Beschreibung:

Konfiguration konnte während dem Erstellen der Komponente nicht gelesen werden. Weitere Details siehe Logger.

Reaktion:

Der Funktionsbaustein wird an den Ausgängen "StatusID" und "Error" anzeigen, dass ein Fehler aktiv ist, weiters sind alle Funktionen solange nicht verfügbar.

Ursache / Abhilfe:

- Die Konfiguration ist beschädigt
- Die Funktion ist in der Konfiguration nicht aktiviert

Diese Funktionsbausteine / Funktionen können diesen Fehler melden:

- MpDatabaseCore

4.3.5 -1064239098: MpLink wird bereits benützt

Beschreibung:

Dieser MpLink wird bereits benützt.

Reaktion:

Der Funktionsbaustein wird an den Ausgängen "StatusID" und "Error" anzeigen, dass ein Fehler aktiv ist, weiters sind alle Funktionen solange nicht verfügbar.

Ursache / Abhilfe:

- Ist der ausgewählte Download Modus "Overload", sollte im Exit Programm die Mapp Komponente mit "Enable = FALSE" deaktiviert werden. Als Abhilfe kann auch der Modus "Copy" oder "One cycle" verwendet werden. Hierbei ist es nicht nötig die mapp Komponente zu deaktivieren.
- Überprüfen welche Komponente diesen MpLink bereits benützt
- Neuen MpLink erstellen

Diese Funktionsbausteine / Funktionen können diesen Fehler melden:

- MpDatabaseCore

4.3.6 -1064239099: MpLink Inhalt ist ungültig

Beschreibung:

Der Wert in der Variable "MpLink" am Eingang des Funktionsbaustein ist ungültig.

Reaktion:

Der Funktionsbaustein wird an den Ausgängen "StatusID" und "Error" anzeigen, dass ein Fehler aktiv ist, weiters sind alle Funktionen solange nicht verfügbar.

Ursache / Abhilfe:

- Die Werte in der Struktur MpComIdentType dürfen nicht beschrieben werden
- Es muss eine entsprechende Konfiguration für diese mapp Komponenten vorhanden sein (siehe erster Absatz der Beschreibung von den unten angeführten Funktionen / Funktionsbausteinen)

Diese Funktionsbausteine / Funktionen können diesen Fehler melden:

- MpDatabaseCore

4.3.7 -1064239100: MpLink wurde verändert

Beschreibung:

Der Wert am "MpLink" Eingang hat sich zur Laufzeit der Komponenten ("Enable" = `TRUE`) verändert.

Reaktion:

Der Funktionsbaustein wird an den Ausgängen "StatusID" und "Error" anzeigen, dass ein Fehler aktiv ist, weiters sind alle Funktionen solange nicht verfügbar.

Ursache / Abhilfe:

- Den Wert am "MpLink" Eingang nur verändern während die Komponente inaktiv ist ("Enable" = `FALSE`)

Diese Funktionsbausteine / Funktionen können diesen Fehler melden:

- MpDatabaseCore

4.3.8 -1064239101: MpLink Verbindung nicht erlaubt

Beschreibung:

Der verwendete Wert am "MpLink" Eingang ist nicht erlaubt.

Reaktion:

Der Funktionsbaustein wird an den Ausgängen "StatusID" und "Error" anzeigen, dass ein Fehler aktiv ist, weiters sind alle Funktionen solange nicht verfügbar.

Ursache / Abhilfe:

- Komponente ist direkt mit mpCOM_MAIN oder mpCOM_STANDALONE verbunden, das wird nicht unterstützt

Diese Funktionsbausteine / Funktionen können diesen Fehler melden:

- MpDatabaseCore

4.3.9 -1064239102: MpLink ist ein Nullzeiger

Beschreibung:

Der "MpLink" Eingang ist nicht verbunden - NULL Zeiger.

Reaktion:

Der Funktionsbaustein wird an den Ausgängen "StatusID" und "Error" anzeigen, dass ein Fehler aktiv ist, weiters sind alle Funktionen solange nicht verfügbar.

Ursache / Abhilfe:

- "MpLink" Eingang am Funktionsbaustein überprüfen

Diese Funktionsbausteine / Funktionen können diesen Fehler melden:

- MpDatabaseCore

4.3.10 -1064239103: Komponente konnte nicht erstellt werden

Beschreibung:

Die mapp Komponente konnte nicht erstellt werden und wird nicht aktiviert. Weitere Details siehe Logger.

Reaktion:

Der Funktionsbaustein wird an den Ausgängen "StatusID" und "Error" anzeigen, dass ein Fehler aktiv ist, weiters sind alle Funktionen solange nicht verfügbar.

Ursache / Abhilfe:

- Service zur Konfiguration des Funktionsbausteins nicht verfügbar - Problem mit MpCom
- Registry nicht lesbar - Problem mit MpCom
- Details zur Fehlerursache im Logger
- Bei der Verwendung von verschlüsselten Passwörtern in der Konfiguration muss darauf geachtet werden, dass das Passwort im Nachhinein nicht manipuliert wurde, wie beispielsweise hinzufügen oder löschen von Zeichen. Für mehr Information siehe hier im Abschnitt "Passwortverschlüsselung in der Konfiguration"

Diese Funktionsbausteine / Funktionen können diesen Fehler melden:

- MpDatabaseCore

4.3.11 -1064167416: Fehler in der Abfrage

Beschreibung:

Bei der interpretierung der SQL Abfrage ist ein Fehler aufgetreten. Fehlerursache: {2:ErrorNumber}

Reaktion:

Der Funktionsblock wird an den Ausgängen "StatusID" und "Error" anzeigen, dass ein Fehler aktiv ist, weiters sind alle Funktionen solange nicht verfügbar.

Zusätzliche Information:

- {2:ErrorNumber}: Fehlerursache

Ursache / Abhilfe:

- Kontrollieren der Abfrage ob richtige Tabellen- und Spaltennamen verwendet wurden

Konstante

`mpDATABASE_ERR_QUERY_RESULT`

Diese Funktionsblöcke / Funktionen können diesen Fehler melden:

- MpDatabaseQuery

4.3.12 -1064167415: Ungültiger Benutzer

Beschreibung:

Die Abfrage {2:QueryName} kann nur vom Database-Widget ausgeführt werden und nicht vom Funktionsbaustein. Falscher Benutzer.

Reaktion:

Der Funktionsblock wird an den Ausgängen "StatusID" und "Error" anzeigen, dass ein Fehler aktiv ist, weiters sind alle Funktionen solange nicht verfügbar.

Zusätzliche Information:

- {2:QueryName}: Abfrage

Ursache / Abhilfe:

- Abfrage vom Database-Widget ausführen
- Abfrage in MpDatabaseCore Konfiguration anpassen, damit Abfrage von Funktionsbaustein ausgeführt werden kann

Konstante

`mpDATABASE_ERR_INVALID_USER`

Diese Funktionsblöcke / Funktionen können diesen Fehler melden:

- MpDatabaseQuery

4.3.13 -1064289625: Versionen nicht kompatibel

Beschreibung:

Die verwendete Skript-Version ist nicht mit der aktuellen mapp Services Version kompatibel.

Zusätzliche Information

- {2:ErrorNumber}: Fehlernummer. Dabei handelt es sich um eine SQL-Fehlernummer welche zum Beispiel hier gefunden werden kann.

Ursache / Abhilfe:

- Verwenden einer kompatiblen mapp Services und Skript Kombination. Für mehr Information siehe hier.

Konstante

`mpDATABASE_ERR_VERSION_MISMATCH`

4.3.14 -1064289624: Verbindungsaufbau nicht möglich

Beschreibung:

Es konnte keine Verbindung zum Skript aufgebaut werden.

Ursache / Abhilfe:

- Die aktuelle Skript-Version kann nicht mit der verwendeten mapp Services Version verwendet werden. Für mehr Information siehe hier.

Konstante

`mpDATABASE_ERR_CONNECTION_NOT_ESTABLISHED`

4.3.15 -1064167419: Ungültiger Datentyp

Beschreibung:

Der Datentyp {2:TypeName} wird nicht unterstützt.

Reaktion:

Der Funktionsblock wird an den Ausgängen "StatusID" und "Error" anzeigen, dass ein Fehler aktiv ist, weiters sind alle Funktionen solange nicht verfügbar.

Zusätzliche Information

- {2:TypeName} : Datentyp

Ursache / Abhilfe:

- Verwenden eines anderen Datentypen. Eine Liste mit unterstützten Datentypen kann hier im Abschnitt "Angabe von Variablen" gefunden werden
- Es gilt zu beachten, dass keine verschachtelten Strukturen verwendet werden dürfen

Konstante

`mpDATABASE_ERR_INVALID_DATATYPE`

Diese Funktionsblöcke / Funktionen können diesen Fehler melden:

- MpDatabaseQuery

4.3.16 1083316228: Keine weiteren Daten

Beschreibung:

Die angefragten Daten wurden alle bereits in die angegebene PV kopiert. Es stehen keine weiteren Daten zur Verfügung.

Reaktion:

Der Funktionsblock wird diese Information an dem Ausgang "StatusID" anzeigen.

Ursache / Abhilfe:

- Keine weiteren Daten anfragen durch "Next = `TRUE`" an MpDatabaseQuery
- Starten einer neuen Abfrage

Konstante

`mpDATABASE_INF_NO_DATA`

Diese Funktionsblöcke / Funktionen können diesen Fehler melden:

- MpDatabaseQuery

4.3.17 -1064167421: Abfrage doppelt

Beschreibung:

Die Abfrage {2:QueryName} kann nicht hinzugefügt werden, da die Abfrage bereits existiert.

Reaktion:

Der Funktionsblock wird an den Ausgängen "StatusID" und "Error" anzeigen, dass ein Fehler aktiv ist, weiters sind alle Funktionen solange nicht verfügbar.

Zusätzliche Information

- {2:QueryName}: Name der Abfrage

Ursache / Abhilfe:

- Verwenden eines anderen Abfrage-Namens

Konstante

`mpDATABASE_ERR_DUPLICATE_QUERY`

Diese Funktionsblöcke / Funktionen können diesen Fehler melden:

- MpDatabaseQuery

4.3.18 -1064167422: Ungültige Abfrage

Beschreibung:

Die Abfrage {2:QueryName} existiert nicht.

Reaktion:

Der Funktionsblock wird an den Ausgängen "StatusID" und "Error" anzeigen, dass ein Fehler aktiv ist, weiters sind alle Funktionen solange nicht verfügbar.

Zusätzliche Information

- {2:QueryName}: Name der Abfrage

Ursache / Abhilfe:

- Überprüfen des angegebenen Abfragen-Namen
- Verwenden eines anderen Abfrage-Namen
- Überprüfen ob die richtige Skript-Version verwendet wird. Welche mapp Services Version mit welcher Skript-Version funktioniert, ist im Abschnitt Server installieren erklärt.

Konstante

`mpDATABASE_ERR_INVALID_QUERY`

Diese Funktionsblöcke / Funktionen können diesen Fehler melden:

- MpDatabaseQuery

4.3.19 -1064167423: HTTP Server Fehler

Beschreibung:

Verbindung mit Datenbank nicht möglich. Der Server meldet einen Fehler. Für mehr Information siehe Logger. Fehlerursache: {2:ErrorNumber}

Reaktion:

Der Funktionsblock wird an den Ausgängen "StatusID" und "Error" anzeigen, dass ein Fehler aktiv ist, weiters sind alle Funktionen solange nicht verfügbar.

Zusätzliche Information

- {2:ErrorNumber}: Fehlernummer

Ursache / Abhilfe:

- Angegebene IP Adresse und Portnummer in der MpDatabaseCore Konfiguration überprüfen
- Überprüfen ob Skript aktiviert wurde. Für mehr Information siehe hier
- Für mehr Information siehe Logger
- Überprüfen der Lösungsvorschläge in Abschnitt FAQ

Konstante

`mpDATABASE_ERR_HTTP_SERVER`

Diese Funktionsblöcke / Funktionen können diesen Fehler melden:

- MpDatabaseCore

4.3.20 -1064167424: SQL Server Fehler

Beschreibung:

Verbindung mit Datenbank nicht möglich. Der SQL Server meldet einen Fehler. Für mehr Information siehe Logger. Fehlerursache: {2:ErrorNumber}

Reaktion:

Der Funktionsblock wird an den Ausgängen "StatusID" und "Error" anzeigen, dass ein Fehler aktiv ist, weiters sind alle Funktionen solange nicht verfügbar.

Zusätzliche Information

- {2:ErrorNumber}: Fehlernummer. Dabei handelt es sich um eine SQL-Fehlernummer welche zum Beispiel hier gefunden werden kann.

Ursache / Abhilfe:

- Überprüfen des angegebenen Datenbank-Namens in der MpDatabaseCore Konfiguration
- Für mehr Information siehe Logger
- Für weitere Lösungsvorschläge siehe Abschnitt FAQ

Konstante

`mpDATABASE_ERR_SQL_SERVER`

Diese Funktionsblöcke / Funktionen können diesen Fehler melden:

- MpDatabaseCore

4.3.21 -1064289626: Abfrage kann nicht gestartet werden

Beschreibung:

Die angegebene Abfrage kann ohne Tabellen-Namen nicht gestartet werden.

Ursache / Abhilfe:

- Überprüfung ob der Name einer Tabelle angegeben wurde

4.3.22 -1064289627: Angegebene Zeit nicht gültig

Beschreibung:

Die angegebene Zeit "{2:Time}" welche für die Abfrage verwendet wurde ist in dem verwendeten Datenbank-System nicht gültig.

Ursache / Abhilfe:

- Überprüfung der angegebenen Zeit
- Negative Zeitangaben dürfen in MS SQL und PostgreSQL nicht verwendet werden

Zusätzliche Information

- {2:Time}: Angegebene Zeit

4.3.23 -1064289630: Falsche PV-Angabe

Beschreibung:

Die in der SQL-Abfrage angegebene PV stimmt nicht mit den Spalten der Tabelle in der Datenbank überein.

Ursache / Abhilfe:

- Überprüfung der PV
- Bei Angabe einer Struktur müssen die PV-Namen mit den Spalten-Namen der Tabelle in der Datenbank übereinstimmen

4.3.24 -1064289636: PV existiert bereits

Beschreibung:

Die Prozess Variable {2:PVName} existiert bereits in dieser Abfrage.

Zusätzliche Information

- {2:PVName} : Prozess Variable

Ursache / Abhilfe:

- Überprüfen der PV
- Angabe einer anderen PV

4.3.25 -1064289637: SQL Fehlernachricht

Beschreibung:

Der SQL Server meldet einen Fehler {2:ErrorNumber} mit der Fehlernachricht {3:ErrorMessage}.

Zusätzliche Information

- {2:ErrorNumber}: Fehlernummer
- {3:ErrorMessage}: Fehlernachricht

Ursache / Abhilfe:

- Für mehr Information siehe Logger
- Für weitere Lösungsvorschläge siehe Abschnitt FAQ

4.3.26 -1064289638: HTTP Reaktion ungültig

Beschreibung:

Die HTTP Reaktion ist ungültig. Das Skript um Verbindung zur Datenbank aufzubauen ist ungültig.

Ursache / Abhilfe:

- Überprüfen des Python-Skripts. Für mehr Information siehe hier. Im Abschnitt FAQ sind weitere Lösungsvorschläge zu diesem Problem zu finden.

4.3.27 -1064289639: HTTP Anfrage ungültig

Beschreibung:

HTTP Anfrage ungültig. Das Skript welches die Verbindung zur Datenbank aufbaut ist ungültig.

Ursache / Abhilfe:

- Überprüfen des Python-Skripts. Für mehr Information siehe hier. Im Abschnitt FAQ sind weitere Lösungsvorschläge zu diesem Problem zu finden.

4.3.28 -1064289640: SQL Server nicht verbunden

Beschreibung:

Es besteht keine Verbindung zum SQL Server.

Ursache / Abhilfe:

- Überprüfen der allgemeinen Einstellungen in der MpDatabaseCore Konfiguration
- Für weitere Lösungsvorschläge siehe Abschnitt FAQ

4.3.29 -2137909241: Warnung für Abfrage

Beschreibung:

Beim interpretieren der SQL-Abfrage, hat die PV {2:PvName} eine Warnung ausgelöst

Reaktion

Der Funktionsblock wird am Ausgang "StatusID" anzeigen, dass eine Warnung aktiv ist.

Zusätzliche Information

- {2:PVName}: Prozessvariable

Ursache / Abhilfe:

- Angabe eines anderen Datentypen für die Prozessvariable. Die erlaubten Datentypen sind im Abschnitt Angabe von Variablen aufgelistet
- Eine Spalte liefert den Wert NULL zurück
- Für weitere Information siehe Logger
- Die Spaltenanzahl der Datenbank-Tabelle stimmt nicht mit der angegebenen PV überein

Konstante

`mpDATABASE_WRN_QUERY_RESULT`

4.3.30 -2138031458: PV nicht gefunden

Beschreibung:

Die Prozessvariable {2:PVName} konnte nicht lokalisiert werden.

Zusätzliche Information

- {2:PVName}: Prozessvariable

Ursache / Abhilfe:

- Überprüfen der Prozessvariable
- Überprüfen ob Prozessvariable im Programm verwendet wird

4.3.31 -2138031457: PV außerhalb des Limits

Beschreibung:

Die Prozessvariable befindet sich außerhalb des erlaubten Limits

Ursache / Abhilfe:

- Überprüfen der Prozessvariable
- Angabe eines anderen Datentypen für PV

4.3.32 -2138031456: Wert von Datenbank ist NULL

Beschreibung:

Der erhaltene Wert von der Datenbank enthält NULL.

Ursache / Abhilfe:

- Wert in der Datenbank prüfen

4.3.33 -2138031455: PV stimmt nicht mit Spalte überein

Beschreibung:

Für eine Abfrage wurde eine Prozessvariable verwendet, deren Struktur nicht mit der Spalte in der Datenbank-Tabelle übereinstimmt. Die Struktur der PV enthält Variablen welche nicht in der Datenbank-Tabelle vorkommen.

Ursache / Abhilfe:

- Angabe einer PV deren Struktur mit der in der Datenbank-Tabelle übereinstimmt
- Für weitere Information siehe "Meldungen in der Kommando-Konsole" im Abschnitt FAQ.

Konstante

`mpDATABASE_WRN_RESPONSE_TO_PV_MISMATCH`