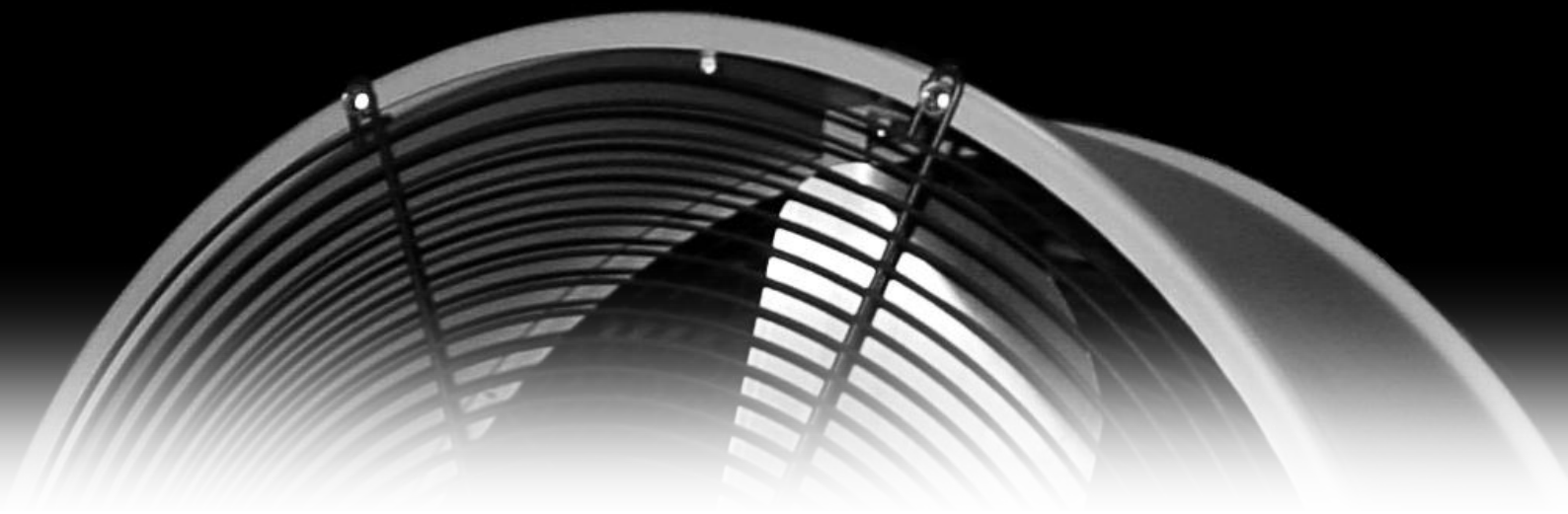
DataCool®



Schnelle Anwendungsentwicklung auf Client-/Server-Basis

Entwicklerhandbuch

Version 2020
Änderungsstand 19.10.2019

www.datacool.net

Inhaltsverzeichnis

1 Einleitung

1.1 Vorwort

DataCool ist ein Entwicklungswerkzeug für Client-/Server-Anwendungen. Mit Hilfe der intuitiven Benutzeroberfläche können versierte Anwender innerhalb kurzer Zeit Datenbank-Anwendungen erstellen. Viele zeitintensive Programmieraufgaben werden von DataCool automatisch im Hintergrund für Sie erledigt. Auf diese Weise kommen Sie auch ohne große Vorkenntnisse schnell ans Ziel.

Was ist eine Client-/Server-Anwendung?

Die Client-/Server-Technologie basiert auf der SQL-Abfragesprache. SQL ist die Abkürzung für ***S****tructured* ***Q****uery* ***L****anguage*. Hierbei handelt es sich um eine Sprache zur Definition, Manipulation und Abfrage von Daten in relationalen Datenbanken. Die Kommunikation zwischen dem Server und dem Client erfolgt mit Hilfe dieser Sprache.

Was sind die Vorzüge der Client-/Server-Technik?

- Die SQL-Abfragesprache ist standardisiert und bietet somit eine weitgehende Unabhängigkeit von der verwendeten Software.

- Die Verarbeitung der Daten erfolgt zum überwiegenden Teil im Server. Zwischen Server und Client werden nur Befehle und Ergebnisse übertragen. Diese Vorgehensweise führt zu einer Reduktion der Netzwerklast.

- Sperrkonflikte oder defekte Indexdateien, wie sie in Datei-basierten Datenbanken häufig auftreten, gehören der Vergangenheit an.

- Mit Hilfe der Transaktionsverarbeitung können Befehlsfolgen zu einer logischen Einheit zusammengefasst werden. Die Befehlskette wird dann nur komplett oder gar nicht ausgeführt.

Muss ich SQL lernen, bevor ich mit DataCool arbeiten kann?

DataCool setzt keine Kenntnisse in SQL voraus. Mit dem Formulargenerator erstellen Sie in einem Arbeitsgang Eingabemaske und SQL-Tabelle. Danach können Sie sofort mit der Datenerfassung beginnen.

Für die wichtigsten Programmieraufgaben stehen Ihnen Hilfswerkzeuge (Wizards) zur Verfügung. Mit der Hilfe der Wizards können Sie die SQL-Sprache spielend erlernen.

Für anspruchsvolle Aufgaben steht Ihnen die DataCool Scriptsprache zur Verfügung. Diese Scriptsprache verbindet auf einfache Weise Elemente aus SQL und BASIC.

1.2 Vereinbarungen zur Schreibweise

In diesem Handbuch werden verschiedene Schreibweisen verwendet um eine bessere Lesbarkeit zu gewährleisten:

F2	Funktionstaste
Neu	Schaltfläche oder Toolbar-Button
`Courier`	Scriptbefehl oder Ableitungsformel
	Nützlicher Hinweis
	Wichtige Information, die unbedingt beachtet werden sollte
	Neues Feature ab Version DataCool 2020

2 Installation und Konfiguration

2.1 Funktionsweise von DataCool-Anwendungen

Für die Erstellung einer DataCool-Anwendung benötigen Sie eine SQL-Datenbank.

Im **Einzelplatzbetrieb** wird hierfür der „**SQL Server Express 2012 Local DB**“ vom DataCool Setup installiert. Verwenden Sie hierzu eines der folgenden Setup-Programme:

- Setup DataCool 2020 32 Bit.exe
- Setup DataCool 2020 64 Bit.exe

Das Einzelplatz-Setup ist für folgende Betriebssysteme geeignet:

- Windows 7
- Windows 8
- Windows 10

Im **Netzwerkbetrieb** können Sie den „**SQL Server Express 2012**“ kostenfrei von Microsoft herunterladen und auf dem Server installieren. Einzelheiten hierzu erfahren Sie bei Bedarf in Kapitel 9. Auf den einzelnen Arbeitsplätzen verwenden Sie dann nur das „Client Setup“ von DataCool:

- Setup DataCool 2020 Client.exe

Das Client Setup ist für folgende Betriebssysteme geeignet (32 Bit und 64 Bit):

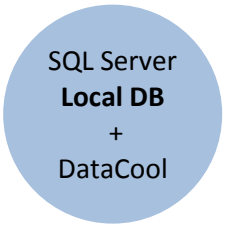
- Windows 7
- Windows 8
- Windows 10

Das Client Setup müssen Sie nur dann auf dem Server ausführen, falls Sie direkt auf dem Server mit DataCool arbeiten möchten. Andernfalls genügt es den SQL Server Express auf der Servermaschine zu installieren.

In der gesamten nachfolgenden Dokumentation wird unabhängig von der eingesetzen Version stets die Bezeichnung „SQL Server“ verwendet.

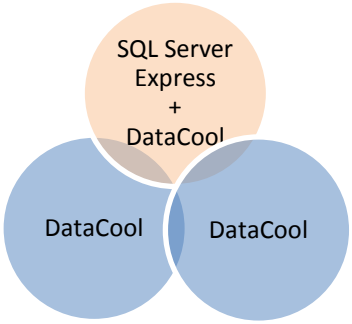
Eine mit dem SQL Server Express LocalDB entwickelte Datenbank können Sie zu einem späteren Zeitpunkt jederzeit auf den SQL Server Express migrieren, wenn Sie von Einzelplatz- auf Netzwerkbetrieb umsteigen möchten.

Je nach Anwendungsfall ergeben sich folgende Installationsmöglichkeiten:



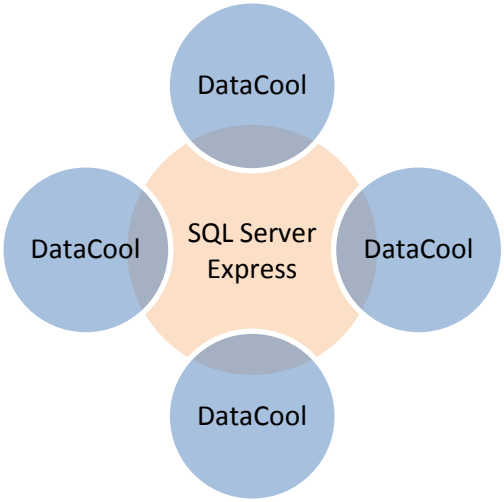
Einzelplatz

Der SQL Server Express Local DB und DataCool werden auf demselben Rechner installiert. Hierfür benötigen Sie nur eine DataCool Lizenz.



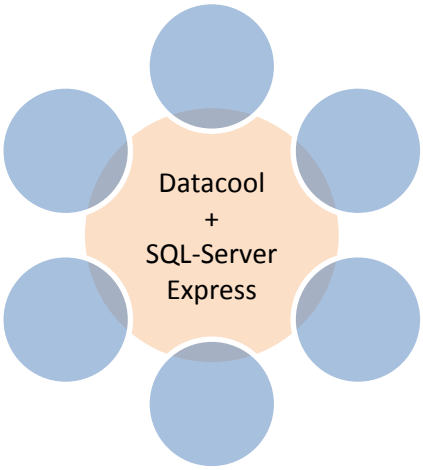
Peer-To-Peer Netzwerk

Der schnellste Rechner in Ihrer Arbeitsgruppe wird als Server ausgewählt. Auf diesem Rechner wird der SQL Server Express installiert (siehe Kapitel 9. Im nebenstehenden Beispiel ist diese Station rot markiert. Auf allen Rechnern der Arbeitsgruppe wird mit dem Client Setup DataCool installiert. Sie benötigen für alle Rechner der Arbeitsgruppe eine DataCool-Lizenz.



Client-/Server-Netzwerk

Der SQL Server Express wird auf dem Fileserver installiert. DataCool wird auf jedem Netzwerkarbeitsplatz installiert. Hierfür benötigen Sie eine DataCool Lizenz für jeden Client. Auf Wunsch können Sie auch auf dem Server mit DataCool arbeiten. In diesem Fall benötigen Sie auch für den Server eine DataCool Lizenz.



Terminal-Server Betrieb

Die Installation von DataCool erfolgt nur **einmal** auf dem Terminal Server. Der SQL Server Express wird ebenfalls auf dem Terminal Server installiert. In großen Netzwerken kann die Datenbank aber auch auf einem zusätzlichen Server installiert werden, um den Terminal Server zu entlasten. In dieser Betriebsart werden die DataCool Instanzen gezählt. Die benötigte Anzahl der Lizenzen hängt also davon ab, wie viele Benutzer auf dem Terminal Server **gleichzeitig** mit DataCool arbeiten.

2.2 Einzelplatz-Installation

Je nach Art Ihrer Windows-Version (32 Bit oder 64 Bit) laden Sie eines der folgenden Setups von www.datacool.net herunter:

- Setup DataCool 2020 32 Bit.exe
- Setup DataCool 2020 64 Bit.exe

Das **Einzelplatz Setup** installiert **DataCool** und den **SQL Server Express Local DB** in einem Arbeitsgang.

Bitte verwenden Sie dieses Setup nur für die **Erstinstallation** auf einem **Einzelplatz**. Für das Update von DataCool auf eine hörere Version verwenden Sie bitte das Client Setup.

DataCool Verzeichnis

Während der Einrichtung werden Sie nach dem Installationsverzeichnis für DataCool gefragt. Die Vorgabe **C:\Programme\DataCool** sollten Sie in der Regel nicht verändern.

Speicherort für Datenbanken

Datenbankdateien die Sie mit Hilfe von DataCool erstellen werden automatisch im lokalen Profilpfad des Benutzers abgelegt.

Das Einzelplatz-Setup installiert automatisch eine Testanwendung zur Lernzwecken. Weitere Einzelheiten zum „DataCool Tutorial“ erfahren Sie in Kapitel 3.3.

2.3 Client-Installation und Update

In einer Netzwerkumgebung benötigen einen SQL Server, der auf einem Server-Computer oder auf einem leistungsstarken Computer Ihrer Arbeitsgruppe installiert ist. Einzelheiten zur Installation des SQL Servers erfahren Sie bei Bedarf in Kapitel 9.

Auf den einzelnen **Netzwerkarbeitsplätzen** wird das **Client Setup** ausgeführt. Es kommt in folgenden Situationen zum Einsatz:

- **Erstinstallation** von DataCool auf einem **Netzwerk-Client**

- **Update** einer bestehenden Installation (Einzelplatz oder Client)

Achtung: Sollten Sie DataCool auf einem **Terminal Server** installieren, so muss dieser vor dem Start des Setup in den **Installationsmodus** versetzt werden. Verwenden Sie hierzu folgenden Befehl: **change user /install**. Nach erfolgter Installation schalten Sie den Terminal Server wieder in den Ausführungsmodus zurück: **change user /execute**. Diese Maßnahme ist zwingend erforderlich, da andernfalls keine korrekte Installation gewährleistet ist!

DataCool Verzeichnis

Während der Einrichtung werden Sie nach dem Installationsverzeichnis für DataCool gefragt. Wir empfehlen die Installation des Client auf einem **lokalen Laufwerk**. Die Vorgabe **C:\Programme\DataCool** sollten Sie deshalb in der Regel nicht verändern. Ihre DataCool-Anwendungen sind selbstverständlich trotzdem mehrplatzfähig, auch wenn Sie den Client auf einem lokalen Laufwerk installieren.

Update einer vorhandenen Installation

Mit dem Client Setup können Sie auch eine bestehende Installation von DataCool aktualisieren. Sie können das Programm einfach „drüber installieren“. Eine vorherige Deinstallation ist nicht erforderlich. Achten Sie aber darauf, dass während der Installation alle Instanzen von DataCool geschlossen sind.

2.4 Lizenzierung von DataCool

Nach der erstmaligen Installation von DataCool auf einem Rechner beginnt eine Testphase von 90 Tagen. Während dieser Testphase können Sie mit DataCool uneingeschränkt arbeiten. Die Restdauer der 90-Tagesfrist können Sie im Hauptmenü von DataCool in der Statusleiste ablesen. Nach Ablauf der Testphase müssen Sie eine Lizenz erwerben, bevor Sie weiterarbeiten können.

Es stehen Ihnen zwei unterschiedliche Lizenzformen zur Verfügung: **USB-Dongle** und **Lizenzcode**. Der Unterschied dieser beiden Lizenzen wird nachfolgend erläutert.

Nach dem Update von DataCool können Sie zunächst 30 Tage weiterarbeiten. Während dieser Zeit haben Sie Gelegenheit Ihren Dongle umzutauschen oder einen neuen Lizenzcode zu erwerben.

2.4.1 USB-Dongle

Der Dongle ist ein kleiner Stecker, der in einen freien USB-Port Ihres Rechners gesteckt wird. Der Vorteil dieser Lizenzform besteht darin, dass sie jederzeit auf einen anderen Rechner übertragen werden kann.

Auf einem Arbeitsplatz-Rechner können Sie beliebig viele DataCool-Sitzungen starten, wenn ein USB-Dongle eingesteckt ist. Auf einem Rechner mit Server-Betriebssystem berechtigt der USB-Dongle lediglich zum Aufruf einer einzelnen Sitzung.

USB-Dongle installieren

Stecken Sie den Dongle in einen beliebigen freien USB-Port Ihres Rechners. Der USB-Dongle wird automatisch erkannt und installiert. Die benötigten Treiberdateien sind bereits Bestandteil von Windows.

Eine zuvor installierte Demo-Version wird durch Einstecken des Dongles automatisch freigeschaltet. Zusätzliche Arbeitsschritte sind nicht erforderlich.

2.4.2 Lizenzcode

Der Lizenzcode ist ein 32-stelliger Schlüssel, der die Freischaltung Ihrer DataCool-Installation ermöglicht. Im Gegensatz zum USB-Dongle ist der Lizenzcode an einen bestimmten Rechner gebunden. Sollten Sie Ihren Rechner in der Zukunft tauschen, dann erhalten Sie von uns kostenfrei einen neuen Lizenzcode.

Lizenzcode installieren

Bei der Bestellung eines Lizenzcodes müssen Sie uns Ihre eindeutige Computer-ID mitteilen. Sie können Ihre Computer-ID wie folgt abrufen: Klicken Sie auf das Fragezeichen im Hauptmenü von DataCool um das Info-Fenster zu öffnen. In diesem Fenster erhalten Sie Informationen über die Programmversion und die gegenwärtige Lizenzierungsform. Bitte klicken Sie auf die Option „Lizenzcode freischalten“, um den nachfolgenden Dialog aufzurufen. Unter Vista und Windows 7 müssen Sie DataCool mit der rechten Maustaste und der Option „Als Administrator ausführen“ starten, damit der Dialog aufgerufen werden kann.

Im obigen Beispiel lautet die Computer-ID: 71A-8AE-225-ABC. Bitte teilen Sie uns Ihre ID zusammen mit der Bestellung mit. Mit dem nebenstehenden Pfeil können Sie die Computer-ID in die Zwischenablage kopieren.

Sobald Sie den Lizenzcode von uns erhalten haben kopieren Sie diesen mit dem Pfeilsymbol in das dafür vorgesehene Feld. Klicken Sie nun auf ***Freischalten*** um den Lizenzcode zu aktivieren.

Sollte Ihre Demo-Version bereits abgelaufen sein, so öffnet sich beim Start automatisch das DataCool-Infofenster. Auf diese Weise können Sie die Aktivierung auch nachträglich vornehmen.

Bei der Installation von DataCool auf einem **Terminal-Server** müssen Sie stets einen **Lizenzcode** verwenden. Der USB-Dongle ist in diesem Fall zur Lizenzierung nicht geeignet, da Sie auf dem Terminal-Server mit dem Dongle lediglich eine Sitzung gleichzeitig starten können. Bei der Bestellung eines Lizenzcodes für einen Terminal-Server ist neben der Computer-ID auch die Angabe der gewünschten Benutzerzahl erforderlich. Aus dem Lizenzcode ergibt sich dann die Zahl der Benutzer, die das Programm **gleichzeitig** auf dem Terminal-Server starten können. Eine Erhöhung der Benutzerzahl auf dem Terminal-Server ist zu einem späteren Zeitpunkt jederzeit möglich. In diesem Fall erhalten Sie einen Lizenzcode für die neue Gesamtanzahl der Benutzer. Hierbei wird Ihnen die Differenz zwischen alter und neuer Benutzerzahl in Rechnung gestellt.

3 Erstellung einer DataCool-Anwendung

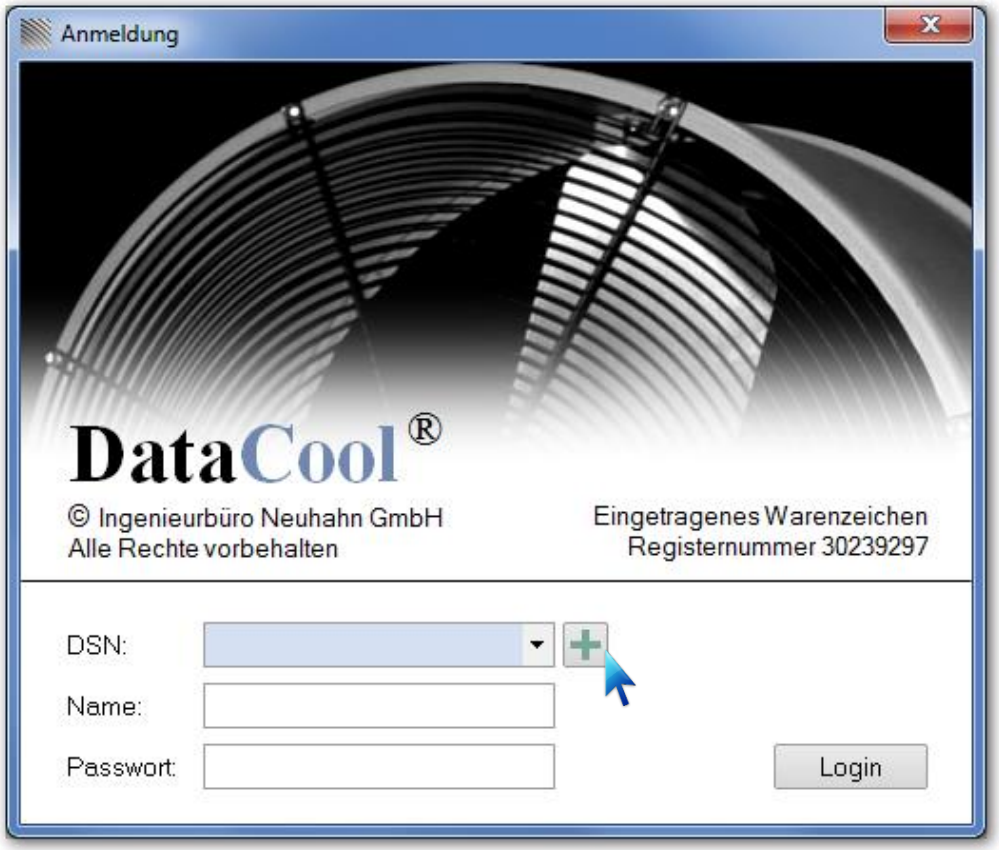
3.1 Anlage einer neuen Datenbank

Für jede DataCool-Anwendung wird eine separate SQL-Datenbank benötigt. Sie können beliebig viele Datenbanken gleichzeitig betreiben. **Das Anlegen einer neuen Datenbank muss auf dem Server-PC erfolgen.**

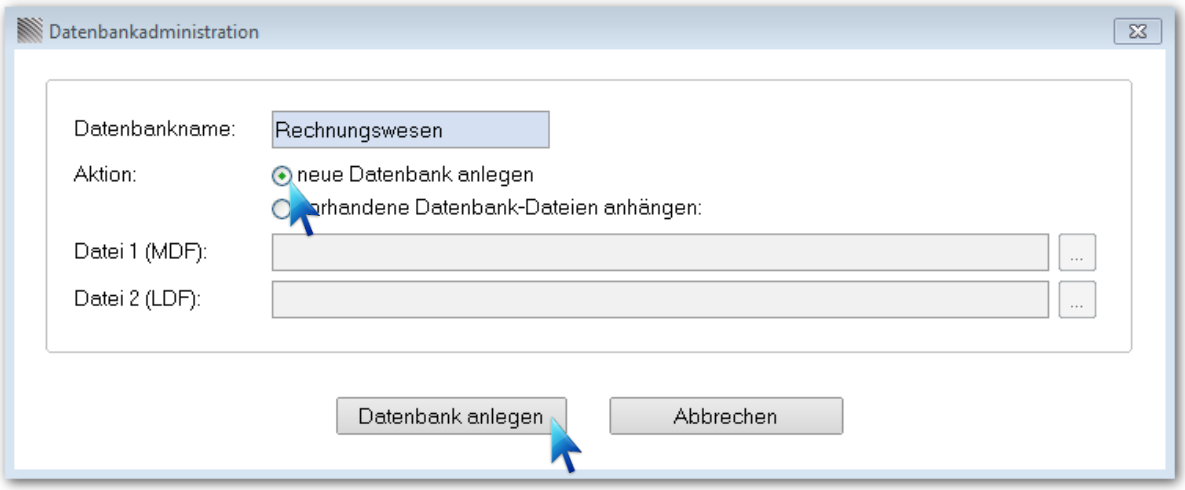
DataCool speichert neben den Daten auch die Programmstruktur Ihrer Anwendung in der SQL Datenbank. Hierzu wird eine Reihe von Systemtabellen benötigt, die alle mit der Bezeichnung „SYS_“ beginnen. Die Systemtabellen bilden zusammen das sogenannte „**Repository**“. Beim erstmaligen Aufruf einer neuen Datenbank wird das Repository automatisch angelegt.

Die nachfolgend beschriebene Methode zum Anlegen einer neuen Datenbank gilt nur wenn Sie mit der Einzelplatzversion von DataCool arbeiten. Im Netzbetrieb verwenden Sie stattdessen das Tool „SQL Server Management Studio“ auf dem Server-Computer. Einzelheiten hierzu erfahren Sie in Kapitel 9.1.

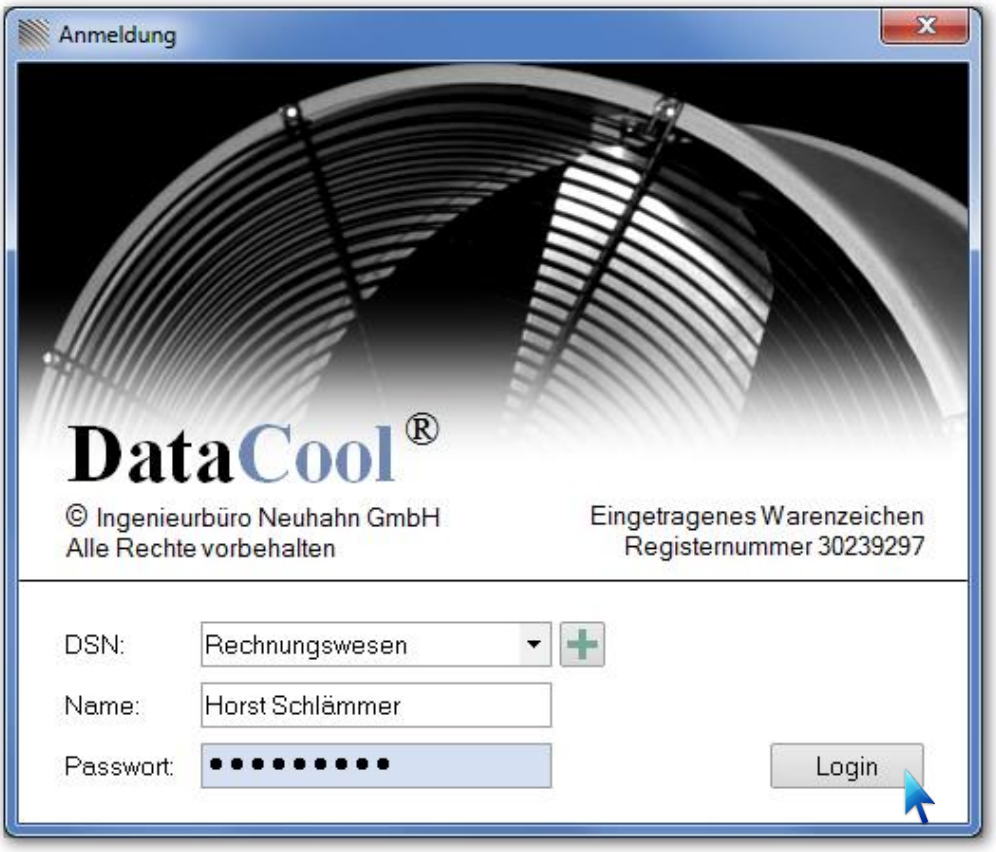
Klicken Sie nach dem Start von DataCool auf das grüne „+“ Symbol:



Vergeben Sie nun einen Namen für die neue Datenbank und klicken Sie anschließend auf den Button ***Datenbank anlegen***:



Zusammen mit der Datenbank wird auch ein DSN-Eintrag (Data Source Name) erstellt. Der DSN-Eintrag ermöglicht die Auswahl der Datenbank im Login-Bildschirm von DataCool:



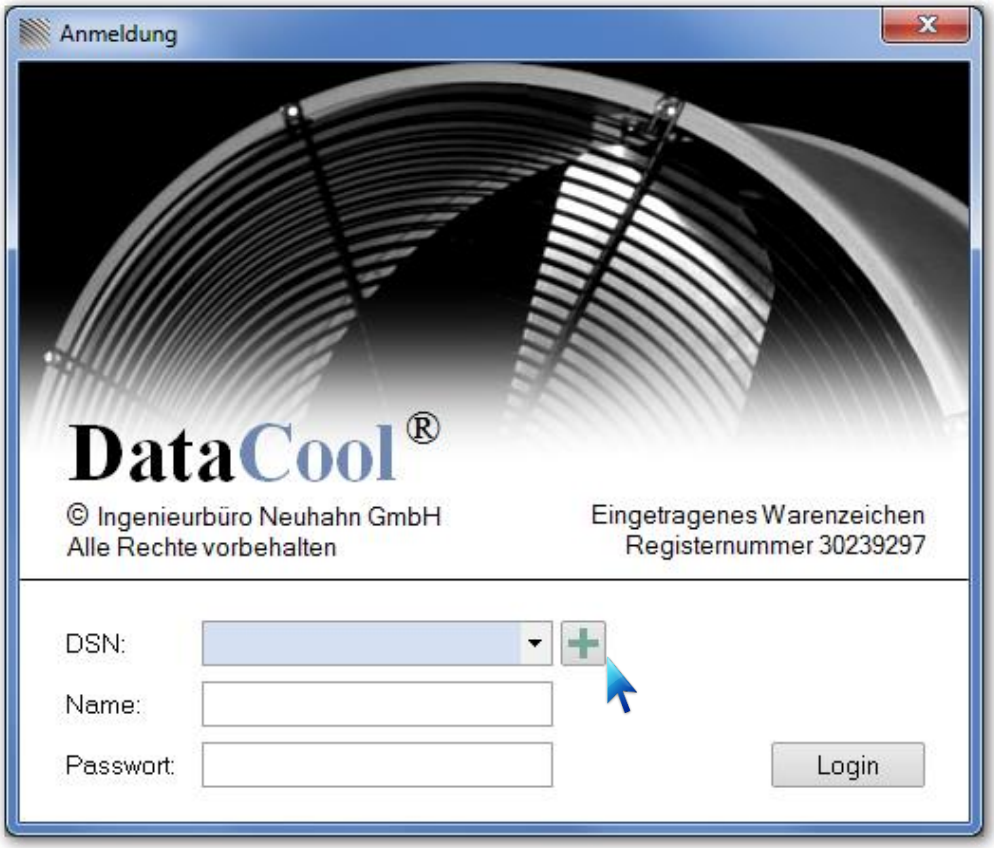
Bei der erstmaligen Anmeldung können Sie Benutzername und Passwort frei vergeben. Der neue Benutzer wird automatisch zum Administrator der neuen DataCool Anwendung.

3.2 Anhängen einer bestehenden Datenbank

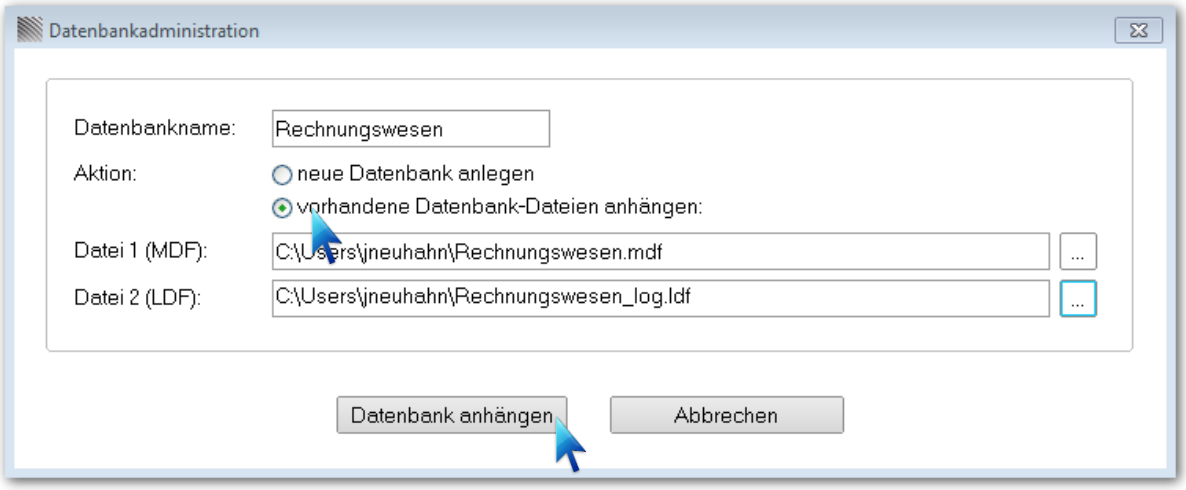
Mit der Funktion „Datenbank anhängen“ können Sie eine komplette Datenbank (Programmstruktur und Nutzerdaten) von einem Rechner auf einen anderen Rechner übertragen. **Das Anhängen einer Datenbank muss auf dem Server-PC erfolgen.**

Die nachfolgend beschriebene Methode zum Anhängen einer neuen Datenbank gilt nur wenn Sie mit der Einzelplatzversion von DataCool arbeiten. Im Netzbetrieb verwenden Sie stattdessen das Tool „SQL Server Management Studio“ auf dem Server-Computer. Einzelheiten hierzu erfahren Sie in Kapitel 9.2.

Klicken Sie hierzu nach dem Start von DataCool auf das grüne „+“-Symbol:



Bevor Sie die Datenbank anhängen können, müssen Sie die beiden Datenbankdateien mit der Endung MDF und LDF in Ihr **Profilverzeichnis** kopieren. Wählen Sie dann die Option „vorhandene Datenbank-Dateien anhängen“ und klicken Sie auf den Button ***Datenbank anhängen***:



3.3 Die Übungsdatenbank „DataCool Tutorial“

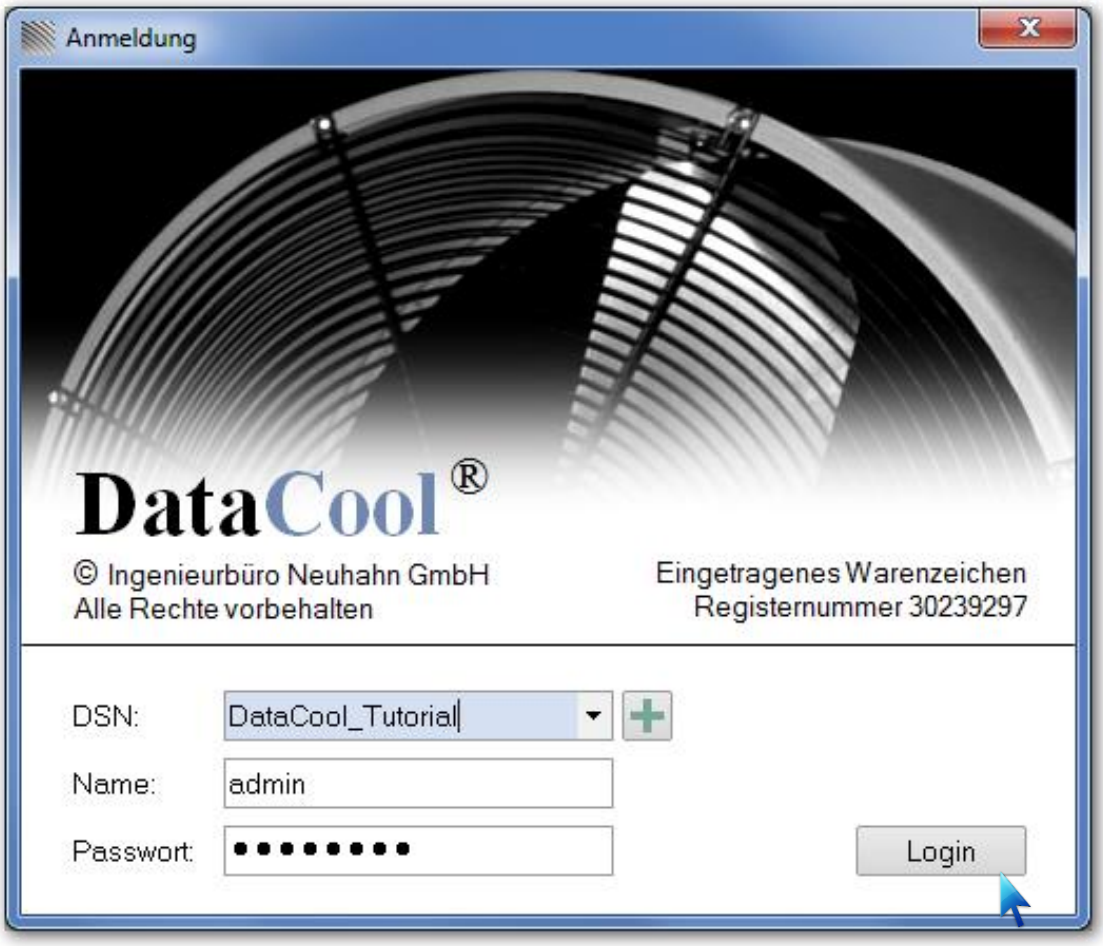
Beim Einzelplatz-Setup wird automatisch das **DataCool Tutorial** mitinstalliert. Es handelt sich hierbei um ein **komplettes Rechnungsprogramm** für die Übungsfirma „AquaCool“. Anhand der fertigen Anwendung können Sie sich schnell einen Überblick über die Leistungsfähigkeit von DataCool verschaffen.

Das Programm beinhaltet folgende Funktionen:

- Adressverwaltung
- Artikelverwaltung
- Druck von Angeboten
- Druck von Auftragsbestätigungen
- Druck von Lieferscheinen
- Druck von Rechnungen
- Druck von Gutschriften
- automatisches 3-stufiges Mahnwesen
- Buchhaltungsmodul mit Teilzahlungsfunktion

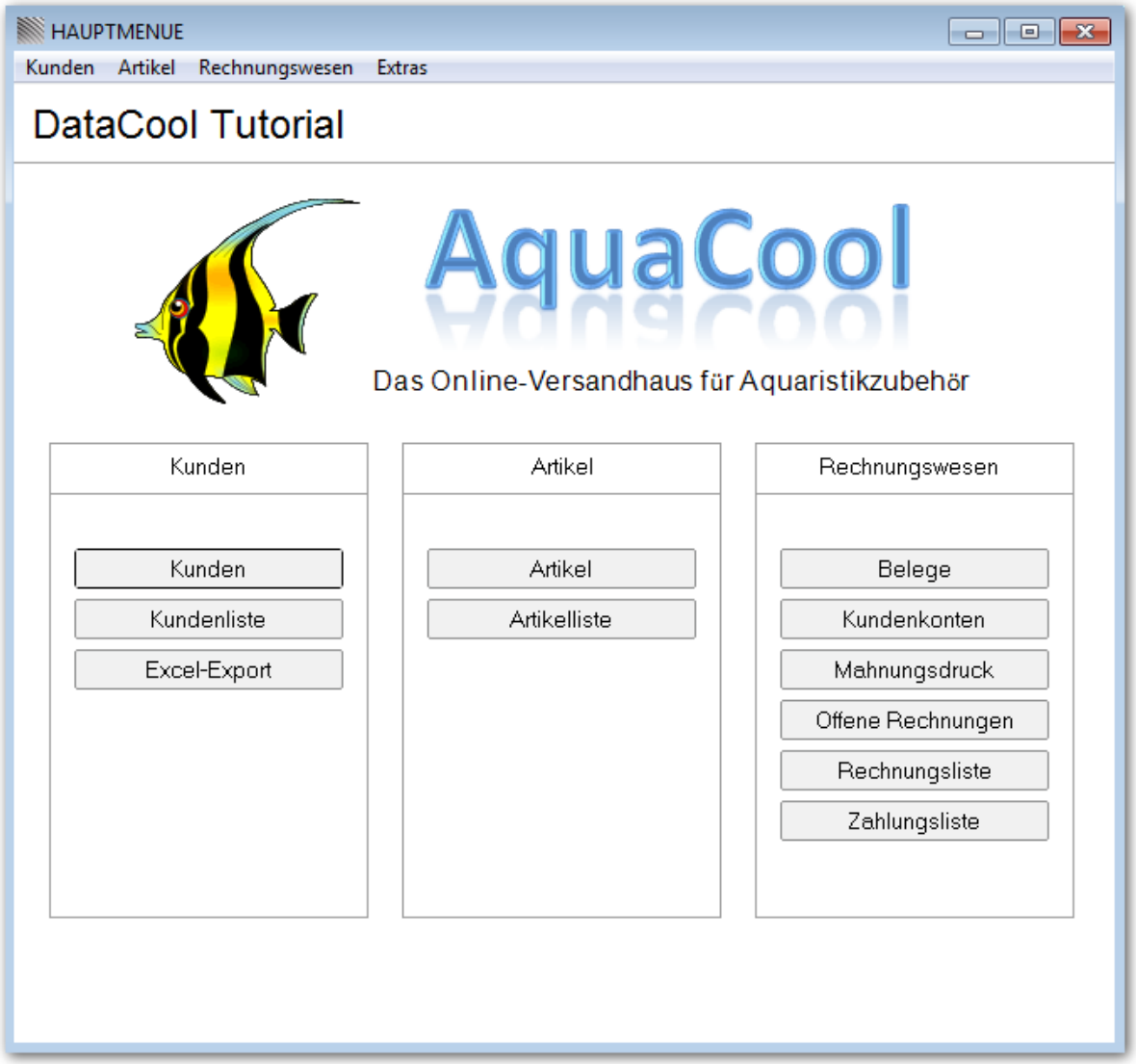
Sie können das Rechnungsprogramm aus dem DataCool Tutorial gerne in Originalform oder als Teil eigener Anwendungen weitervertreiben. Separate Lizenzkosten fallen hierbei nicht an.

Bitte starten Sie das Tutorial wie folgt:



Sobald Sie die DSN „DataCool_Tutorial“ auswählen werden Benutzername (**admin**) und Kennwort (**DataCool**) automatisch eingetragen.

Klicken Sie anschließend auf den Login-Button. Die Benutzeroberfläche der fertigen Anwendung können Sie per Doppelklick auf das Dokument „HAUPTMENUE“ starten:



Viel Spaß beim Ausprobieren!

3.4 Programmstart und Befehlszeilenparameter

Normalerweise benötigen Sie drei Angaben zum Start einer DataCool-Anwendung:

- DSN (Einzelplatzbetrieb) oder Servername und Datenbankname (Netzbetrieb)
- Benutzername
- Passwort

Diese Informationen werden in der Login-Maske abgefragt. Mit Hilfe von Befehlszeilenparametern können Sie eine oder mehrere Angaben vorbelegen:

```
DataCool.exe [/dsn="Data Source Name (Einzelplatzbetrieb)"]
             [/srv="SQL-Server-Instanz (Netzbetrieb)"]
             [/db="Datenbankname (Netzbetrieb)"]
             [/usr="Benutzer"]
             [/pwd="Passwort"]
             [/pic="Bildpfad"]
             [/netlogon]
```

Die eckigen Klammern bedeuten, dass es sich um optionale Angaben handelt.

Ist lediglich die Angabe **/dsn** vorhanden, so ist der Datenbankname vorausgewählt und der Cursor steht beim Programmstart direkt in der Benutzerzeile. Diese Einstellung wird im Einzelplatzbetrieb benutzt.

Im Netzbetrieb ist der Parameter **/srv** nötig. Geben Sie den Computernamen des SQL-Servers und ggf. einen vorhandenen Instanznamen an (z.B. SQL01\DATACOOL). Wenn Sie den Server im Netz mit /srv bestimmt haben, dann können Sie mit dem Parameter **/db** auch den Namen der Datenbank festlegen. Fehlt der Parameter /db, dann werden alle verfügbaren Datenbanken zur Auswahl angeboten.

Ist nur der Parameter **/usr** vorhanden, so ist lediglich die Eingabe des Passwortes in der Startmaske erforderlich.

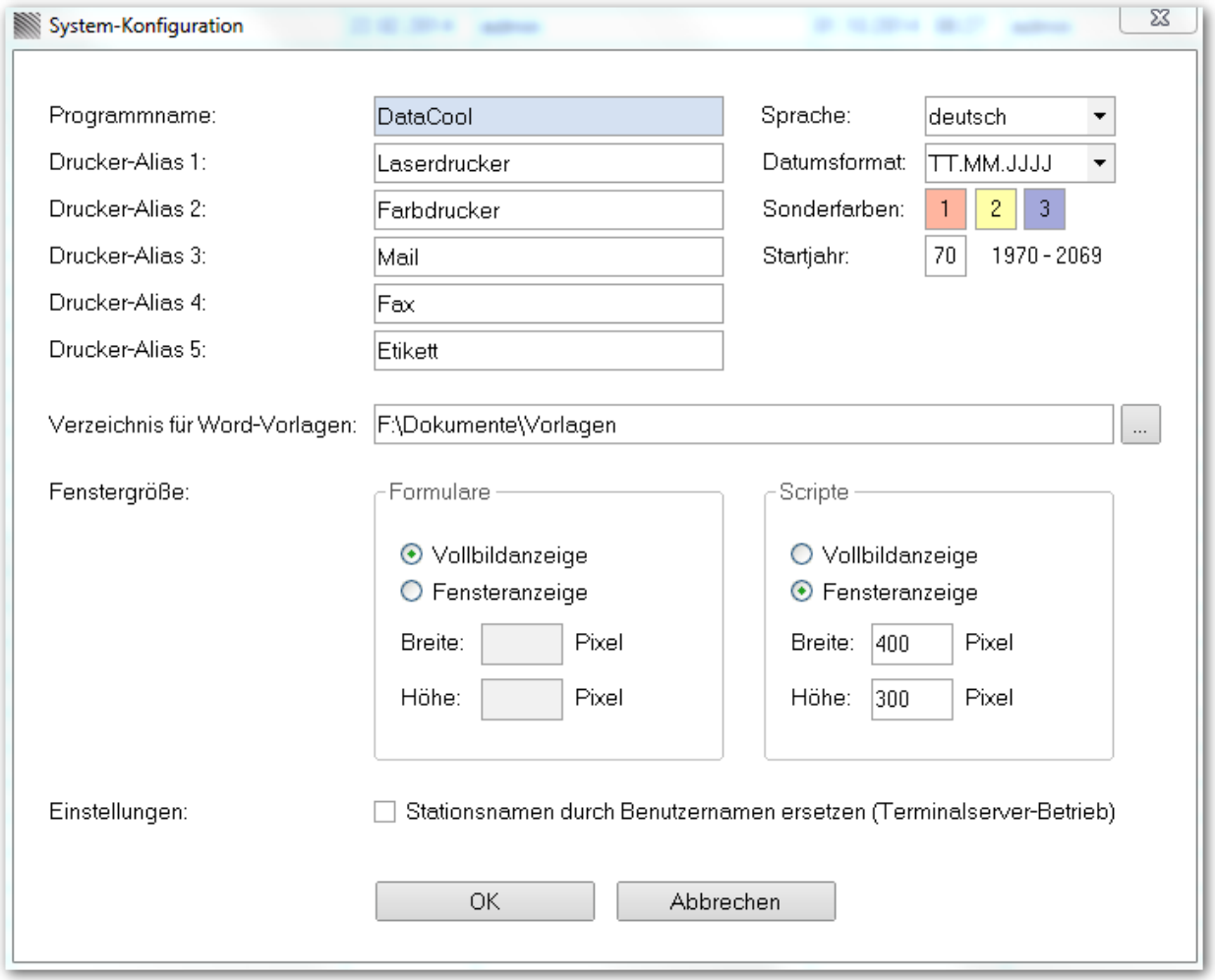
Sind die Parameter **/usr** und **/pwd** in der Befehlszeile vorhanden, so startet die Anwendung direkt ohne Anzeige der Login-Maske.

Mit Hilfe des Parameters **/pic** kann die Grafik in der Login-Maske ausgetauscht werden. Hierzu muss eine BMP-Datei mit Pfadangabe übergeben werden. Durch Austausch der Startgrafik können Sie Ihrer fertigen Anwendung ein eigenes Branding geben. Die BMP-Datei benötigt die Abmessungen 500 x 400 Pixel.

Der Parameter ***/netlogon*** kann in Netzwerken mit einer Windows Domäne benutzt werden. In diesem Fall wird der an der Domäne angemeldete Benutzer direkt von DataCool übernommen. Die Eingabe eines Kennwortes ist in dann nicht erforderlich. Bitte beachten Sie, dass in der Benutzerverwaltung von DataCool ein Benutzername angelegt werden muss, der mit dem Windows Login-Name übereinstimmt.

3.5 Systemkonfiguration

Nach dem ersten Aufruf Ihrer neuen Anwendung sollten Sie zunächst einige grundlegende Einstellungen vornehmen. Bitte rufen Sie hierzu in der Menüzeile die Option **Konfiguration ⇨ System** auf:



Allgemeine Einstellungen

Programmname: Hier können Sie den Namen Ihrer Anwendung festlegen. Der Programmname wird bei verschiedenen Systemmeldungen angezeigt.

Druckeralias 1-5: Mit Hilfe von Aliasnamen können Sie in DataCool-Scripten auf unterschiedliche Drucker zugreifen. Die Zuordnung zwischen Aliasname und Treiber erfolgt in der Arbeitsplatzkonfiguration (siehe Kapitel 3.7). Auf diese Weise können Sie Druckernamen aufgabenspezifisch definieren (z.B. Farbdrucker, Laserdrucker, Etikettendrucker) und für jede Station im Netzwerk den nächstgelegenen passenden Drucker zuordnen.

Verzeichnis für Word-Vorlagen: DataCool besitzt eine Schnittstelle zu Microsoft Word. Mit Hilfe dieser Schnittstelle können Sie aus jedem Formular heraus Briefe erstellen. Hierzu wird eine Word-Vorlage ausgewählt. In der Vorlage können Sie Platzhalter definieren, die von DataCool mit Feldinhalten ersetzt werden. Das Vorlage-Verzeichnis ist der Ordner, in dem Ihre Vorlagedateien (*.dot) abgelegt sind. Weitere Einzelheiten zur Word-Schnittstelle finden Sie im Anwenderhandbuch.

Sprache: Hier können Sie die Benutzersprache einstellen.

Datumsformat: Diese Einstellung ist für künftige Versionen vorbehalten. Als Datumsformat steht derzeit nur TT.MM.JJJJ zur Verfügung. Hinweis: In den Regions- und Sprachoptionen von Windows muss als kurzes Datumsformat auch TT.MM.JJJJ eingestellt sein, damit DataCool korrekt arbeitet.

Sonderfarben: Hier können Sie Farben für Eingabefelder definieren. Die Einstellungen wirken sich auf alle Formulare aus. Die Farben 1, 2 und 3 sind Hervorhebungsfarben und die Farbe F dient zur Markierung des aktuellen Eingabefeldes.

Startjahr: Zweistellige Jahreszahlen werden von DataCool automatisch in vierstellige Jahreszahlen umgewandelt. Das Startjahr bestimmt hierbei den zugrundegelegten Zeitraum. Die Einstellung „70“ bedeutet, dass zweistellige Jahreszahlen in den Zeitraum von 1970 bis 2069 fallen.

Fenstereinstellungen

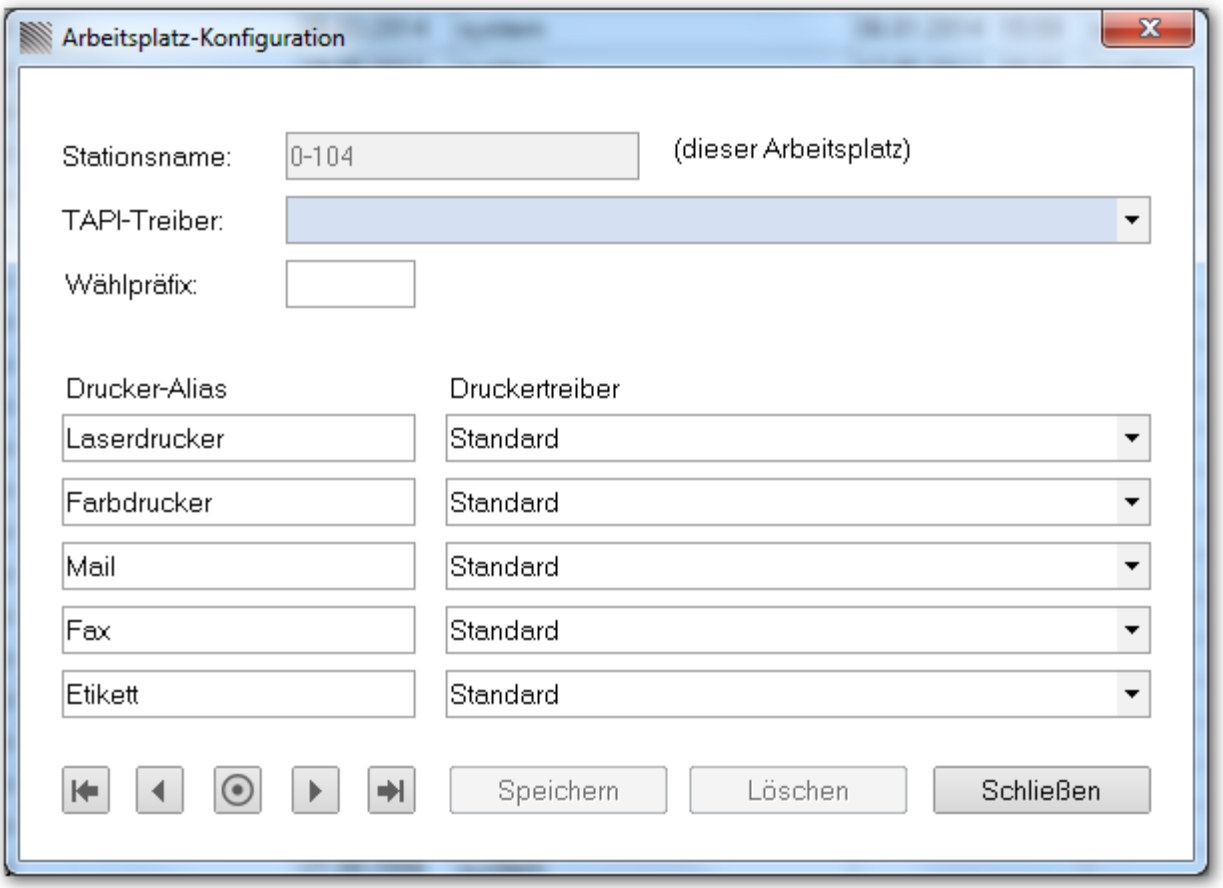
An dieser Stelle bestimmen Sie, ob ein Dokument als **Vollbild** oder als **Fenster** angezeigt wird. Formulare werden in der Regel als Vollbild angezeigt, während Scripte überwiegend den Fenstermodus verwenden. Die Einstellungen gelten als Vorgabewerte und können in jedem Dokument individuell abgeändert werden.

Einstellungen

Die Option „Stationsnamen durch Benutzernamen ersetzen“ ist für den Terminal-Server-Betrieb gedacht. Jeder Rechner im lokalen Netzwerk besitzt normalerweise einen eindeutigen Computernamen. DataCool benutzt diesen Stationsnamen zur Identifikation der unterschiedlichen Arbeitsplätze. Bei einer Terminalserver-Installation funktioniert diese Technik allerdings nicht, da alle Benutzer auf demselben Rechner arbeiten. Aus diesem Grund müssen Sie hier den Stationsnamen durch den Benutzernamen ersetzen.

3.6 Arbeitsplatzkonfiguration

Unter **Konfiguration ⇨ Arbeitsplatz** können Sie individuelle Einstellungen für jede Station im Netz definieren. DataCool legt für jeden Netzwerkarbeitsplatz automatisch einen Konfigurationsdatensatz an. Als eindeutiges Merkmal dient hierbei der Computername der Station. Bei einer Terminal-Server-Installation kann auch der Benutzername als eindeutiges Merkmal dienen, wenn die entsprechende Option in der Systemkonfiguration aktiviert wurde.



Stationsname: Computername im Netzwerk oder Benutzername auf einem Terminal-Server. Beim erstmaligen Aufruf von DataCool an einer Station wird automatisch ein Konfigurationsdatensatz angelegt.

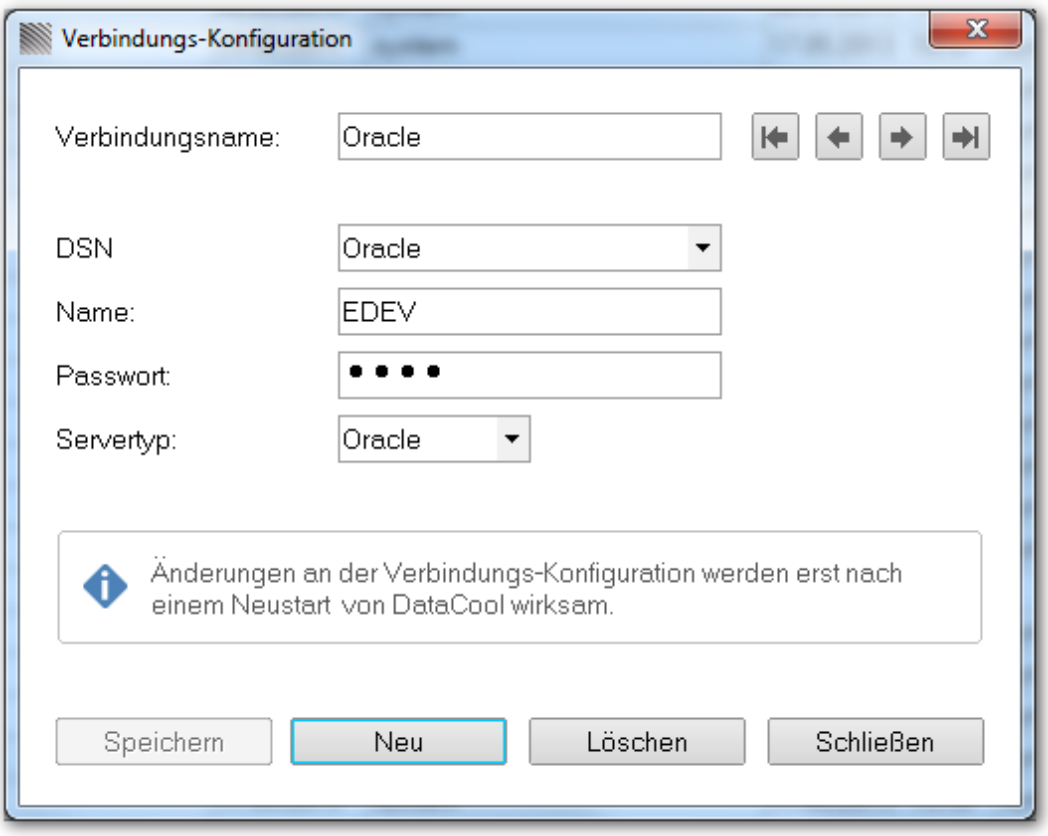
TAPI-Treiber: TAPI steht für **T**elephony **A**pplication **P**rogramming **I**nterface. Mit Hilfe eines TAPI-Treibers kann DataCool automatisch Telefonnummern wählen. Einzelheiten zur Installation und Konfiguration des TAPI-Treibers entnehmen Sie bitte den Unterlagen Ihrer Telefonanlage.

Wählpräfix: An dieser Stelle können Sie eine Ziffer für die Amtsholung definieren. Der Wählpräfix wird allen Telefonnummern bei der automatischen Wahl vorangestellt.

Drucker-Alias:	Drucker-Aliasnamen dienen zur Auswahl von Druckern in Scripten. Die Festlegung der Aliasnamen erfolgt in der Systemkonfiguration.
Drucker-Treiber:	In der Arbeitsplatzkonfiguration wird jedem Aliasnamen ein Druckertreiber zugeordnet. Die Einstellung „Standard“ bedeutet, dass der momentan eingestellte Standarddrucker von Windows benutzt wird. Sollte ein hier eingestellter Drucker nicht mehr verfügbar sein (z.B. weil er deinstalliert wurde), so erfolgt die Druckerausgabe ebenfalls auf dem Standarddrucker von Windows.

3.7 Verbindungskonfiguration

Mit Hilfe von „Verbindungen“ erhalten Sie Zugriff auf fremde Datenbanken. Unter **Konfiguration ⇨ Verbindungen** können Sie die nötigen Einstellungen vornehmen:



Verbindungsname: Legen Sie einen Kurznamen für den Zugriff auf das fremde System fest. Der Bezeichner dient in SQL-Befehlen zur Auswahl der Connection (z.B.: select * from Kunde@**Oracle**).

DSN: Für den Zugriff auf die fremde Datenquelle wird ein ODBC-Treiber verwendet. Bitte wählen Sie an dieser Stelle den zuvor angelegten **D**ata **S**ource **N**ame aus.

Name: Benutzername zur Anmeldung an der fremden Datenbank.

Passwort: Passwort zur Anmeldung an der fremden Datenbank.

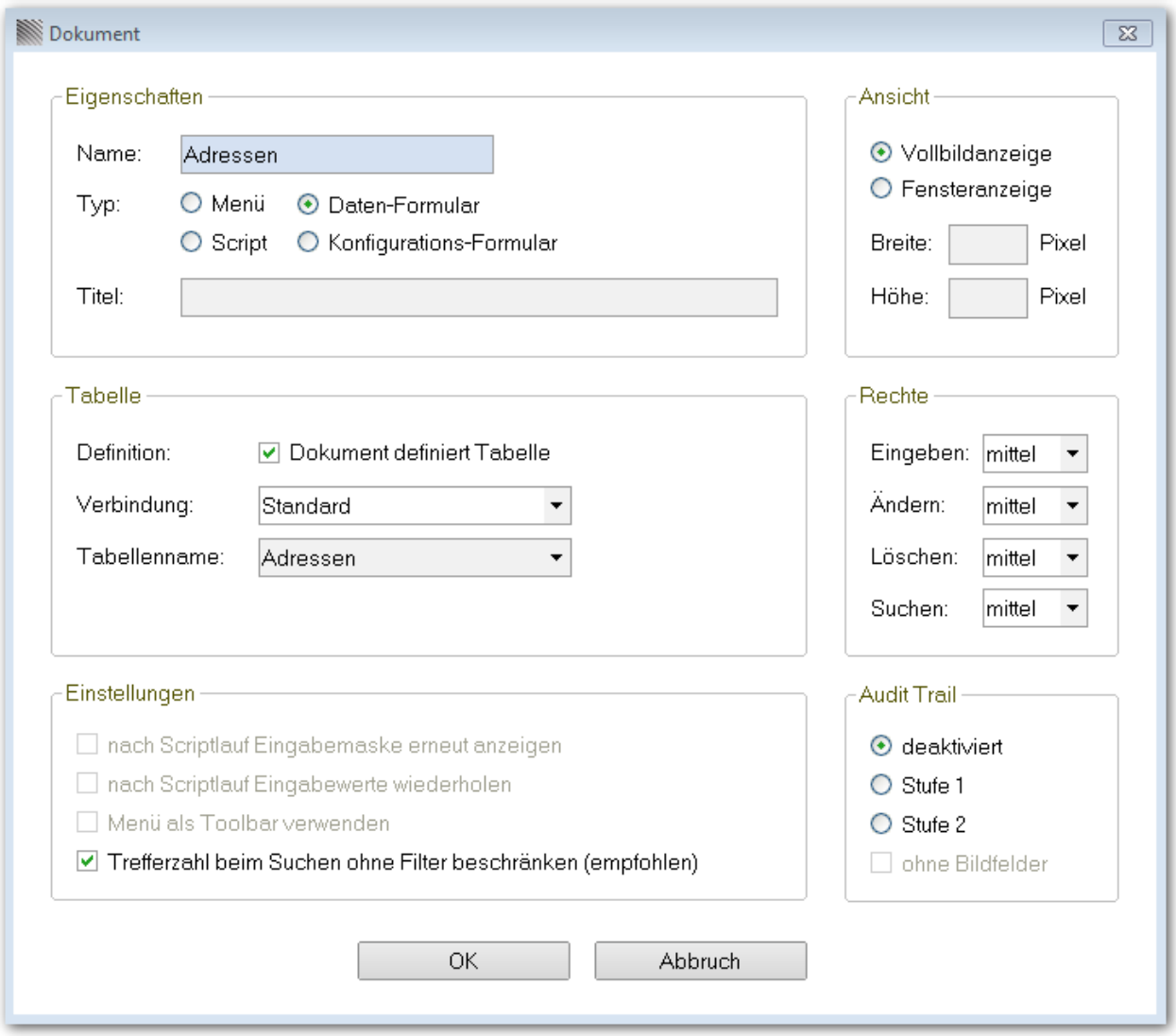
Servertyp: Trotz der Standardisierung der SQL-Abfragesprache gibt es eine Reihe von Unterschieden zwischen den Datenbanken der verschiedenen Hersteller. Bitte wählen Sie deshalb einen der nachfolgenden Hersteller aus: Microsoft, Oracle, MySQL.

Sie müssen DataCool beenden und neu starten bevor Änderungen an der Verbindungskonfiguration wirksam werden. Bitte stellen Sie außerdem sicher, dass der angegebene Data Source Name (DSN) in den ODBC-Einstellungen jedes Arbeitsplatzes eingerichtet wurde.

4 Formulare und Verknüpfungen

4.1 Erstellung eines Formulars

Bitte wählen Sie die Option ***Formulare*** im Hauptmenü von DataCool. Klicken Sie anschließend auf den Button ***Neu*** oder drücken Sie die Taste Einfg um ein neues Formular zu erstellen. Im nachfolgenden Fenster können Sie nun die Formulareigenschaften definieren:



Eigenschaften

Name: Hier können Sie den Namen des Formulars eingeben. Erlaubt sind hierbei die Zeichen zwischen 0 und 9, sowie das Alphabet. Leerstellen werden zu _ gewandelt, Umlaute zu ae, oe, ue und ß zu ss.

Typ: DataCool kennt vier verschiedene Dokumentarten:

Ein **Menü** ist ein einfaches Formular dem keine SQL-Tabelle zugrunde liegt. Mit Hilfe von Menüs können Anwender die Funktionen Ihrer Applikation steuern.

Ein **Daten-Formular** ermöglicht das Eingeben, Ändern oder Löschen von Datensätzen in einer SQL-Tabelle. Darüberhinaus stehen dem Anwender diverse Suchfunktionen zur Verfügung. Bei der Erstellung eines Daten-Formulars wird von DataCool automatisch eine passende SQL-Tabelle erstellt.

Ein **Konfigurations-Formular** ist ein Formular, dem nur ein einzelner Datensatz zugrunde liegt. Dieser Datensatz wird beim Aufruf des Formulars automatisch angezeigt.

Mit einem **Script** können Listen und automatische Abläufe erstellt werden. Auch ein Script kann ein Formular besitzen, wenn es vor dem Start zusätzliche Benutzereingaben erfordert.

Titel: Überschrift in der Titelleiste von Scripten und Menüs.

Ansicht

Sie können zwischen **Vollbild-** und **Fensteranzeige** wählen. Bei der Fensteranzeige müssen Sie die gewünschte Breite und Höhe des Fensters festlegen. Die Größenangabe erfolgt in der Einheit „Pixel“ (=Bildschirmpunkte).

Tabelle

Im Normalfall ist die Option „**Dokument definiert Tabelle**“ aktiviert. Dies bedeutet, dass DataCool zusammen mit der Erstellung des Formulars automatisch eine passende SQL-Tabelle anlegt. Bei nachträglichen Änderungen am Formular wird die SQL-Tabelle durch spezielle Befehle automatisch mit dem Formular synchronisiert.

Mit DataCool können Sie jedoch auch Formulare für fremde SQL-Tabellen erstellen. In diesem Fall müssen Sie **Verbindungs-** und **Tabellenname** angeben. Fremde SQL-Tabellen werden von DataCool nicht verändert.

Rechte

Sie können jedem DataCool Benutzer eine der nachfolgenden Berechtigungsstufen zuweisen: **niedrig**, **mittel**, **hoch** und **admin**. In den Formulareigenschaften wird hierbei festgelegt, welche Stufe der Benutzer mindestens besitzen muss, damit er eine bestimmte Aktion ausführen kann.

Einstellungen

Die Option „**Eingabemaske erneut anzeigen**“ bestimmt, dass das Eingabeformular eines Scriptes nach Ablauf des Scriptes erneut angezeigt werden soll. In diesem Fall hat der Anwender die Möglichkeit das Script mit anderen Startparametern erneut auszuführen.

Mit der Option „**Eingabewerte wiederholen**“ kann die jeweils letzte Benutzereingabe als Vorgabe in die Eingabemaske eines Scriptes eingetragen werden.

Die Option „**Menü als Toolbar verwenden**“ wird im Kapitel 5.2 näher erläutert.

Die Option „**Trefferzahl beim Suchen ohne Filter beschränken**“ ist standardmäßig aktiviert. Diese Einstellung bewirkt, dass bei Suchvorgängen jeweils nur die letzten **100 Treffer** geladen werden, **wenn kein Suchfilter aktiv ist**. Sobald ein Suchfilter aktiv ist werden stets alle Suchergebnisse geladen. Diese Funktion dient dazu die Ladezeiten bei Suchergebnissen und beim Start von Formularen zu optimieren. Wenn Sie den Haken entfernen, dann wird die Begrenzung der Suchergebnisse abgeschaltet. In der Praxis gibt es jedoch trotzdem eine Beschränkung auf 100.000 Datensätze.

Wir empfehlen die Beschränkung aktiv zu lassen, da beim Suchen mit einem Filterkriterium ohnehin stets alle Treffer angezeigt werden. Lediglich bei Formularen, die eine geringe Gesamtanzahl von einigen Hundert oder Tausend Datensätzen haben kann die Abschaltung der Begrenzung sinnvoll sein.

Audit Trail

Bei aktivierter **Audit Trail** Funktion wird jede Änderung an einem Datensatz vom System automatisch dokumentiert. Auf diese Weise kann die Historie eines Datensatzes nachvollzogen werden. Bei **Stufe 1** werden folgende Informationen protokolliert: Benutzer, Datum, Uhrzeit, Station. Bei **Stufe 2** wird zusätzlich der Grund für die Änderung abgefragt und dokumentiert. Sobald die Audit Trail Funktion eingeschaltet ist, wird vom System bei jeder Änderung eine Kopie des angezeigten Datensatzes in der Systemtabelle SYS_Trail abgespeichert.

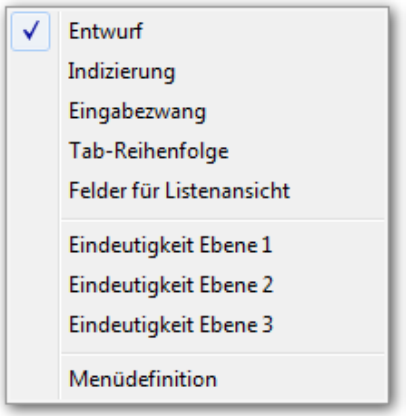
Mit der Audit Trail Funktion können Sie Anwendungen erstellen, die den Anforderung von GAMP 4 (Good Automated Manufacturing Practice) gerecht werden.

Durch die Protokollierung von Datensatzänderungen können große Datenmengen anfallen, die in der Systemtabelle SYS_Trail gespeichert werden müssen. Die Audit Trail Funktion sollten Sie daher nur für solche Formulare einschalten, in denen eine Dokumentation von Änderungen wirklich benötigt wird. Außerdem können Sie bei Bedarf die Option „**ohne Bildfelder**“ aktivieren um Speicherplatz zu sparen. Die Inhalte von Bildfeldern werden dann nicht im Audit Trail abgespeichert.

Die nächsten Schritte

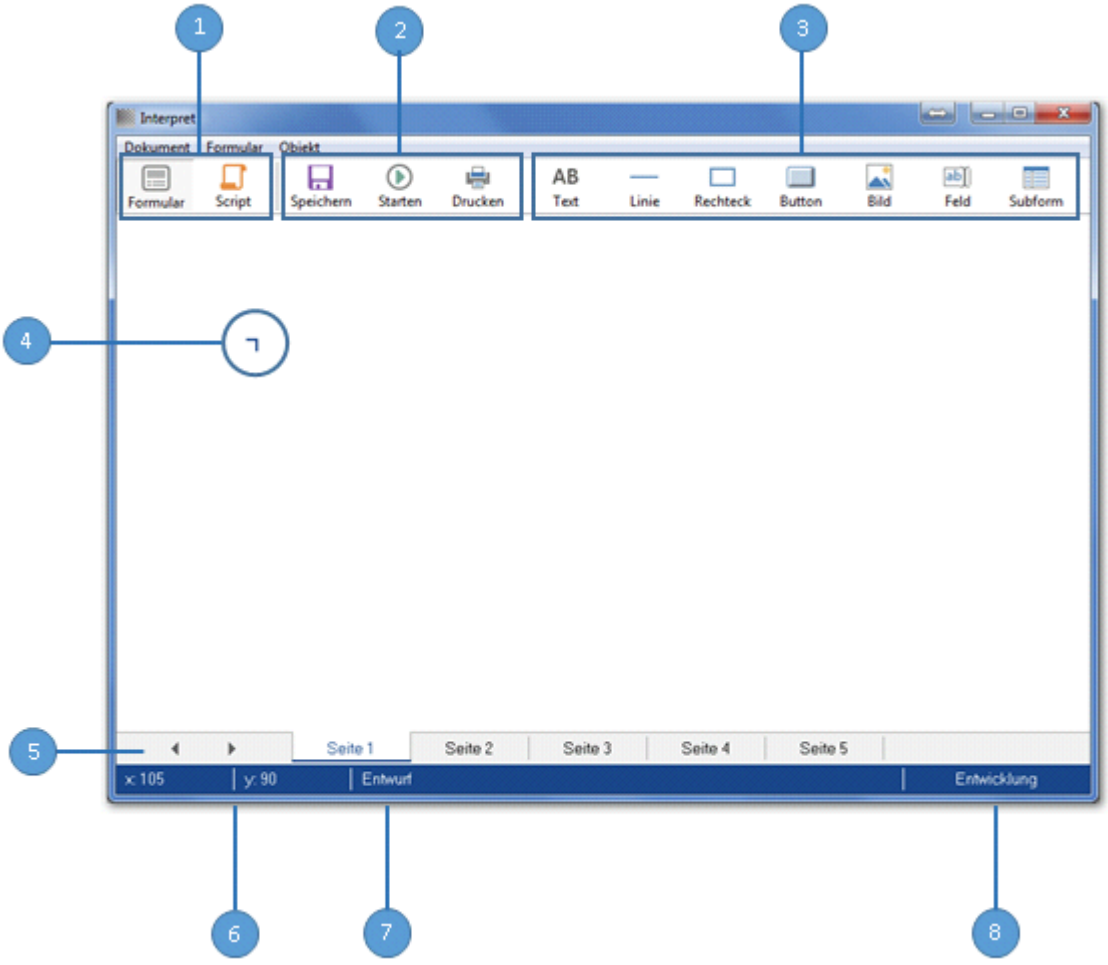
Nachdem die Formulareigenschaften festgelegt sind, können Sie mit der Gestaltung der Eingabemaske beginnen. Der Formulareditor verfügt hierzu über verschiedene Modi, die in den folgenden Kapiteln beschrieben werden.

4.2.1 Formular: Entwurf



Der Entwurfsmodus dient zur Gestaltung der Eingabemaske. Nach der Festlegung der Formulareigenschaften befinden Sie sich automatisch in diesem Modus.

Ein Formular besteht aus statischen und dynamischen Elementen. Statische Elemente sind Text, Linien und Rechtecke. Dynamische Elemente sind Eingabefelder, Buttons oder Bildschaltflächen. Die nachfolgende Abbildung zeigt die Bedienungselemente des Editors:



1: Jedes Dokument besteht aus einer Eingabemaske und einem dazugehörigen Script. Die beiden Buttons dienen zur Umschaltung zwischen Formular- und Scripteditor.

2: Grundlegende Bearbeitungsfunktionen:
- Speichern des Dokumentes
- Start des Dokumentes im Anwendermodus
- Drucken der Formulardefinition

3: Schaltflächen zum Hinzufügen eines Objektes.
Das neue Objekt wird an der aktuellen Cursorposition platziert.

4: Aktuelle Cursorposition

5: Ab der Version 2010 kann jedes Formular bis zu 5 Bildschirmseiten umfassen. Die Bildschirmseiten werden nun als Registertabs am unteren Bildschirmrand dargestellt. Auf diese Weise können Sie schnell per Mausklick ein bestimmtes Register ansteuern. Im Entwicklungsmodus können Sie den einzelnen Registern individuelle Namen vergeben. Klicken Sie hierzu mit der rechten Maustaste auf die Registerbezeichnung. Register mit dem Namen „**noexist**“ werden zur Laufzeit nur dem Administrator anzeigt.

6: x- und y-Koordinate des Cursors in der Maßeinheit Pixel (Bildschirmpunkte)

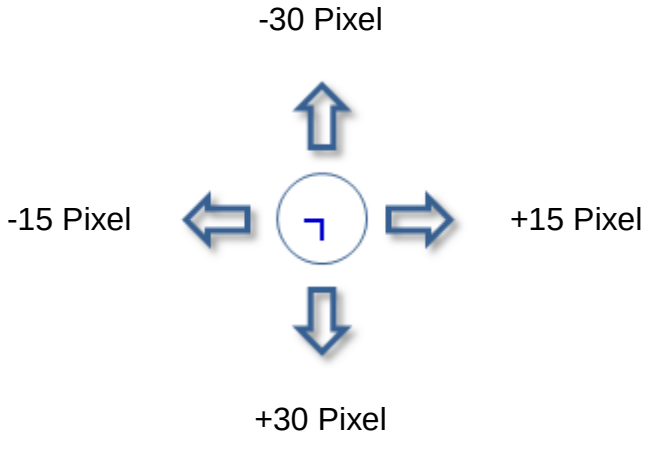
7: Aktiver Bearbeitungsmodus des Formulareditors

8: Ab der Version 2010 zeigt die Statusbar am rechten Rand den Namen der Datenbank. Viele Entwickler arbeiten während der Programmierphase mit einer Kopie einer Live-Anwendung. Durch die Anzeige des Datenbanknamens können Sie nun stets sehen, welche Anwendungsversion gerade aktiv ist.

Ab der Version 2014 können Sie ein Bild als Formularhintergrund laden. Klicken Sie hierzu einfach mit der rechten Maustaste auf eine freie Stelle des Bildschirms und wählen Sie eine Bilddatei aus.

Neue Objekte hinzufügen

Den Modus „Hinzufügen“ erkennen Sie an der **blauen** Cursorfarbe. Bewegen Sie den Cursor an die gewünschte Stelle links unterhalb des neuen Objektes. Am einfachsten können Sie den Cursor mit den Pfeiltasten auf der Tastatur positionieren:



Mit den Pfeiltasten bewegen Sie den Cursor auf die jeweils nächstgelegene Rasterposition. Das Raster besitzt einen horizontalen Abstand von 15 Pixel und einen vertikalen Abstand von 30 Pixel. Mit Tab bzw. Shift-Tab bewegen Sie den Cursor auf die nächste Tabulatorposition. Mit der Tastenkombination Strg+Pfeiltaste springt der Cursor an den jeweiligen Bildschirmrand.

Natürlich können Sie den Cursor auch mit der Maus platzieren. Klicken Sie hierzu einfach auf die gewünschte Stelle. Der Cursor springt dann automatisch auf die nächstgelegene Rasterposition.

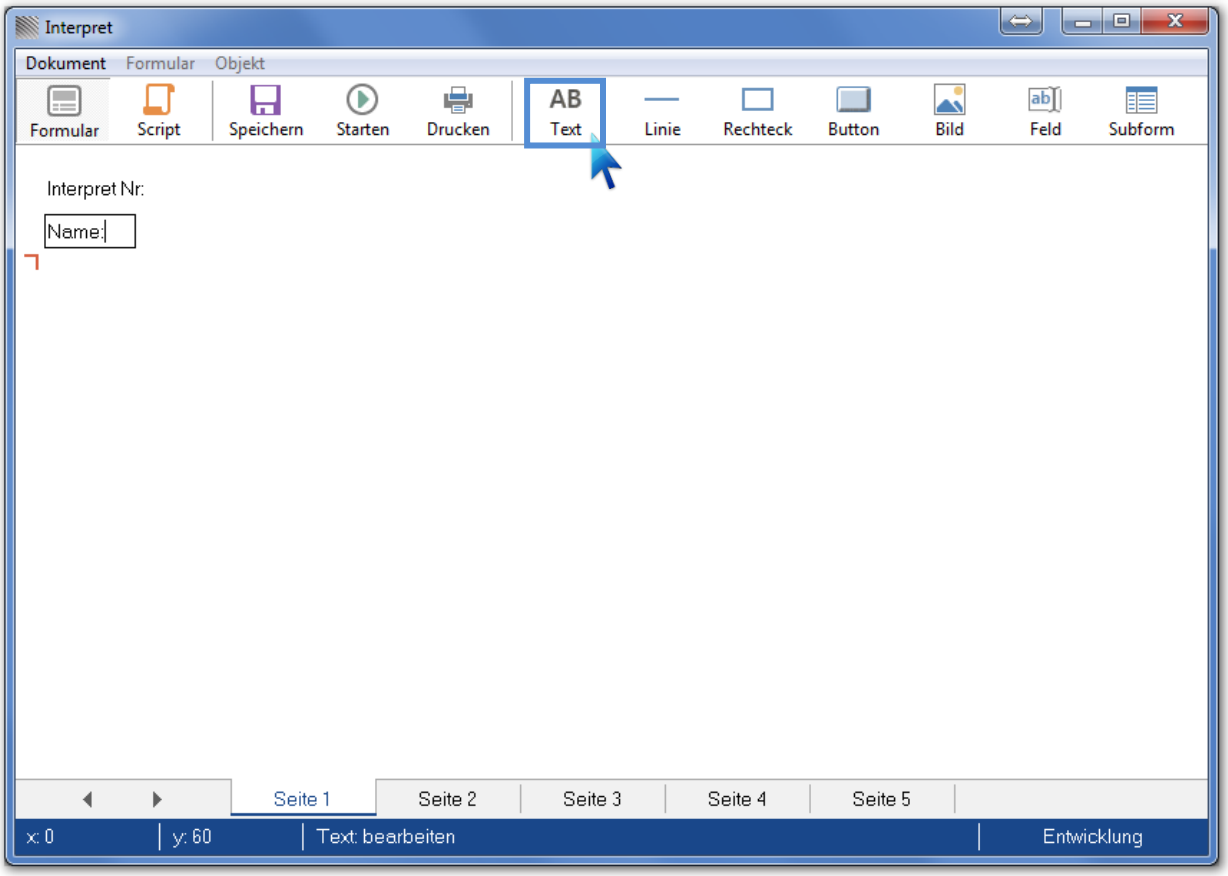
Bei Bedarf lässt sich der Cursor auch pixelgenau verschieben: Verwenden Sie hierzu die Tastenkombination Shift+Pfeiltaste.

Im Regelfall sollten Sie alle Objekte an den vordefinierten Rasterpositionen belassen. Größe und Lage der einzelnen Objekte sind optimal aufeinander abgestimmt. Sie erhalten dadurch im Handumdrehen ein harmonisches Erscheinungsbild für Ihre Formulare. Außerdem ist sichergestellt, dass alle Masken einer Anwendung den gleichen „Look“ aufweisen.

Nachdem der Cursor richtig platziert ist wählen Sie ein Objekt mit der Buttongruppe 3 aus der Toolbar. Die einzelnen Objekte und deren Eigenschaften werden auf den folgenden Seiten näher erläutert.

4.2.1.1 Objekt hinzufügen: Text

Ein neues Textobjekt wird auf folgende Weise erstellt: Bewegen Sie den Cursor an die gewünschte Stelle und wählen Sie den Button „Text“ aus der Toolbar. Alternativ dazu können Sie auch Strg+T drücken oder **einfach zu schreiben beginnen**.



Sobald das Objekt angelegt ist wechselt der Cursor in den Bearbeitungsmodus (Farbe rot) und Sie können den Text innerhalb des schwarzen Rahmens eingeben. Wenn der Text fertig ist drücken Sie Enter um die Bearbeitung des Textobjektes abzuschließen.

Text bearbeiten

Klicken Sie mit der rechten Maustaste auf den Text und wählen Sie „Bearbeiten“ aus dem Kontextmenü. Sie können den Bearbeitungsmodus auch mit der Tastenkombination Strg+B aktivieren wenn der Cursor links unter dem Text steht.

Text verschieben

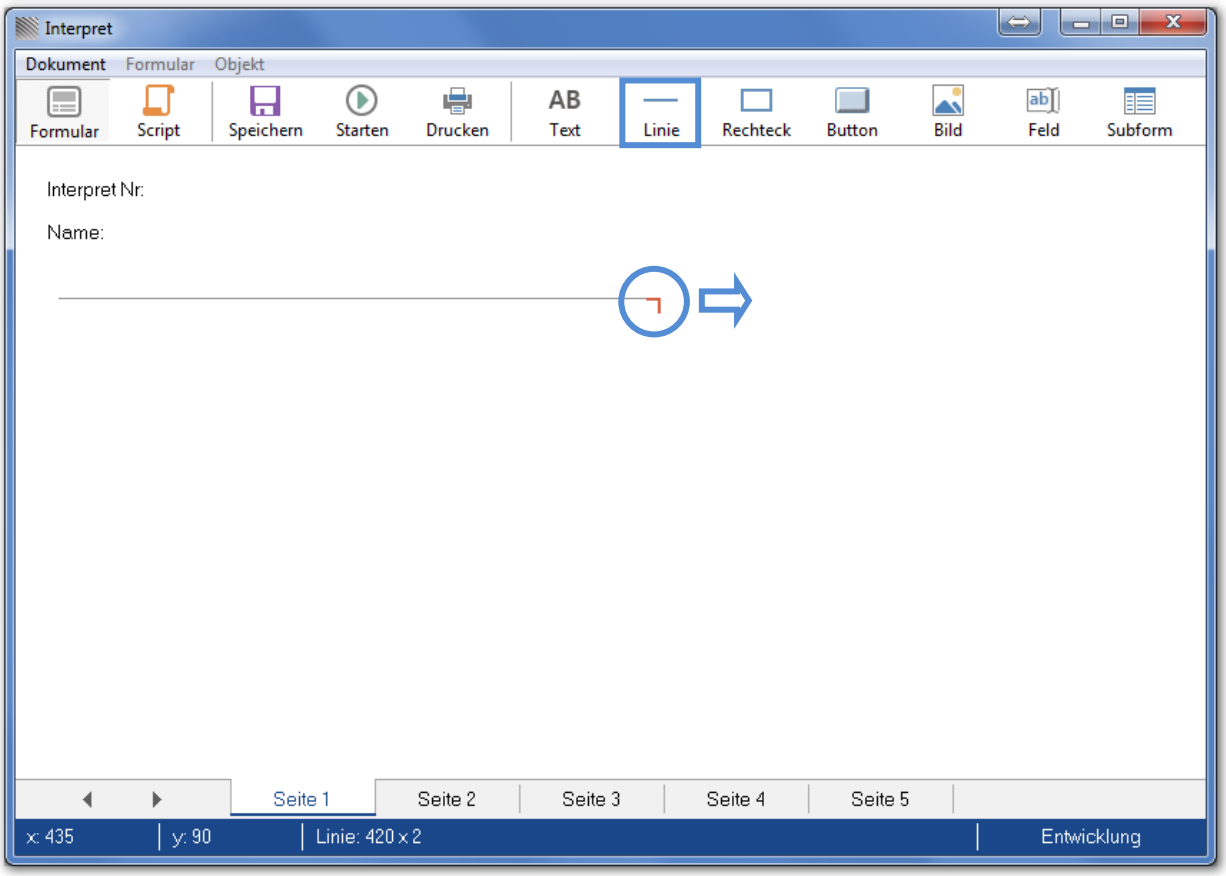
Klicken Sie mit der rechten Maustaste auf den Text und wählen Sie „Verschieben“ aus dem Kontextmenü. Sie können den Verschiebungsmodus auch mit der Tastenkombination Strg+M aktivieren wenn der Cursor links unter dem Text steht.

Text löschen

Zum Entfernen eines Textobjektes klicken Sie mit der rechten Maustaste auf das Objekt und wählen Sie die Option „Löschen“ aus dem Kontextmenü. Sie können Textobjekte auch mit der Taste Entf löschen, wenn der Cursor links unter dem Text steht.

4.2.1.2 Objekt hinzufügen: Linie

Mit DataCool können Sie horizontale oder vertikale Linien erstellen. Bewegen Sie zunächst den Cursor an den Anfangspunkt. Wählen Sie nun „Linie“ in der Toolbar oder drücken Sie Strg+L. Anschließend bewegen Sie den Cursor an den Endpunkt. Mit Return wird die Bearbeitung der Linie abgeschlossen.



Linie bearbeiten

Klicken Sie mit der rechten Maustaste auf die Linie und wählen Sie die Option „Bearbeiten“ aus dem Kontextmenü. Der Cursor nimmt die Farbe rot an. Bewegen Sie nun den Anfangs- bzw. Endpunkt auf die neue Position. Mit Return wird die Bearbeitung der Linie abgeschlossen. Mit der Taste ESC können Sie die Bearbeitung jederzeit abbrechen.

Linie verschieben

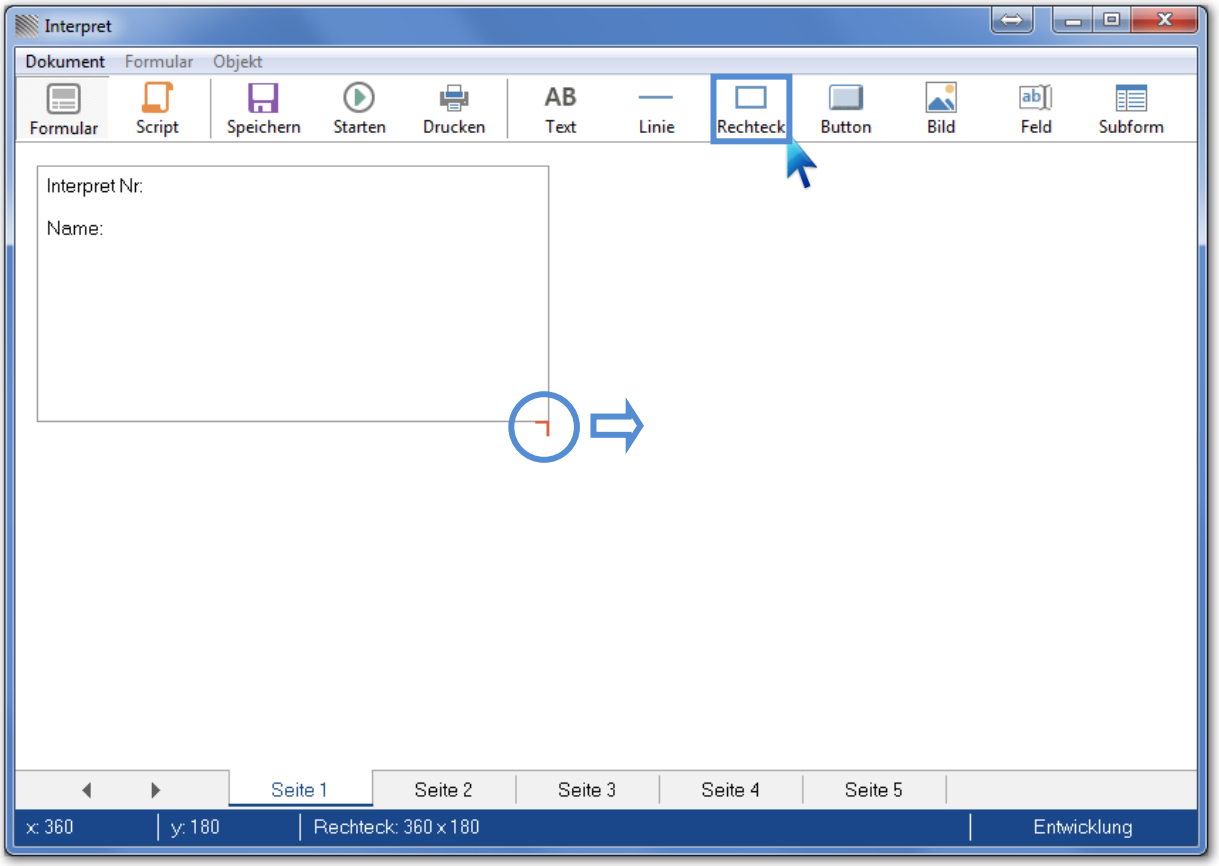
Klicken Sie mit der rechten Maustaste auf die Linie und wählen Sie die Option „Verschieben“ aus dem Kontextmenü. Der Cursor nimmt die Farbe gelb an. Bewegen Sie nun die Linie mit den Cursortasten an die gewünschte Position. Mit Return wird die Positionierung der Linie abgeschlossen. Mit ESC verbleibt die Linie an Ihrer ursprünglichen Position.

Linie löschen

Klicken Sie mit der rechten Maustaste auf die Linie und wählen Sie die Option „Löschen“ aus dem Kontextmenü. Sie können die Linie auch mit Entf löschen, wenn der Cusor auf dem Startpunkt der Linie steht.

4.2.1.3 Objekt hinzufügen: Rechteck

Rechtecke werden in der Regel dazu verwendet, bestimmte Eingabebereiche Ihrer Maske zu gruppieren. Bewegen Sie zunächst den Cursor an eine Ecke des gewünschten Rechtecks. Wählen Sie nun „Rechteck“ in der Toolbar oder drücken Sie Strg+R. Anschließend bewegen Sie den Cursor an die gegenüberliegende Ecke. Mit Return wird die Bearbeitung des Rechtecks abgeschlossen.



Rechteck bearbeiten

Klicken Sie mit der rechten Maustaste auf eine Linie des Rechtecks und wählen Sie die Option „Bearbeiten“ aus dem Kontextmenü. Der rote Cursor springt automatisch auf die nächstgelegene Ecke. Bewegen Sie nun den Eckpunkt mit den Cursortasten an die gewünschte Position. Mit Return wird die Bearbeitung des Rechtecks abgeschlossen. Mit der Taste ESC können Sie die Bearbeitung jederzeit abbrechen.

Rechteck verschieben

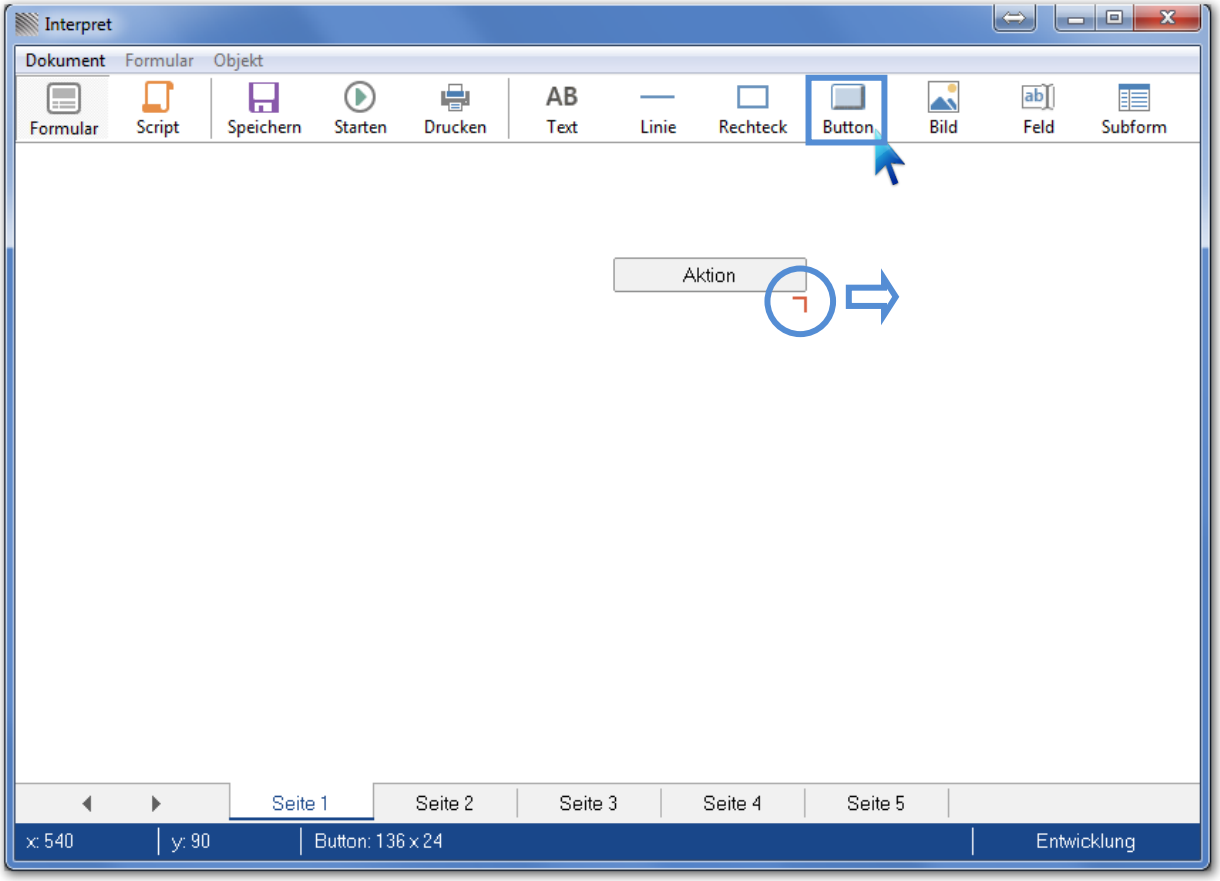
Klicken Sie mit der rechten Maustaste auf eine Linie des Rechtecks und wählen Sie die Option „Verschieben“ aus dem Kontextmenü. Der gelbe Cursor springt automatisch auf die linke untere Ecke des Rechtecks. Mit den Cursortasten können Sie das Rechteck an die gewünschte Position bewegen. Mit Return wird die Positionierung abgeschlossen. Mit ESC wird die Verschiebung rückgängig gemacht.

Rechteck löschen

Klicken Sie mit der rechten Maustaste auf eine Linie des Rechtecks und wählen Sie die Option „Löschen“ aus dem Kontextmenü. Sie können das Rechteck auch mit Entf löschen, wenn der Cusor an der linken unteren Ecke steht.

4.2.1.4 Objekt hinzufügen: Button

Ein Button ist eine rechteckige Schaltfläche, die beim Anklicken eine Aktion auslöst. Bewegen Sie den Cursor an die linke untere Ecke und wählen Sie die Option „Button“ aus der Toolbar. Anschließend bewegen Sie den Cursor nach rechts bis die gewünschte Breite des Buttons erreicht ist. Mit Return wird die Bearbeitung des Buttons abgeschlossen. Im Anschluss können Sie den Button beschriften und mit einer Aktion versehen.



NEU

Ab der Version 2010 können Sie auch die Höhe des Buttons einstellen. Auf diese Weise können Sie nun große Schaltflächen erstellen, die sich optimal auf einem Touchscreen bedienen lassen.

Button bearbeiten

Klicken Sie mit der rechten Maustaste auf den Button und wählen Sie die Option „Bearbeiten“ aus dem Kontextmenü. Der rote Cursor springt automatisch auf die rechte untere Ecke des Buttons. Durch Bewegung der Cursortasten können Sie die gewünschte Breite und Höhe einstellen. Mit Return wird die Bearbeitung des Buttons abgeschlossen. Mit der Taste ESC können Sie die Bearbeitung jederzeit abbrechen.

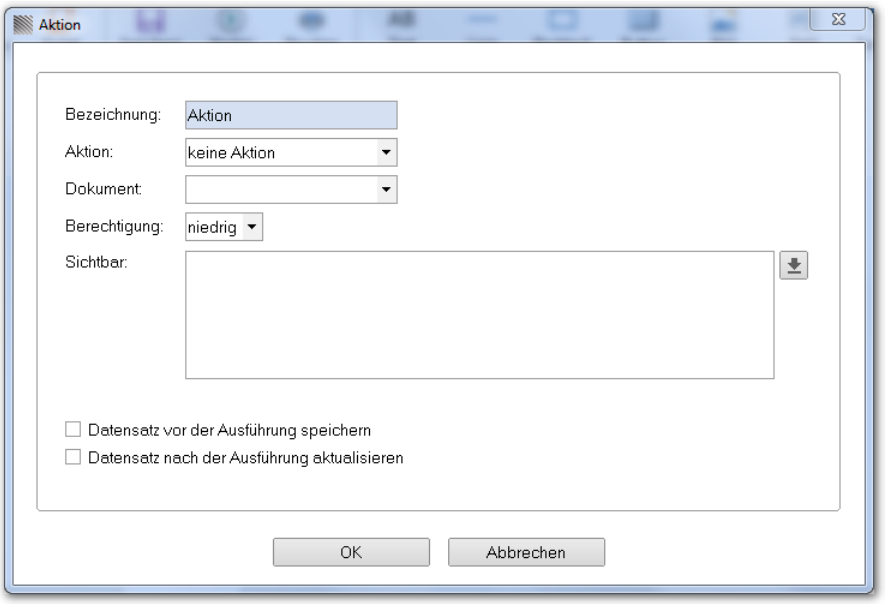
Button verschieben

Klicken Sie mit der rechten Maustaste auf den Button und wählen Sie die Option „Verschieben“ aus dem Kontextmenü. Der gelbe Cursor springt automatisch auf die linke untere Ecke. Mit den Cursortasten können Sie den Button an die gewünschte Position bewegen. Mit Return wird die Positionierung abgeschlossen. Mit ESC wird die Verschiebung rückgängig gemacht.

Button löschen

Klicken Sie mit der rechten Maustaste auf den Button und wählen Sie die Option „Löschen“ aus dem Kontextmenü. Sie können den Button auch mit Entf löschen, wenn der Cusor an der linken unteren Ecke steht.

Aktion definieren



Klicken Sie mit der rechten Maustaste auf den Button und wählen Sie die Option „Aktion“ aus dem Kontextmenü. Im nebenstehenden Dialog können Sie nun die Beschriftung und die auszuführende Aktion des Buttons festlegen. Außerdem legen Sie die Berechtigungsstufe fest, die ein Benutzer mindestens besitzen muss um die Aktion ausführen zu können. Folgende Berechtigungsstufen sind möglich: **niedrig**, **mittel**, **hoch** und **admin**.

Die Option „**Datensatz vor der Ausführung speichern**“ bewirkt, dass die angezeigten Daten abgespeichert werden, **bevor** die Aktion ausgeführt wird. Sollte das Speichern fehlschlagen (z.B. aufgrund von fehlenden Benutzereingaben), so wird die Aktion nicht ausgeführt.

Die Option „**Datensatz nach der Ausführung aktualisieren**“ bewirkt, dass der angezeigte Datensatz **nach** Ausführung der Aktion frisch aus der SQL-Tabelle geladen wird. Diese Funktion ist nützlich, wenn die Ausführung der Aktion eine Veränderung des Datensatzes bewirkt (z.B. durch einen Scriptlauf).

Sie haben die Wahl zwischen folgenden Aktionen:

keine Aktion	Wählen Sie diese Option, wenn Sie den Button erst zu einem späteren Zeitpunkt mit einer Aktion belegen wollen.
Menü öffnen	Öffnet das ausgewählte Menü.
Formular öffnen	Öffnet das ausgewählte Formular.
Script starten	Startet das ausgewählte Script.
Import starten	Startet einen Datenimport auf der Basis der angegebenen Importdefinition.

Export starten	Startet einen Datenexport auf der Basis der angegebenen Exportdefinition.
Code ausführen	Startet das im Dokument eingebettete Script. Als „Event ID“ können Sie einen freien Text festlegen, der innerhalb des Scriptes mit der Systemvariable **current.event** abgefragt werden kann. Auf diese Weise kann das Script auf das eingetretene Ereignis reagieren.
Auswahl definieren	Eingabefelder vom Typ „Auswahl“ stellen dem Anwender eine Reihe vordefinierter Optionen zur Verfügung. Mit der Aktion „Auswahl definieren“ geben Sie dem Benutzer die Möglichkeit zur Veränderung einer Auswahlliste.
Verknüpfung öffnen	Eine Verknüpfung ist eine Beziehung (Relation) zwischen zwei Formularen. Beim Ausführen dieser Aktion wird das angegebene Formular mit dem verknüpften Datensatz geöffnet. Die Funktionsweise von Verknüpfungen wird im Kapitel 4.4 näher erläutert.
Arbeitsplatzkonfiguration	Beim Ausführen dieser Aktion wird der Dialog zur Konfiguration des Arbeitsplatzes aufgerufen. Auf diese Weise kann der Anwender die Druckerkonfiguration und die TAPI-Einstellungen für seinen Arbeitsplatz selbst anpassen.

Sichtbar

Ab der Version 2010 können Sie Buttons definieren, die nicht immer sichtbar sind. Sie können Sie Sichtbarkeit des Buttons mit Hilfe einer Ableitungsformel steuern. Der Button ist **sichtbar** wenn das Ergebnis der Ableitungsformel den Wert **true** annimmt. Andernfalls ist der Button unsichtbar.

4.2.1.5 Objekt hinzufügen: Bild

Bilder dienen der Verschönerung von Formularen und Menüs. Sie können aber auch mit Aktionen versehen werden. Auf diese Weise können Sie beispielsweise selbstgestaltete grafische Buttons realisieren. Bewegen Sie den Cursor an die linke untere Bildposition und wählen Sie die Option „Bild“ aus der Toolbar. Anschließend öffnet sich ein Dialog zur Auswahl der Grafikdatei. Folgende Grafikformate werden unterstützt: **gif**, **jpg**, **bmp**, **ico**, **tif**, **png**.

Bilder werden von DataCool in der Datenbank als Binärdaten gespeichert. Sie können die Bilddatei also bei Bedarf löschen - trotzdem bleibt das Bild als Teil Ihrer Eingabemaske erhalten. Die Abmessungen des Bildes können mit DataCool nicht verändert werden. Sie müssen das Bild ggf. mit einem Bildbearbeitungsprogramm (z.B. IrfanView) entsprechend vorbereiten. Bitte beachten Sie auch, dass die Dateigröße nicht mehr als 50-100 KByte betragen sollte, damit es zu keinen Wartezeiten beim Laden aus der Datenbank kommt.

Im Programmverzeichnis finden Sie die beiden ZIP-Dateien AppIcons16x16.zip und AppIcons24x24.zip. Diese Dateien enthalten eine Reihe von Standard-Icons im Look von DataCool 2020, die Sie bei Bedarf in Ihren Anwendungen verwenden können.

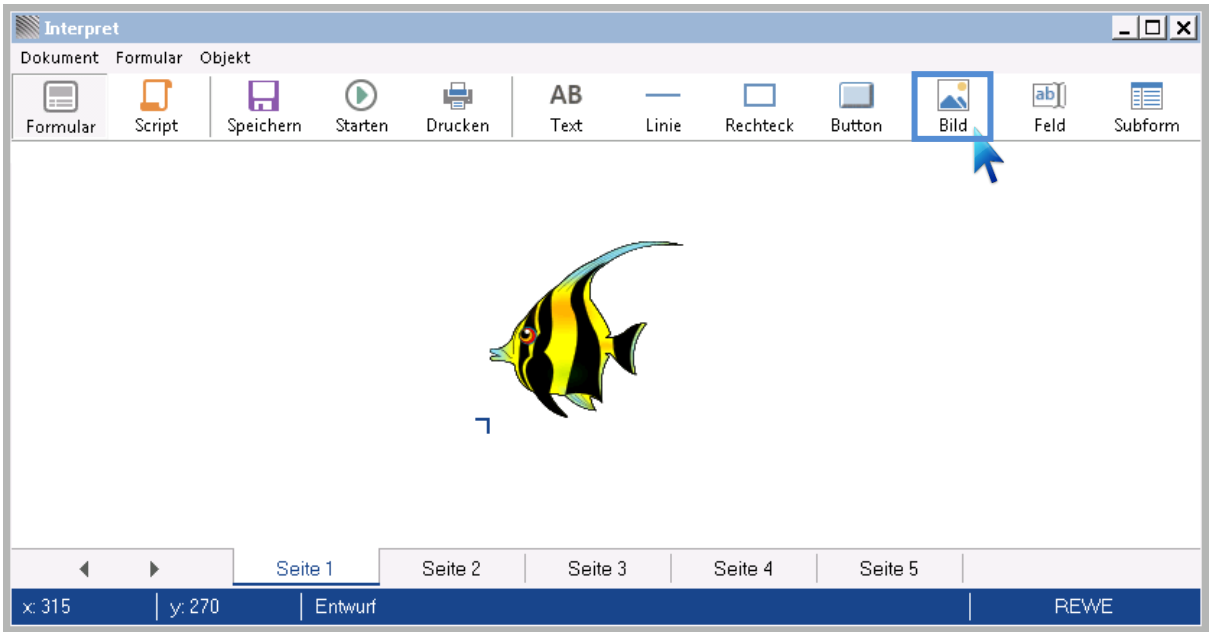


Bild bearbeiten

Klicken Sie mit der rechten Maustaste auf das Bild und wählen Sie die Option „Bearbeiten“ aus dem Kontextmenü. Sie können nun mit dem Dateidialog eine neue Grafikdatei auswählen.

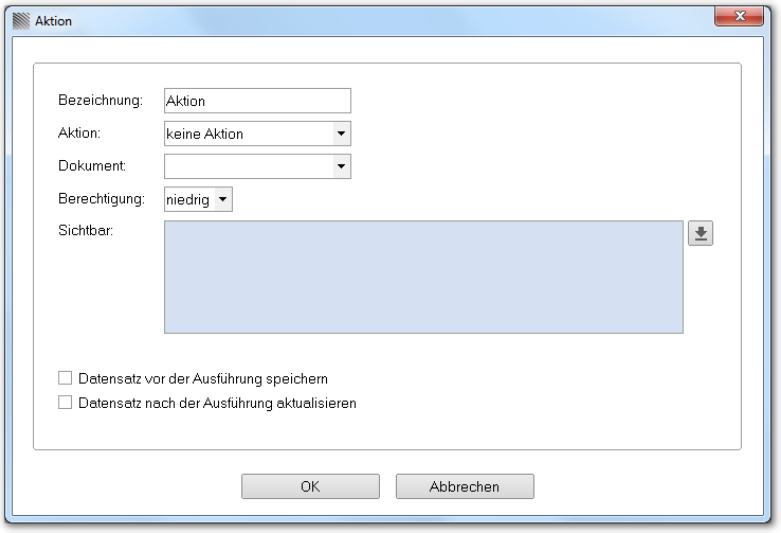
Bild verschieben

Klicken Sie mit der rechten Maustaste auf das Bild und wählen Sie die Option „Verschieben“ aus dem Kontextmenü. Der gelbe Cursor springt automatisch auf die linke untere Ecke. Mit den Cursortasten können Sie das Bild an die gewünschte Position bewegen. Mit Return wird die Positionierung abgeschlossen. Mit ESC wird die Verschiebung rückgängig gemacht.

Bild löschen

Klicken Sie mit der rechten Maustaste auf das Bild und wählen Sie die Option „Löschen“ aus dem Kontextmenü. Sie können das Bild auch mit Entf löschen, wenn der Cusor an der linken unteren Ecke steht.

Aktion definieren



Klicken Sie mit der rechten Maustaste auf das Bild und wählen Sie die Option „Aktion“ aus dem Kontext-menü. Im nebenstehenden Dialog können Sie nun den Tooltip-Text und die auszuführende Aktion defi-nieren. Außerdem legen Sie die Bere-chtigungsstufe fest, die ein Benutzer mindestens besitzen muss um die Aktion ausführen zu können. Fol-gende Berechtigungsstufen sind möglich: **niedrig**, **mittel**, **hoch** und **admin**.

Die Option „**Datensatz vor der Ausführung speichern**“ bewirkt, dass die angezeigten Daten abgespeichert werden, **bevor** die Aktion ausgeführt wird. Sollte das Speichern fehlschlagen (z.B. aufgrund von fehlenden Benutzereingaben), so wird die Aktion nicht ausgeführt.

Die Option „**Datensatz nach der Ausführung aktualisieren**“ bewirkt, dass der angezeigte Datensatz **nach** Ausführung der Aktion frisch aus der SQL-Tabelle geladen wird. Diese Funktion ist nützlich, wenn die Ausführung der Aktion eine Veränderung des Datensatzes bewirkt (z.B. durch einen Scriptlauf).

Sie haben die Wahl zwischen folgenden Aktionen:

keine Aktion	Wählen Sie diese Option, wenn Sie erst zu einem späteren Zeitpunkt eine Aktion festlegen wollen.
Menü öffnen	Öffnet das ausgewählte Menü.
Formular öffnen	Öffnet das ausgewählte Formular.
Script starten	Startet das ausgewählte Script.
Import starten	Startet einen Datenimport auf der Basis der angegebenen Importdefinition.
Export starten	Startet einen Datenexport auf der Basis der angegebenen Exportdefinition.
Code ausführen	Startet das im Dokument eingebettete Script. Als „Event ID“ können Sie einen freien Text festlegen, der innerhalb des Scriptes mit der Systemvariable **current.event** abgefragt werden kann. Auf diese Weise kann das Script auf das eingetretene Ereignis reagieren.
Auswahl definieren	Eingabefelder vom Typ „Auswahl“ stellen dem Anwender eine Reihe vordefinierter Optionen zur Verfügung. Mit der Aktion „Auswahl definieren“ geben Sie dem Benutzer die Möglichkeit zur Veränderung einer Auswahlliste.
Verknüpfung öffnen	Eine Verknüpfung ist eine Beziehung (Relation) zwischen zwei Formularen. Beim Ausführen dieser Aktion wird das angegebene Formular mit dem verknüpften Datensatz geöffnet. Die Funktionsweise von Verknüpfungen wird im Kapitel 4.4 näher erläutert.
Arbeitsplatzkonfiguration	Beim Ausführen dieser Aktion wird der Dialog zur Konfiguration des Arbeitsplatzes aufgerufen. Auf diese Weise kann der Anwender die Druckerkonfiguration und die TAPI-Einstellungen für seinen Arbeitsplatz selbst anpassen.

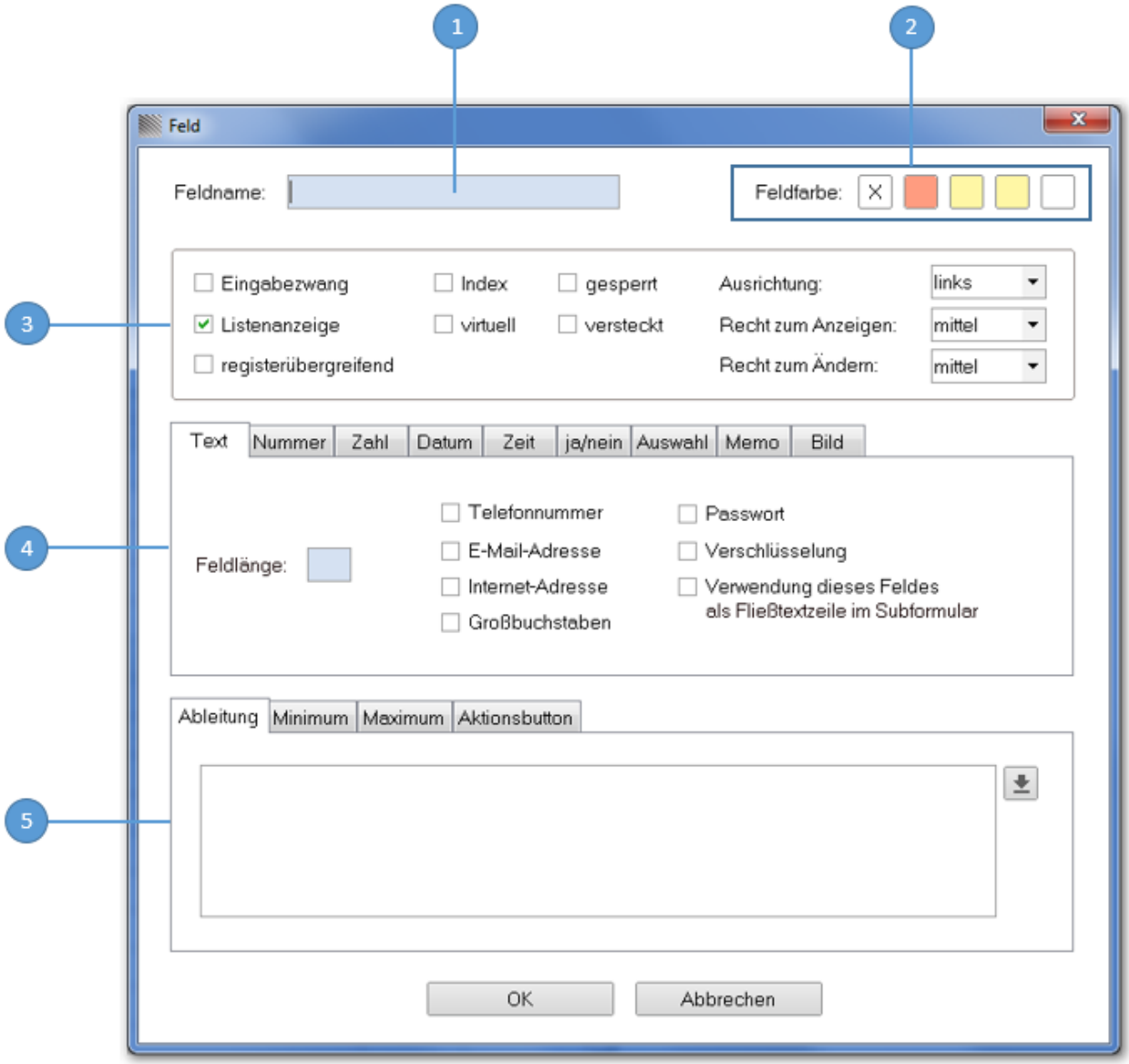
Sichtbar

Ab der Version 2010 können Sie Bildschaltflächen definieren, die nicht immer sichtbar sind. Sie können Sie Sichtbarkeit des Bildes mit Hilfe einer Ableitungsformel steuern. Das Bild ist **sichtbar** wenn das Ergebnis der Ableitungsformel den Wert **true** annimmt. Andernfalls ist das Bild unsichtbar.

4.2.1.6 Objekt hinzufügen: Feld

Felder sind die wichtigsten Elemente eines Formulars. Sie ermöglichen die Eingabe von Daten, die vom System in einer SQL-Tabelle abgespeichert werden. DataCool erstellt für jedes Formular automatisch eine passende SQL-Tabelle. Die Spalten der SQL-Tabelle entsprechen dabei den Feldern der Eingabemaske. Jeder Datensatz ist eine Zeile in der SQL-Tabelle.

Zur Anlage eines neuen Feldes bewegen Sie den Cursor an die gewünschte Stelle und wählen Sie die Option „Feld“ aus der Toolbar. Sie können auch die Taste F10 benutzen um ein Feld zu erstellen oder zu ändern.



Der Dialog zur Felddefinition ist in verschiedene Bereiche unterteilt:

1: Feldname
2. Feldfarbe
3. Feldeinstellungen
4. Feldtyp
5. Berechnungen und Prüfungen

Feldname

Der Feldname dient als Bezeichner für die Spalte in der SQL-Tabelle. Jeder Feldname darf innerhalb eines Formulars nur einmal verwendet werden, da der Feldname **eindeutig** sein muss. Feldbezeichner dürfen nur die Buchstaben A-Z, a-z sowie die Ziffern 0-9 enthalten. Sonderzeichen, Umlaute oder Leerzeichen sind nicht zulässig. DataCool wandelt die Umlaute ä, ö, ü automatisch in ae, oe und ue.

Tipp: Normalerweise steht links neben jedem Feld ein Text als Beschriftung. Wenn Sie zuerst die Beschriftung erstellen und danach rechts davon das Feld anlegen, so wird der Feldname automatisch aus dem nebenstehenden Text abgeleitet.

Feldfarbe

Die Standard-Feldfarbe ist **weiß für Eingabefelder** und **grau für gesperrte Felder**. Diese Farbe erhalten Sie durch Anklicken des ersten Buttons links. Zusätzlich stehen Ihnen **drei Sonderfarben** zur Verfügung. Diese Sonderfarben sind in allen Formularen gleich und werden zentral in der Systemkonfiguration eingestellt. Der letzte Button auf der rechten Seite bewirkt eine Darstellung des Feldes in Textfarbe. Zur Laufzeit wird zusätzlich der Rahmen des Feldes ausgeblendet, so dass das Feld wie ein Teil der Formularbeschriftung wirkt.

Eingabezwang

Wenn Sie diese Option ankreuzen, dann wird der Benutzer zur Eingabe eines Wertes in diesem Feld gezwungen. Sie können einen Datensatz nur dann abspeichern, wenn alle Felder mit Eingabezwang ausgefüllt sind.

Listenanzeige

Zur Laufzeit hat der Anwender die Möglichkeit zwischen Formular- und Tabellenansicht umzuschalten. Mit der Option „Listenanzeige“ können Sie festlegen, ob dieses Feld in der Tabellensicht gezeigt wird.

registerübergreifend

Seit der Version 2014 werden die einzelnen Bildschirmseiten eines Formulares als Registertabs dargestellt. Die Option „registerübergreifend“ bewirkt, dass das Feld auf jeder Registerseite sichtbar ist. Wenn Sie beispielsweise ein Kundenformular erstellen, dann empfiehlt es sich den Kundennamen auf jeder Registerseite darzustellen. Dadurch weiß der Anwender stets welche Kundendaten gerade auf dem Bildschirm zu sehen sind.

Index

Ein Index ist eine sortierte Liste, die von der Datenbank im Hintergrund gepflegt wird, damit Sie in diesem Feld schnell suchen oder sortieren können. Verknüpfungsfelder oder Felder in denen Sie häufig suchen sollten indiziert werden.

Bitte gehen Sie bei der Vergabe der Index-Eigenschaft mit Bedacht um, da zu viele Indizes den Speichervorgang verlangsamen können. Außerdem sollten Sie Felder nur dann indizieren wenn die Tabelle mehr als 1.000 Datensätze enthalten wird. Andernfalls ist das System auch ohne Index schnell genug.

virtuell

Virtuelle Felder werden nicht als Teil der SQL-Tabelle gespeichert. Stattdessen enthalten Sie eine Ableitungsformel, die beim Aufruf des Datensatzes jeweils neu berechnet wird. Bitte beachten Sie, dass virtuelle Felder bei Scriptläufen nicht zur Verfügung stehen. Sie müssen die angezeigten Werte dann ggf. im Script neu berechnen. Virtuelle Felder werden oft zur Anzeige von Werten aus verknüpften Formularen verwendet. Änderungen in der verknüpften Tabelle werden dann beim Aufruf eines Datensatzes automatisch reflektiert. Dies entspricht dem Konzept des relationalen Datenbankmodells, bei dem die doppelte Speicherung von Informationen vermieden wird.

gesperrt

Gesperrte Eingabefelder können vom Anwender nicht verändert werden. In der Regel werden diese Felder mittels einer Ableitungsformel oder durch das Formularscript ausgefüllt.

versteckt

Versteckte Felder sind zur Laufzeit für den Anwender nicht sichtbar. Versteckte Felder erhalten automatisch das Attribut „gesperrt“.

Ausrichtung

DataCool wählt für jeden Feldtyp automatisch die optimale Ausrichtung. In Ausnahmefällen kann es sinnvoll sein von der normalen Feldausrichtung abzuweichen. Wählen Sie einen der Werte **links**, **mittig** oder **rechts**.

Recht zum Anzeigen

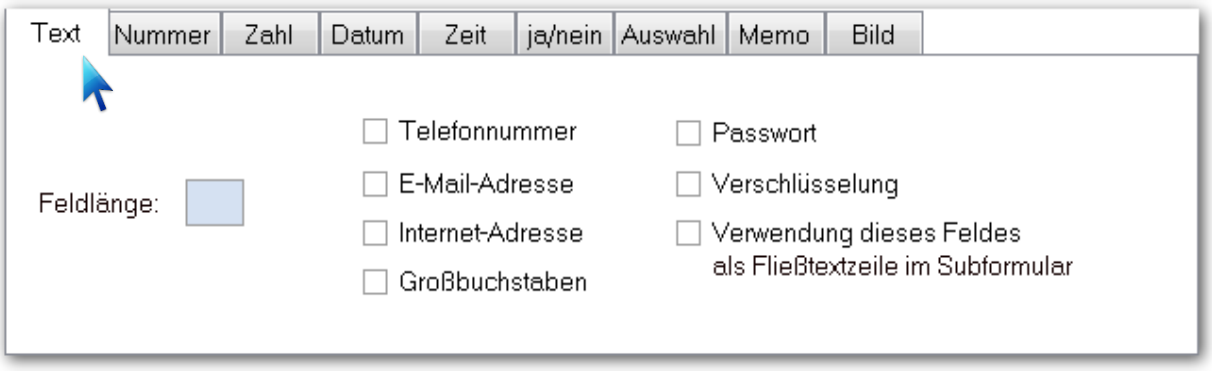
Wählen Sie die minimale Berechtigungsstufe, die ein Benutzer für die **Anzeige** des Feldinhaltes benötigt. Folgende Berechtigungsstufen sind verfügbar: **niedrig**, **mittel**, **hoch** und **admin**. Die Zuweisung von Rechten erfolgt in der Benutzerverwaltung.

Recht zum Ändern

Wählen Sie die minimale Berechtigungsstufe, die ein Benutzer zur **Änderung** von Feldinhalten benötigt. Folgende Berechtigungsstufen sind verfügbar: **niedrig**, **mittel**, **hoch** und **admin**. Die Zuweisung von Rechten erfolgt in der Benutzerverwaltung.

Feldtyp „Text“

Der Feldtyp „Text“ wird zur Speicherung von alphanumerischen Daten verwendet. Sie müssen eine maximale Feldlänge zwischen einem und 200 Zeichen festlegen.



Darüberhinaus stehen weitere Feldeigenschaften für diesen Typ zur Verfügung:

Telefonnummer	Die Eingabe einer Telefonnummer erfolgt im Format 0841/37071-0. Beim Verlassen des Feldes wird die Rufnummer automatisch umformatiert um eine optimale Lesbarkeit zu erzielen: (08 41) 3 70 71-0. Hierbei werden die Ziffern von rechts nach links in Zweierblöcken aufgeteilt und durch Leerzeichen getrennt. Bei internationalen Vorwahlen sollten Sie das Zeichen „+“ voranstellen. Somit wird aus +49841/37071-0 die Zeichenfolge +49 (8 41) 3 70 71-0. DataCool erkennt alle internationalen Vorwahlen und zieht diese vor die Klammer mit der Ortsvorwahl.
E-Mail-Adresse	Eingabe einer gültigen E-Mail-Adresse, z.B. info@datacool.net
Internet-Adresse	Eingabe einer URL, z.B. www.datacool.net.
Großbuchstaben	Eingaben werden automatisch in Großbuchstaben gewandelt.
Passwort	Während der Eingabe erscheinen Sternchen anstelle von Buchstaben, damit niemand das Passwort am Bildschirm mitlesen kann.
Verschlüsselung	DataCool verschlüsselt die Eingaben vor der Speicherung in der SQL-Tabelle. Die Informationen können danach nur noch innerhalb von DataCool gelesen werden. Auf diese Weise wird beispielsweise ein verschlüsseltes Passwortfeld vor einem Direktzugriff auf die SQL-Tabelle geschützt.
NEU Fließtextzeile	Bei Aktivierung dieser Option wird das Textfeld in einem Subformular so behandelt, als wäre es ein großes Memofeld.

Feldtyp „Nummer“

Ein Nummernfeld ist ein Zahlenfeld mit führenden Nullen. Bei Bedarf kann die Nummer auch mit Hilfe von Trennzeichen formatiert werden. Folgende Trenner sind zulässig: Doppelpunkt, Schrägstrich, Bindestrich, Punkt.

Der Formatstring dient zur Festlegung der Stellenzahl und der Position von Formatierungszeichen. Die Ziffer „0“ im Formatstring symbolisiert eine Zahl an der jeweiligen Stelle.

Beispiele für gültige Formatstrings: 00000, 000/00000, 000:000, 00-0000, 00.0000.

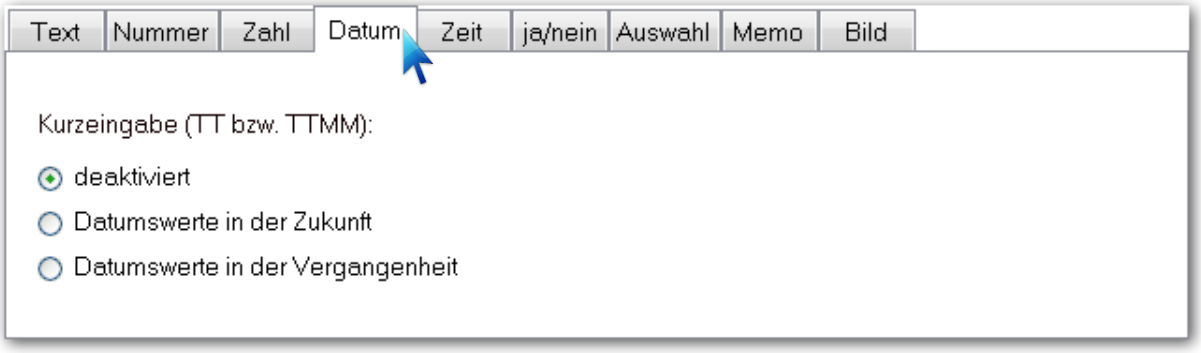
Feldtyp „Zahl“

Der Feldtyp „Zahl“ ist für numerische Daten vorgesehen. Hierbei wird zwischen **Ganzzahlen**, **Festkommazahlen** und **Gleitkommazahlen** unterschieden.

Bei **Ganzzahlen** legen Sie die gewünschte Stellenzahl fest und geben 0 im Feld „Nachkomma“ an. Bei **Festkommawerten** geben Sie im Feld „Stellen“ die Anzahl der Vorkommastellen an und im Feld „Nachkomma“ die Anzahl der Dezimalstellen. Bei **Gleitkommazahlen** füllen Sie nur das Feld „Stellen“ aus und lassen die Angabe für die Nachkommastellen frei.

Feldtyp „Datum“

Dieser Feldtyp ist für Datumswerte vorgesehen.



Zur Laufzeit kann die Eingabe eines Datums auf verschiedene Weisen erfolgen:

TT.MM.JJJJ	Angabe mit Punkten, vierstellige Jahreszahl
TTMMJJJJ	Angabe ohne Punkte, vierstellige Jahres zahl
TT.MM.JJ	Angabe mit Punkten, zweistellige Jahreszahl
TTMMJJ	Angabe ohne Punkte, zweistellige Jahreszahl
H	Heute: bei Eingabe von „H“ wird das Systemdatum übernommen

Fehlende Punkte werden von DataCool automatisch ergänzt. Zweistellige Jahreszahlen werden von DataCool in vierstellige Jahreszahlen gewandelt. Hierbei wird das **Startjahr** aus der Systemkonfiguration zugrundegelegt. Die Einstellung „70“ bedeutet, dass zweistellige Jahreszahlen in den Zeitraum von 1970 bis 2069 fallen.

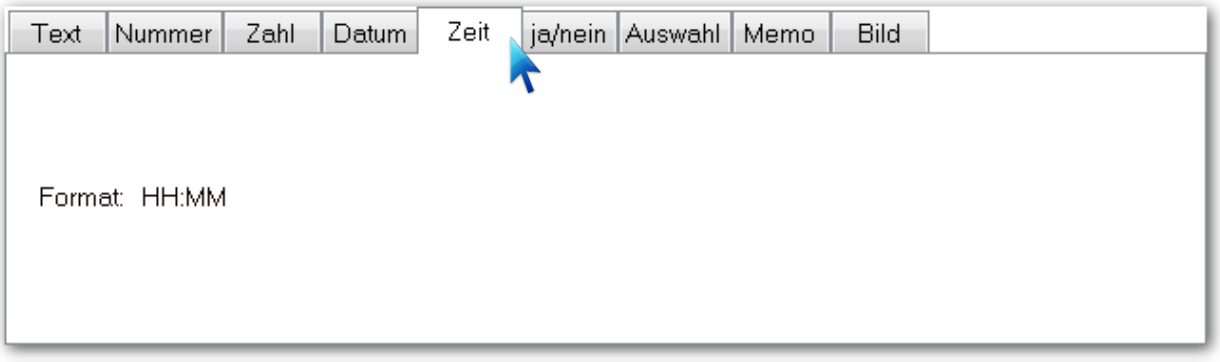
Die Option der **Kurzeingabe** ermöglicht auch die Eingabe von Datumsfragmenten im Format TT oder TTMM. Hierbei müssen Sie festlegen, ob die Kurzeingabe als Datumswert in der Zukunft oder in der Vergangenheit interpretiert werden soll. DataCool ergänzt dann Ihre Eingabe, so dass der nächstgelegene Datumswert in der Zukunft oder Vergangenheit erreicht wird.

Sofern die Kurzeingabe aktiviert ist stehen Ihnen weitere Hotkeys zur Veränderung des Datums zur Verfügung:

+	Weiterschaltung um einen Tag in die Zukunft
-	Weiterschaltung um einen Tag in die Vergangenheit
E	Erster des Monats bzw. Weiterschaltung um einen Monat
L	Letzter des Monats bzw. Weiterschaltung um einen Monat

Feldtyp „Zeit“

Der Feldtyp „Zeit“ ermöglicht die Eingabe einer Uhrzeit im Format **Stunden:Minuten**. Zeitangaben werden von DataCool als Textwerte in der SQL-Datenbank gespeichert.



Feldtyp „ja/nein“

Ja/Nein-Felder können wahlweise als **Auswahlfeld** oder als **Markierungsfeld** dargestellt werden. Ein Auswahlfeld kann drei Werte annehmen: leer, nein oder ja. In der SQL-Datenbank entspricht dies den Werten null, 1 oder 2. Ein Markierungsfeld ist entweder angekreuzt (SQL-Wert: 2) oder nicht (SQL-Wert: 1).

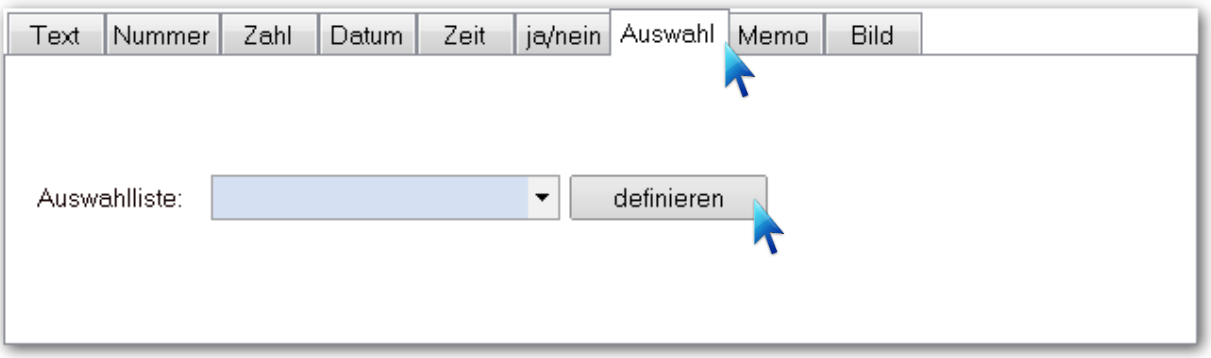
Bitte beachten Sie die Besonderheit bei Ja/Nein-Feldern: Die Abfrage der **Eingabemaske** gibt den **Textwert** „nein“ oder „ja“ zurück während die Abfrage der **SQL-Tabelle** den **Zahlenwert** 1 oder 2 liefert.

Zusammenfassung: „ja“ = SQL-Wert 2, „nein“ = SQL-Wert 1

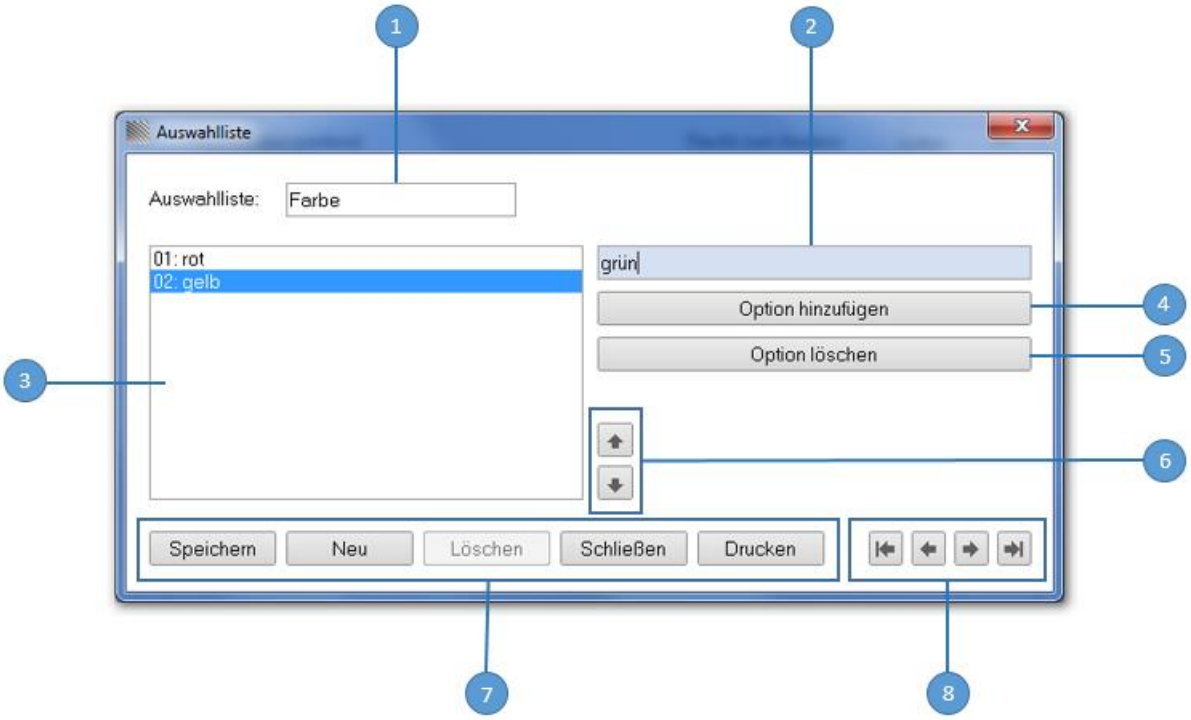
Sie können jederzeit zwischen den beiden Darstellungsformen umschalten, ohne dass hierbei Daten verloren gehen.

Feldtyp „Auswahl“

Der Feldtyp „Auswahl“ bietet dem Anwender eine vordefinierte Anzahl von Optionen. Auswahllisten werden von DataCool an zentraler Stelle gespeichert und stehen deshalb **formularübergreifend** zur Verfügung. Jede Auswahlliste besitzt einen eindeutigen Namen. Änderungen an der Liste wirken sich unmittelbar auf alle betroffenen Formulare aus.



So definieren Sie eine neue Auswahlliste:



Vergeben Sie im Feld (1) einen eindeutigen Namen für die neue Auswahlliste. Geben Sie die erste Option im Feld (2) an und klicken Sie anschließend auf den Button (4). Die Option wird dadurch am Ende der Liste (3) hinzugefügt. Wiederholen Sie diesen Vorgang für jede benötigte Auswahloption.

Zum Löschen einer Option wählen Sie diese in der Liste (3) aus und klicken Sie anschließend auf den Button (5).

Sie können auch die Reihenfolge verändern. Markieren Sie hierzu die gewünschte Zeile in der Liste und verschieben Sie diese mit Hilfe der Buttongruppe (6).

In der Buttongruppe (7) finden Sie diverse Bearbeitungsfunktionen für die Liste: Speichern, Neu, Löschen, Schließen und Drucken. Mit Hilfe der Navigationsleiste (8) können Sie zwischen den Auswahllisten navigieren.

Eine Auswahlliste lässt sich nur löschen, wenn diese an kein Feld mehr gebunden ist. Sie können die Liste also nicht löschen, solange sie von einem anderen Formular verwendet wird.

Auswahlfelder werden in der SQL-Tabelle als Zahl (1-99) gespeichert. Sie können die Auswahloptionen deshalb nicht gefahrlos ändern, wenn die Tabelle bereits Datensätze enthält. In diesem Fall müssen Sie ggf. mit Hilfe von SQL-Befehlen die Datensätze in geeigneter Weise angleichen. Neue Optionen am Ende einer Auswahlliste können jederzeit gefahrlos hinzugefügt werden.

Feldtyp „Memo“

Memofelder sind mehrzeilige Textfelder mit automatischem Zeilenumbruch. Bei der Definition eines Memofeldes können Sie die **Breite** in Zeichen und die **Höhe** in Zeilen angeben. Die tatsächlich angezeigte Zeilenanzahl kann in der Praxis höher sein als der angegebene Wert, da DataCool die Höhe des Memofeldes dem Raster anpasst.

Für Memofelder werden grundsätzlich **4096 Zeichen** in der SQL-Tabelle reserviert. Dies gilt unabhängig von der angezeigten Größe des Memofeldes am Bildschirm. Sie können die Breite und Höhe des Memofeldes also jederzeit gefahrlos ändern.

Memofelder führen einen automatischen Zeilenumbruch aus. Sie können jedoch auch manuelle Zeilenumbrüche mit der Taste Return einfügen. Manuelle und automatische Umbrüche werden mit dem Datensatz abgespeichert. Beim Ausdruck von Memofeldern werden diese stets so umgebrochen, wie sie zum Zeitpunkt des Speicherns am Bildschirm zu sehen waren.

Feldtyp „Bild“

Mit Hilfe von Bildfeldern können Sie Abbildungen und PDF-Dokumente als Teil des Datensatzes speichern. Bildfelder werden als Binärdaten direkt in der Datenbank abgelegt. Sie können die Originaldateien also bedenkenlos löschen - die Abbildung bleibt trotzdem als Teil des Datensatzes erhalten.

Bildformat	Wählen Sie eines der folgenden Formate: gif, jpg, bmp, ico, tif, png, pdf.
Breite x Höhe	Geben Sie Breite und Höhe des Bildfeldes auf dem Bildschirm an. Wenn Sie die Option „Bild in Originalgröße speichern“ aktivieren, dann werden die Bilder auf dem Bildschirm verkleinert dargestellt aber in der Datenbank in voller Größe gespeichert. In der Regel sollten Sie diese Option jedoch nicht aktivieren, da die Speicherung von Bildern in Originalgröße sehr viel Platz in der Datenbank beansprucht und das Laden der Datensätze verlangsamt.

Wenn Sie das Bild nicht in Originalgröße speichern, dann öffnet sich beim Laden eines Bildes der **neue Bildeditor** von DataCool 2020. Sie können hierbei Bilder aus der Zwischenablage einfügen oder beliebige Bildformate laden. Der Editor ermöglicht Ihnen danach die Auswahl eines Bildausschnittes. Der gewählte Bildausschnitt wird von DataCool automatisch auf die vordefinierte Bildgröße skaliert und verkleinert in der Datenbank gespeichert.

Wenn der Auswahlrahmen die Farbe rot hat, dann stimmt das Seitenverhältnis nicht mit dem angegebenen Seitenverhältnis im Formular überein. DataCool führt trotzdem eine verzerrungsfreie Skalierung durch und füllt die kürzere Bildseite mit einem weißen Rand auf.

Wenn der Auswahlrahmen die Farbe grün hat, dann ist das Seitenverhältnis des gewählten Ausschnittes optimal.

Bild-Verzeichnis

Wählen Sie ein Standard-Verzeichnis für die Auswahl von Bildern. Der Benutzer kann zur Laufzeit auch auf andere Verzeichnisse zugreifen.

Pickup-Verzeichnis

Bilder, die vom Anwender aus dem Pickup-Verzeichnis übernommen werden, werden vom System automatisch gelöscht. Diese Einstellung ist sinnvoll wenn Sie die Bilder ausschließlich in der Datenbank speichern wollen. Sollte nur eine Datei im Pickup-Verzeichnis stehen, so wird diese ohne Auswahldialog direkt übernommen und gelöscht.

Berechnungen und Prüfungen

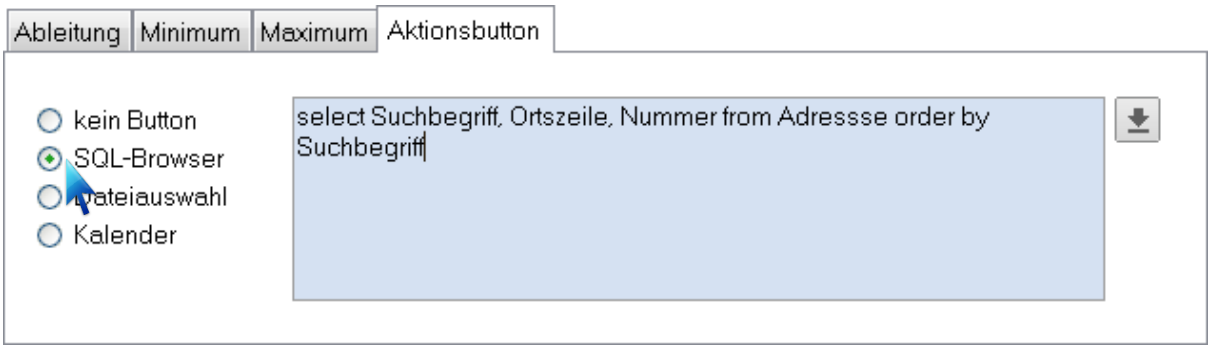
Im unteren Bereich der Felddefinition finden Sie vier Register zur Festlegung weiterer Eigenschaften:

Ableitung	Eine Ableitung veranlasst DataCool, den Wert eines Feldes automatisch zu bestimmen. Die Berechnung des Feldwertes erfolgt mit Hilfe einer **Ableitungsformel**. Eine Formel kann aus Konstanten, Feldnamen, Systemvariablen, Funktionen oder Operatoren bestehen. Die Feldnamen können sich auf Felder des gleichen Formulars oder auf Felder aus einem verknüpften Formular beziehen. Beispiele für Ableitungsformeln: “EUR“ current.date Nettobetrag*1.19 Kunde.Postleitzahl Name & “ “ & Vorname if(Datum=current.date, “Heute“, null) Eine vollständige Auflistung der Funktionen und Operatoren finden Sie im Referenzteil.
Minimum	Geben Sie bei Bedarf einen **unteren Grenzwert** an. Sie können einen festen Wert oder eine Formel eingeben.
Maximum	Geben Sie bei Bedarf einen **oberen Grenzwert** an. Sie können einen festen Wert oder eine Formel eingeben.
Aktionsbutton	Mit Hilfe eines Aktionsbuttons können Sie die Funktionalität des Feldes erweitern. Es stehen folgende Möglichkeiten zur Wahl: SQL-Browser, Dateiauswahl, Kalender, Telefonwahl, E-Mail, Internet.

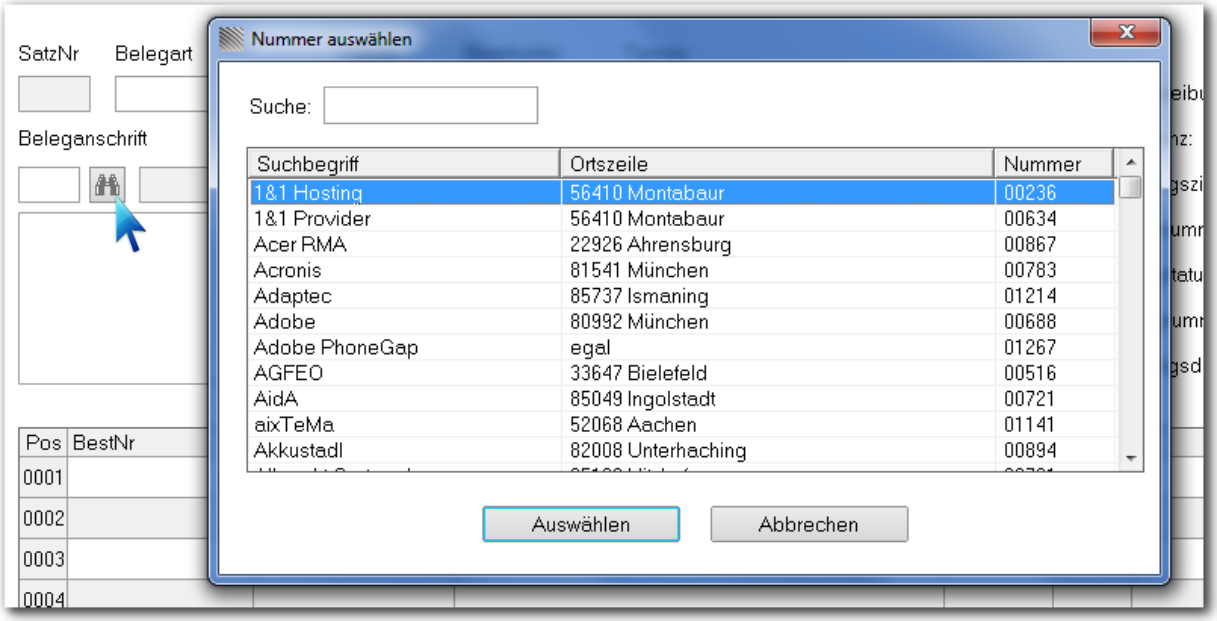
SQL-Browser

Ein **SQL-Browser** gibt dem Anwender die Möglichkeit Daten aus einer fremden Tabelle zu selektieren. Hierzu wird in der Felddefinition ein SQL-Befehl hinterlegt. Dieser Befehl bestimmt die Felder, die in der Auswahltabelle angezeigt werden. Das **zuletzt genannte Feld** ist der **Rückgabewert**.

Im nachfolgenden Beispiel werden die Felder Suchbegriff, Ortszeile und Nummer aus einer Adresstabelle gelistet. Die Datensätze werden nach dem Suchbegriff sortiert. Als Auswahlergebnis wird das Feld „Nummer“ zurückgegeben.



Das Ergebnis sieht zur Laufzeit so aus:

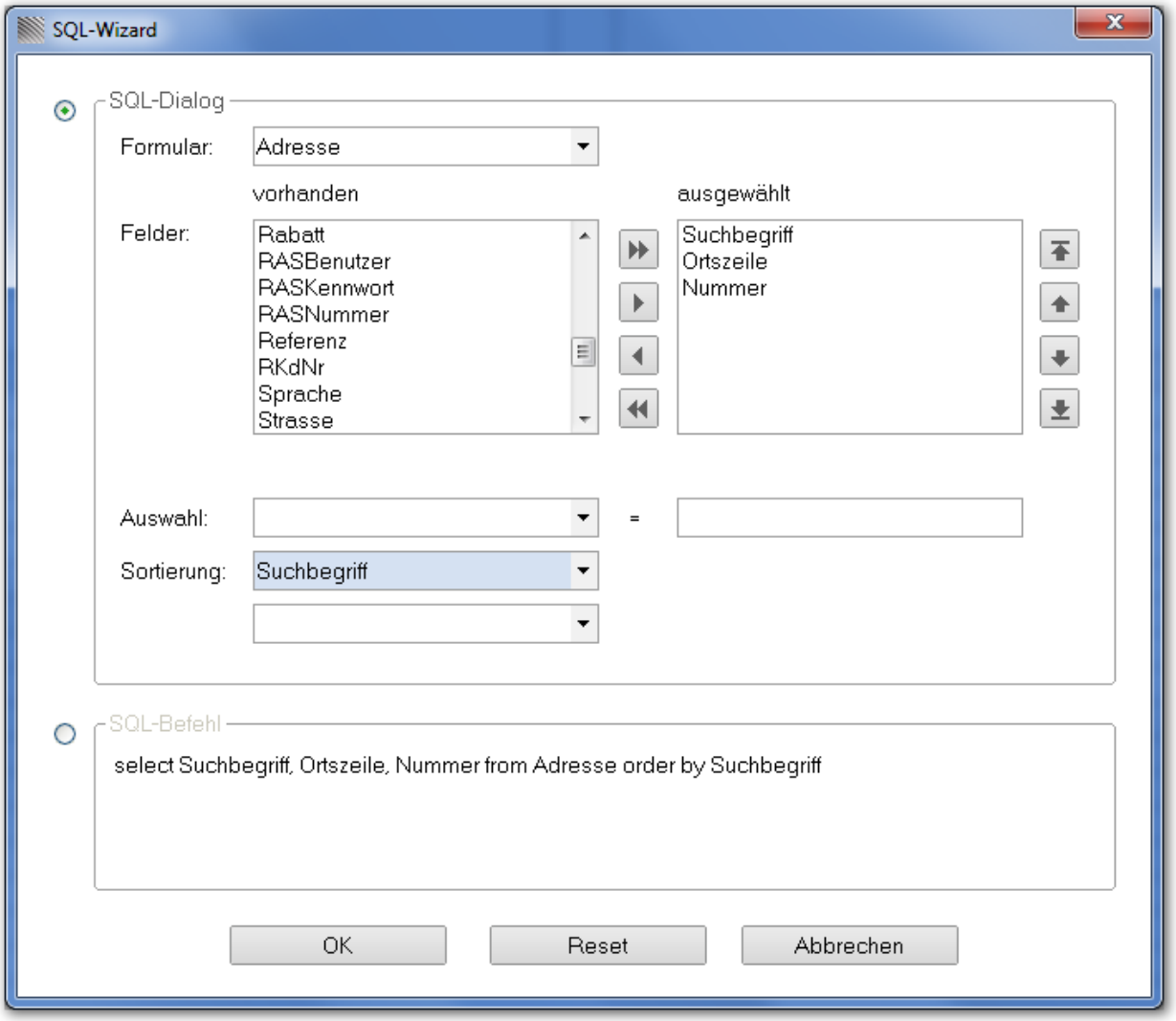


Hierbei kommt der ersten Spalte eine besondere Bedeutung zu: Durch das Tippen von Anfangsbuchstaben können Sie sich dem gewünschten Suchbegriff nähern.

Tipp: Wenn Sie sich in einem Nummernfeld mit Auswahlbutton befinden, dann müssen Sie nicht unbedingt auf den Button klicken um die Auswahl zu öffnen. Sie können einfach einen Buchstaben tippen. Dieser bewirkt dann automatisch eine Öffnung des SQL-Browsers.

Sie können den **SQL-Wizard** zur Erstellung des Auswahlbefehls benutzen, falls Sie noch keine Erfahrungen mit der Abfragesprache SQL besitzen. Klicken Sie hierzu auf das Symbol rechts:

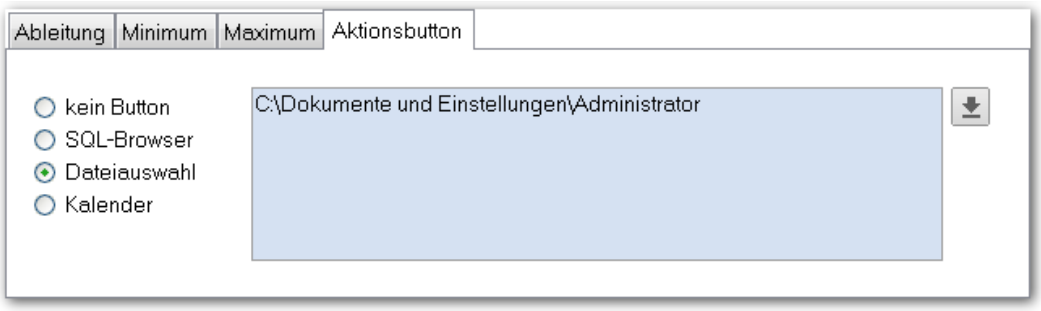
Wählen Sie nun zuerst die Tabelle und dann die Felder aus, die angezeigt werden sollen. In der Regel sollten Sie das erste Feld bei der Sortierung berücksichtigen:



Bereits während der Arbeit mit dem Wizard wird der SQL-Befehl in der unteren Fensterhälfte angezeigt. Sie können auf diese Weise unmittelbar beobachten wie eine Veränderung Ihrer Auswahl den Befehl beeinflusst.

Dateiauswahl

Die **Dateiauswahl** ermöglicht die Auswahl eines Dateinamens. Hierzu wird zur Laufzeit ein kleines Explorerfenster geöffnet. Sie können das Startverzeichnis für den Explorer festlegen:



Kalender

Der Kalenderbutton ermöglicht zur Laufzeit die Auswahl eines Datums aus einem kleinen Kalenderfenster. Sie können diesen Button für Text- und Datumsfelder verwenden.

Internet

Die Option „Internet“ ist nur für Textfelder verfügbar, die als Internet-Adresse gekennzeichnet sind. Zur Laufzeit kann der Benutzer mit einem Button einfach die URL im Browser öffnen.

E-Mail

Die Option „E-Mail“ ist nur für Textfelder verfügbar, die als E-Mail-Adresse gekennzeichnet sind. Zur Laufzeit kann der Benutzer mit einem Button sein E-Mail-Programm starten und ein neues Mail an die Zieladresse verfassen.

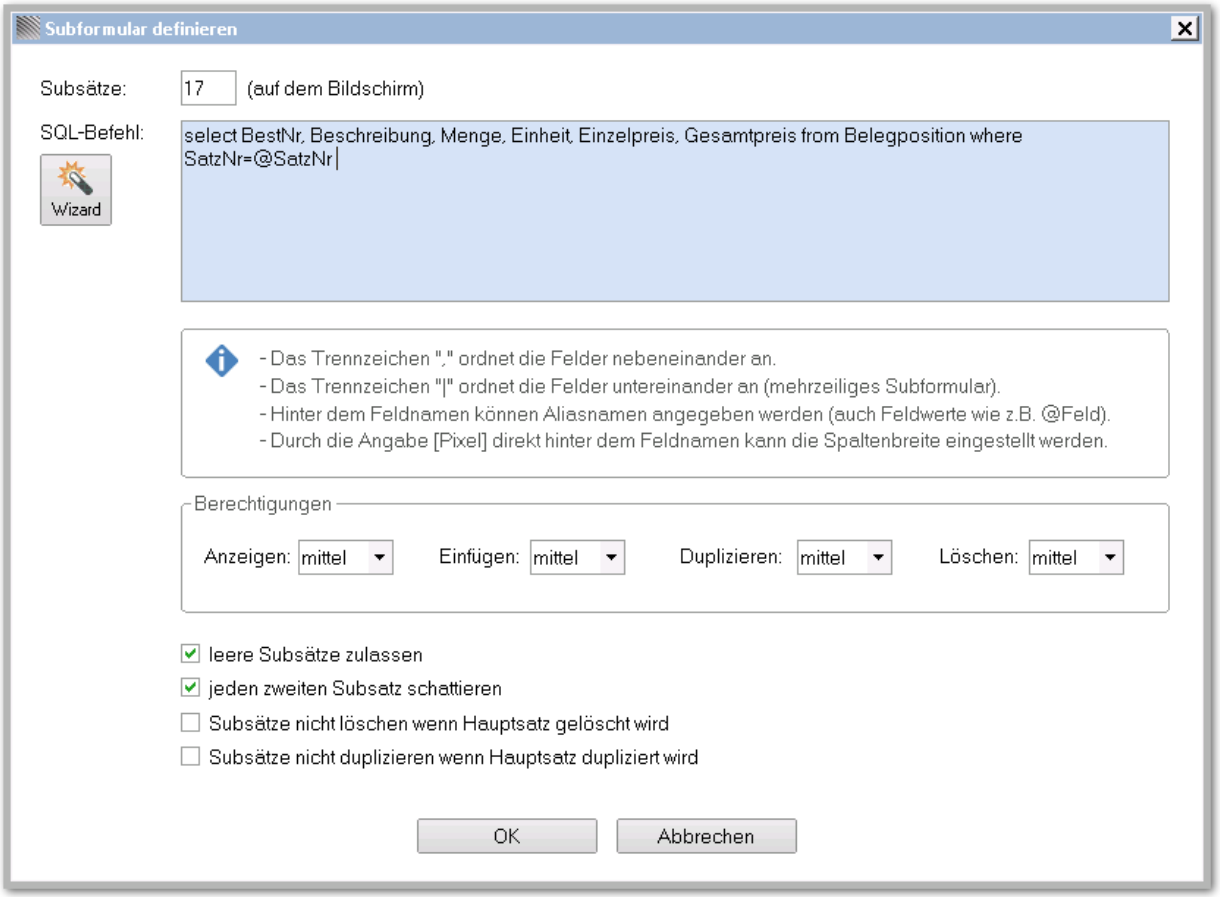
Telefonwahl

Die Option „Telefonwahl“ ist nur für Textfelder verfügbar, die als Telefonnummer gekennzeichnet sind. Der Anwender erhält zur Laufzeit einen Button zur Verfügung gestellt, der eine Telefonnummer automatisch wählt. Zur Nutzung der Wählfunktion benötigten Sie ein TAPI-fähiges Telefon. Zusätzlich muss ein TAPI-Treiber installiert werden und in der Arbeitsplatzkonfiguration von DataCool eingetragen werden. Weitere Einzelheiten entnehmen Sie bitte der Anleitung Ihrer Telefonanlage.

4.2.1.7 Objekt hinzufügen: Subformular

Ein **Subformular** wird zur Abbildung einer **1:N-Beziehung** zwischen zwei Formularen verwendet. Hierbei wird jeweils ein Datensatz im **Hauptformular** mit einer variablen Anzahl von Datensätzen im **Subformular** verknüpft. Das Subformular wird stets in Tabellenform in das Hauptformular eingebettet.

Ein Beispiel aus der Praxis ist ein Rechnungsformular mit einer variablen Anzahl von Rechnungspositionen. Erstellen Sie zuerst beide Formulare einzeln (im nachfolgenden Beispiel: „Beleg“ und „Belegposition“). Danach können Sie das Formular „Belegposition“ als Subformular in das Formular „Beleg“ einbinden. Bewegen den Cursor an die linke untere Ecke und wählen Sie die Option „Subformular“ aus der Toolbar. Der nachfolgende Dialog wird angezeigt:



Die Angabe **Subsätze** bestimmt die Anzahl der Subdatensätze, die auf dem Bildschirm sichtbar sind. Der **SQL-Befehl** dient zur Definition der angezeigten Felder und zur Verknüpfung von Haupt- und Subformular. Das Verknüpfungskriterium wird in der where-Klausel des SQL-Befehls untergebracht. Im obigen Beispiel werden jeweils nur die Positionen angezeigt, bei denen die SatzNr mit der SatzNr des Hauptformulars übereinstimmt.

Die Bedingung lautet also allgemein: „**where Feld=@Feld**“, wobei die Bezeichnung @Feld ein Feld aus dem Hauptformular repräsentiert.

Ab der Version 2010 hat sich die Schreibweise der Verknüpfungsbedingung geändert. Anstelle von „where SatzNr=**Beleg.SatzNr**“ gilt nun die neue Schreibweise „where SatzNr=@**SatzNr**“.

Anders als beispielsweise in DataEase ist für die Erstellung eines Subformulars keine separate Verknüpfung erforderlich, da die Bedingung Teil der Subformulares ist.

Mit Hilfe des SQL-Befehls können Sie nicht nur die Felder auswählen, sondern auch Einfluss auf die Gestaltung des Subformulares nehmen:

- manuelle Einstellung der Feldbreite
- Verwendung von Bildfeldern in Subformularen
- Vergabe von Aliasnamen für Spaltenüberschriften
- mehrzeilige Subformulare (Felder werden untereinander angeordnet)

Beispiel 1: Standard

```
select BestNr, Beschreibung, Menge, Einzelpreis, Gesamtpreis
from Belegposition where SatzNr=@SatzNr
```

Pos	BestNr	Beschreibung	Menge	Einzelpreis	Gesamtpreis
0001					
0002					
0003					
0004					
0005					

Beispiel 2: manuelle Einstellung der Feldbreite

DataCool ermittelt die Spaltenbreiten automatisch. Sie können die Spaltenbreite jedoch beeinflussen, wenn Sie die [Breite in Pixel] direkt hinter dem Feldnamen angeben:

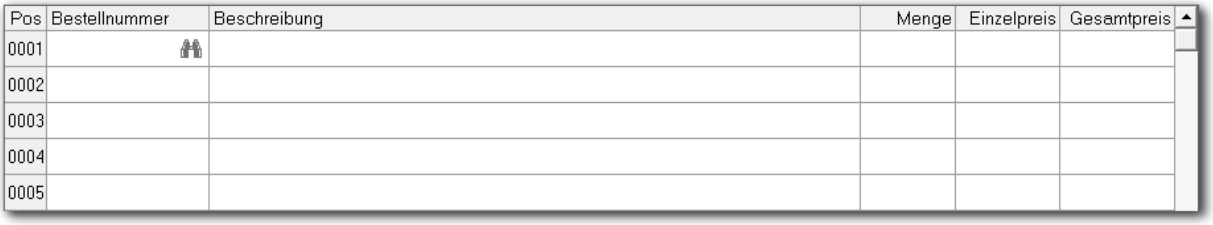
```
select BestNr, Beschreibung[250], Menge, Einzelpreis,
Gesamtpreis from Belegposition where SatzNr=@SatzNr
```

Tipp: Durch Angabe der Spaltenbreite [0] können Sie Felder unsichtbar laden. Dies kann sinnvoll sein, wenn diese Felder für Ableitungsformeln benötigt werden.

Beispiel 3: Verwendung von Aliasnamen

Normalerweise werden die Feldbezeichnungen als Spaltenüberschriften verwendet. Sie können jedoch bei Bedarf auch abweichende Bezeichnungen festlegen, indem Sie hinter dem Feldnamen einen Aliasnamen angeben:

```
select BestNr Bestellnummer, Beschreibung, Menge, Einzelpreis,
Gesamtpreis from Belegposition where SatzNr=@SatzNr
```

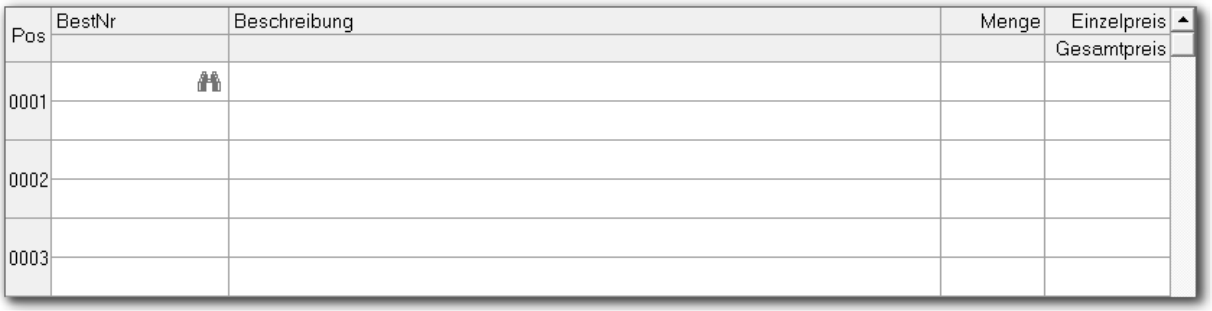


Tipp: Durch Angabe von „-“ als Aliasname können Sie eine Spaltenüberschrift auch komplett abschalten.

Beispiel 4: zweizeiliges Layout

In der Regel entspricht ein Subdatensatz genau einer Zeile in der Tabelle. Bei Bedarf können Sie jedoch auch mehrzeilige Subdatensätze definieren, wenn der Platz am Bildschirm nicht ausreicht. In diesem Fall ersetzen Sie einfach das Komma zwischen den Feldern durch das Feldtrennzeichen „|“. Felder, die mit diesem Zeichen voneinander getrennt sind, werden im Subformular untereinander angeordnet:

```
select BestNr, Beschreibung, Menge, Einzelpreis | Gesamtpreis
from Belegposition where SatzNr=@SatzNr
```



Beispiel 5: Verwendung eines Bildfeldes

Bei Bedarf können Sie sogar Bildfelder in das Subformular integrieren. Das Subformular wird dann automatisch mehrzeilig. Es kann maximal ein Bildfeld eingebunden werden.

```
select BestNr, Bezeichnung, Abbildung from Belegposition where
SatzNr=@SatzNr
```

Pos	BestNr	Bezeichnung	Abbildung
0001	st42s	AGFEO Systemtelefon ST42, silber	
0002	hs920	AGFEO System-Headset 920	

Beispiel 6: Angabe eines Sortierkriteriums

Bei Bedarf können Sie eine Sortierung für das Subformular vorgeben. Verwenden Sie hierzu einfach die Klausel „order by“ in der Subformulardefinition:

```
select BestNr, Beschreibung, Menge, Einzelpreis, Gesamtpreis
from Belegposition where SatzNr=@SatzNr order by BestNr
```

Pos	BestNr	Beschreibung	Menge	Einzelpreis	Gesamtpreis
0001					
0002					
0003					

NEU

Ab der Version 2014 können Sie die vordefinierte Sortierung ändern. Klicken Sie einfach mit der Maus auf die Spalte, die Sie sortieren möchten. Die aktuelle Sortierspalte wird mit blauer Farbe hervorgehoben. Durch erneutes Anklicken einer Sortierspalte können Sie die Sortierreihenfolge umkehren.

Wenn Sie kein Sortierfeld vorgeben, dann wird die Sortierfunktion abgeschaltet und DataCool stellt sicher, dass die Datensätze stets in der eingegebenen Reihenfolge erscheinen.

Berechtigungen

Anzeigen	Im Anwendungsmodus kann der Benutzer den Subdatensatz im Vollbildmodus anzeigen, in dem er auf die jeweilige Positionsnummer klickt. Die Anzeigeberechtigung bestimmt, welche Benutzerstufe für diese Aktion minimal erforderlich ist.
Einfügen	Min. Benutzerstufe zum Hinzufügen von Subdatensätzen.
Löschen	Min. Benutzerstufe zum Löschen von Subdatensätzen.
Duplizieren	Min. Benutzerstufe zum Duplizieren von Subdatensätzen.

Weitere Subformular-Einstellungen

leere Subsätze zulassen:

Normalerweise werden leere Tabellenzeilen nicht als Subsätze in der Datenbank gespeichert. In Einzelfällen (z.B. bei der Erfassung von Trennzeilen zwischen Rechnungspositionen) kann dies jedoch erwünscht sein.

jeden 2. Subsatz schattieren:

Jede zweite Zeile in der Tabelle wird mit einer Grauschattierung versehen.

Subsätze nicht löschen wenn Hauptsatz gelöscht wird:

Aktivieren Sie diese Option um zu Verhindern, dass beim Löschen des Hauptdatensatzes auch die Subdatensätze mitgelöscht werden.

Subsätze nicht duplizieren wenn Hauptsatz dupliziert wird

Aktivieren Sie diese Option um zu Verhindern, dass beim Duplizieren des Hauptdatensatzes auch die Subdatensätze dupliziert werden.

Beim Duplizieren eines Hauptdatensatzes werden die Subsätze zwar mitdupliziert, aber nur mit den Daten die am Bildschirm zu sehen sind.

4.2.1.8 Objekte bearbeiten

Sie können ein Element aus dem Formular jederzeit ändern. Klicken Sie hierzu mit der **rechten Maustaste** auf das Objekt und wählen Sie anschließend die Option „bearbeiten“ aus dem Kontextmenü. Mit der Tastenkombination Strg+B können Sie ebenfalls in den Bearbeitungsmodus wechseln, wenn sich der Cursor an der linken unteren Ecke eines Objektes befindet.

Sie erkennen den Bearbeitungsmodus an der **roten Cursorfarbe**. Je nach Feldtyp stehen Ihnen unterschiedliche Bearbeitungsfunktionen zur Verfügung:

Feldtyp	Bearbeitungsfunktion
Text	Text ändern
Linie	Länge ändern
Rechteck	Ecke verschieben
Button	Breite oder Höhe verändern. Zur Änderung der Beschriftung oder der Aktion wählen Sie bitte „Aktion definieren“ aus dem Kontextmenü.
Bild	Bild austauschen. Zur Änderung der Tooltip-Hilfe oder der Aktion wählen Sie bitte „Aktion definieren“ aus dem Kontextmenü.
Feld	Veränderung der Feldeigenschaften. Beim Speichern des Formulars werden bereits vorhandene Daten in der Tabelle automatisch an die neue Felddefinition angepasst. Hierbei findet bei Bedarf eine Konvertierung der SQL-Datentypen statt (z.B. Zahl nach Text). Sollte die Konvertierung nicht möglich sein, so kann das Formular nicht gespeichert werden.
Subformular	Änderung der Subformulardefinition (z.B. Änderung der Spaltenauswahl oder der Sortierreihenfolge).

Mit der Taste ESC können Sie den Bearbeitungsvorgang abbrechen ohne die Änderungen zu übernehmen. Mit Enter wird die Änderung übernommen. Änderungen an der Feld- oder Subformulardefinition müssen Sie mit ***OK*** oder F2 bestätigen.

4.2.1.9 Objekte verschieben

Sie können ein Element aus dem Formular jederzeit verschieben. Klicken Sie hierzu mit der **rechten Maustaste** auf das Objekt und wählen Sie die Option „verschieben“ aus dem Kontextmenü. Mit der Tastenkombination Strg+M-können Sie ebenfalls in den Verschiebemodus wechseln, wenn sich der Cursor an der linken unteren Ecke eines Objektes befindet.

Sie erkennen den Verschiebemodus an der **gelben Cursorfarbe**.

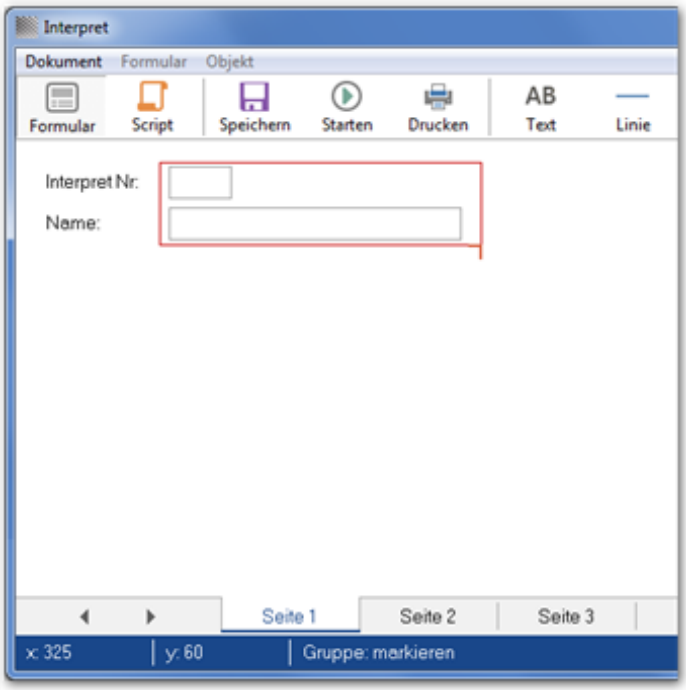
Mit den Pfeiltasten bewegen Sie das Objekt an die jeweils nächstgelegene Rasterposition. Das Raster besitzt einen horizontalen Abstand von 15 Pixel und einen vertikalen Abstand von 30 Pixel. Mit Tab bzw. Shift-Tab bewegen Sie das Objekt an die nächste Tabulatorposition. Mit der Tastenkombination Strg+Pfeiltaste springen Sie an den jeweiligen Bildschirmrand.

Bei Bedarf lassen sich Objekte auch pixelgenau verschieben: Verwenden Sie hierzu die Tastenkombination Shift+Pfeiltaste. Mit Hilfe der Tasten Bild Auf bzw. Bild Ab können Sie Objekte über Bildschirmseiten hinweg verschieben.

Mit ESC können Sie den Verschiebevorgang abrechen. Das Objekt springt dann an seine Ursprungsposition zurück. Mit Enter wird das Objekt an seiner neuen Position verankert.

Gruppen verschieben

Wenn Sie mehrere Objekte auf einmal verschieben möchten, dann müssen Sie zuvor eine **Gruppe markieren**.



Bewegen Sie den Cursor an die linke untere Auswahlposition und wählen Sie **Objekt ⇨ Gruppe markieren** oder drücken Sie Strg+G. Ziehen Sie nun mit den Cursortasten das rote Rechteck über den gewünschten Bereich. Anschließend bestätigen Sie die Auswahl mit Return. Das Rechteck verfärbt sich nun gelb und der Cursor springt an die linke untere Ecke des Auswahlbereiches. Sie können nun alle Objekte der Gruppe gemeinsam verschieben.

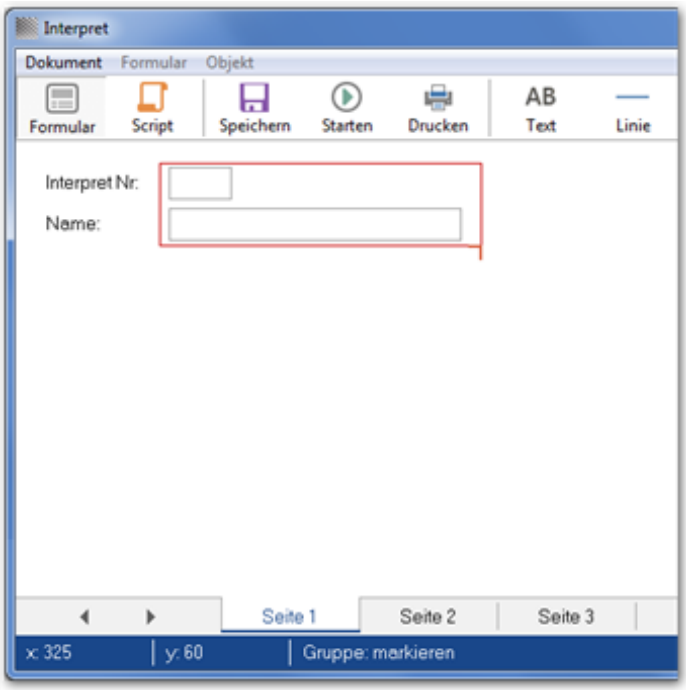
4.2.1.10 Objekte löschen

Klicken Sie mit der **rechten Maustaste** auf das Objekt und wählen Sie die Option „löschen“ aus dem Kontextmenü. Sie können auch die Taste Entf zum Löschen benutzen, wenn sich der Cursor an der linken unteren Ecke eines Objektes befindet.

Wenn Sie ein Feld aus der Maske löschen, dann gehen eventuell bereits erfasste Daten in diesem Feld beim Abspeichern des Formulars unwiderruflich verloren. Vor der Entfernung des Feldes erfolgt deshalb eine Sicherheitsabfrage.

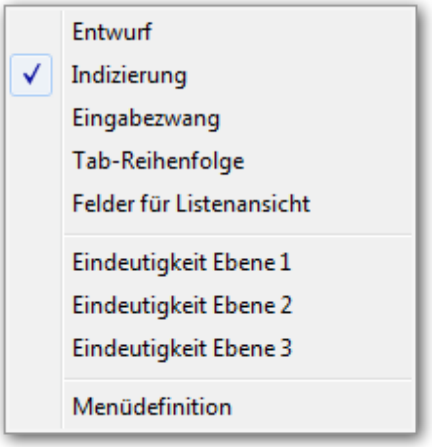
Gruppe löschen

Wenn Sie mehrere Objekte auf einmal löschen möchten, dann müssen Sie zuvor eine **Gruppe markieren**.



Bewegen Sie den Cursor an die linke untere Auswahlposition und wählen Sie **Objekt ⇨ Gruppe markieren** oder drücken Sie Strg+G. Ziehen Sie nun mit den Cursortasten das rote Rechteck über den gewünschten Bereich. Anschließend bestätigen Sie die Auswahl mit Return. Das Rechteck verfärbt sich nun gelb und der Cursor springt an die linke untere Ecke des Auswahlbereiches. Sie können nun mit der Taste Entf alle Objekte auf einmal löschen.

4.2.2 Formular: Indizierung



Mit der Option **Formular ⇨ Indizierung** gelangen Sie in die Indexansicht des Formulareditors. In dieser Ansicht können Sie keine Objekte hinzufügen oder ändern. Stattdessen können Sie durch einen Mausklick auf ein Feld einen Index hinzufügen oder entfernen. Felder mit Index werden farbig hervorgehoben. Sie können einen Index auch direkt in den Feldeigenschaften aktivieren. Die spezielle Indexansicht im Editor verschafft Ihnen jedoch einen optimalen Überblick über alle indizierten Felder eines Formulars.

Was ist ein Index?

Ein Index ist eine sortierte Liste, die zur Optimierung von Suchvorgängen von der Datenbank bereitgehalten wird.

Was sind die Vor- und Nachteile der Indizierung?

Wenngleich ein Index die Suchvorgänge beschleunigt, so benötigt er andererseits zusätzlichen Speicherplatz in der Datenbank. Darüberhinaus muss der Index stets aktuell gehalten werden. Dies kann beim Hinzufügen, Ändern oder Löschen von Datensätzen zu einer Verlangsamung führen. Sie sollten daher stets gründlich überlegen, ob Sie ein Feld indizieren oder nicht. Die geschickte Auswahl der Indexfelder ist ein wichtiger Aspekt bei der Anwendungsentwicklung.

Welche Felder sollten indiziert werden?

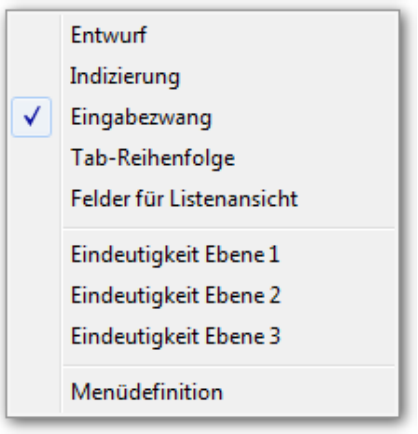
Felder in denen häufig recherchiert wird sollten Sie mit einem Index ausstatten. Dies gilt insbesondere für Schlüsselfelder die der Verknüpfung von Formularen dienen.

Wann kann auf einen Index verzichtet werden?

Wenn Sie nur gelegentlich in einem Feld suchen, dann benötigen Sie keinen Index. Sie können außerdem auf einen Index verzichten, wenn ein Feld lediglich als zweites oder drittes Selektionsmerkmal dient. Besteht ein Selektionskriterium aus mehreren indizierten Feldern, dann wird oftmals nur ein Index bei der Suche genutzt.

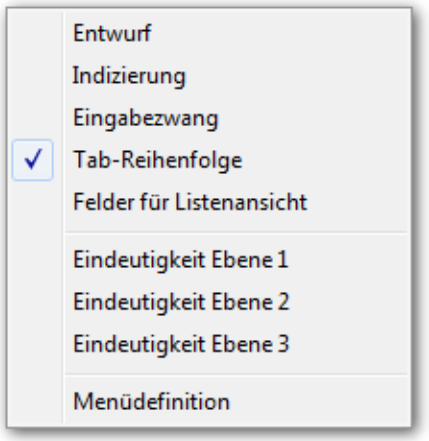
Formulare, in denen nicht mehr als etwa 1.000 Sätze gespeichert werden, sollten Sie grundsätzlich nicht indizieren. Bei kleinen SQL-Tabellen erfolgt der Zugriff auch ohne Index schnell genug. Dies gilt insbesondere für Konfigurationsformulare, die ja lediglich einen Datensatz besitzen.

4.2.3 Formular: Eingabezwang

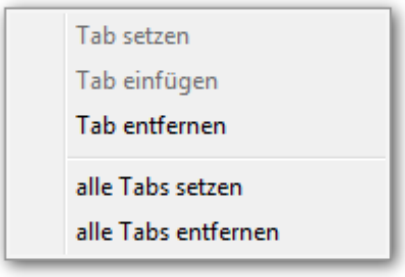


Mit der Option **Formular ⇨ Eingabezwang** können Sie auf einfache Weise die Pflichtfelder eines Formulars festlegen. Alle Felder mit Eingabezwang werden farbig markiert dargestellt. Sie können den Eingabezwang durch Anklicken eines Feldes mit der Maus einfach ein- oder ausschalten.

4.2.4 Formular: Tab-Reihenfolge



Die Option **Formular ⇨ Tab-Reihenfolge** gestattet die Festlegung der Tabulator-Reihenfolge in einem Formular. Hierzu klicken Sie einfach die Felder in der gewünschten Reihenfolge an. Jedes Feld erhält beim Anklicken eine Positionsnummer. Für Felder ohne Tabstopp gilt die sog. Serpentinen-Reihenfolge. Hierbei werden die Felder von oben nach unten und von links nach rechts aktiviert. Felder ohne Tabposition werden stets nach den Feldern mit Tabposition angeordnet.



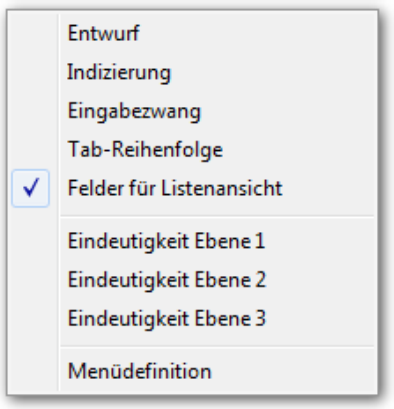
Für nachträgliche Änderungen steht eine Reihe von Zusatzfunktionen zur Verfügung. Klicken Sie hierzu mit der **rechten Maustaste** auf ein Feld und wählen Sie eine Option aus dem Kontextmenü:

Wählen Sie „**Tab entfernen**“ um einen Tabulator zu entfernen.

Mit der Option „**alle Tabs setzen**“ erhalten alle Felder eine Positionsnummer gemäß Serpentinen-Reihenfolge. Sollten bereits Felder mit Tabpositionen existieren, so bleiben diese unverändert erhalten. Die Zählung beginnt dann mit der nächsten freien Positionsnummer. Mit der Option „**alle Tabs entfernen**“ werden alle Tabpositionen auf der aktuellen Bildschirmseite entfernt.

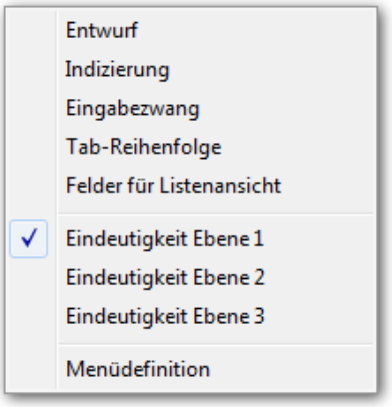
Sie können auch Felder nachträglich in die Tabreihenfolge aufnehmen. Klicken Sie hierzu zunächst auf das **nachfolgende** Feld um es zu markieren. Klicken Sie anschließend mit der rechten Maustaste auf das neue Feld und wählen Sie die Option „**Tab einfügen**“. Das neue Feld erhält die Nummer des zuvor markierten Feldes. Alle nachfolgenden Felder werden automatisch neu nummeriert.

4.2.5 Formular: Felder für Listenansicht



Mit der Option **Formular ⇨ Felder für Listenansicht** können Sie auf einfache Weise bestimmen, welche Felder in der Listenansicht einer Maske angezeigt werden. Felder, die in der Listenanzeige enthalten sind werden farbig markiert. Sie können ein Feld durch Anklicken mit der Maus hinzufügen oder entfernen.

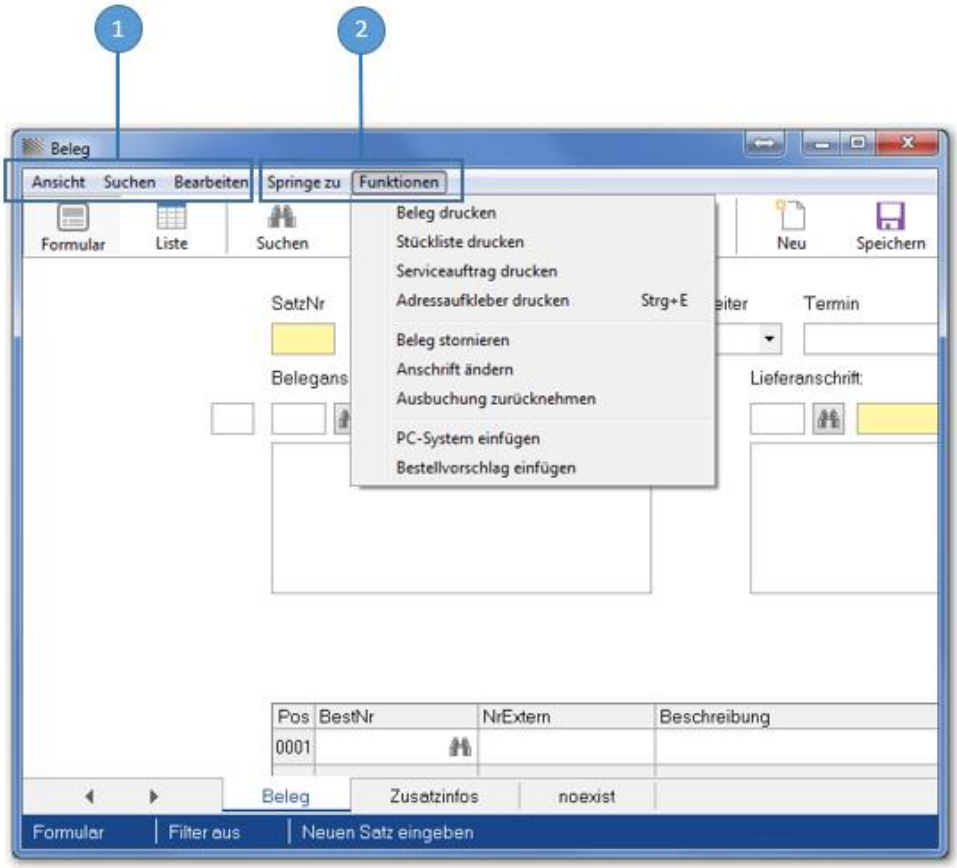
4.2.6 Formular: Eindeutigkeit



Mit der Option **Formular ⇨ Eindeutigkeit** können Sie ein oder mehrere Felder definieren, die gemeinsam eindeutig sind. Die Datenbank prüft dann beim Abspeichern eines Datensatzes ob bereits ein Datensatz mit gleichen Eindeutigkeitsfeldern existiert. Sind alle als eindeutig definierten Felder gleich, so wird der Datensatz als Dublette angesehen und kann nicht gespeichert werden. Die Festlegung der Eindeutigkeit erfolgt durch einen Mausklick auf die gewünschten Felder. Alle eindeutigen Felder werden hierbei farbig markiert.

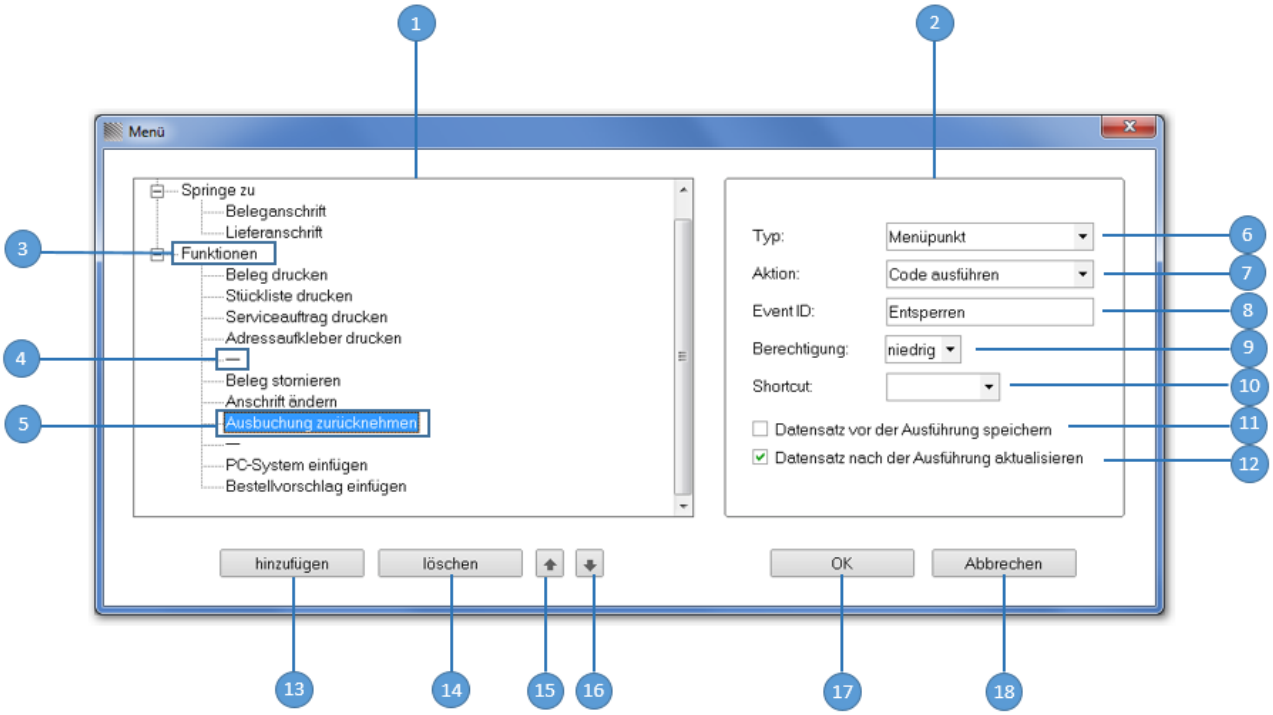
Es stehen insgesamt drei voneinander unabhängige Eindeutigkeitsebenen zur Verfügung. Die farbig markierten Felder einer Ebene bilden ein **gemeinsames** Eindeutigkeitsmerkmal. Ein Datensatz wird nur dann als doppelt angesehen, wenn **alle** Felder gleich sind.

4.2.7 Formular: Menüdefinition



DataCool Formulare besitzen eine Standard-Menüzeile mit den Optionen „Ansicht“, „Suchen“ und „Bearbeiten“ (Bereich 1). Sie können diese Menüzeile um eigene Funktionen erweitern. Die nebenstehende Abbildung zeigt ein Formular mit zwei Zusatzoptionen (Bereich 2).

Zur Gestaltung eines benutzerdefinierten Menüs starten Sie den Menüeditor über **Formular ⇨ Menü definieren**:



1 Menübaum (Treeview)

2 Festlegung der Optionseigenschaften

3 Untermenü (enthält untergeordnete Menüpunkte)

4 Trennlinie (zur Strukturierung der Menüanzeige)

5 Bezeichnung der Menüoption. Die Änderung einer Menübezeichnung klicken Sie auf den Text und drücken Sie die Taste F2.

6 Wählen Sie die Art der Menüposition: Untermenü, Menüpunkt oder Trennlinie.

7 Wählen Sie die Aktion eines Menüpunktes:

keine Aktion	Wählen Sie diese Option, wenn Sie den Menüeintrag erst zu einem späteren Zeitpunkt mit einer Aktion belegen wollen.
Menü öffnen	Öffnet das ausgewählte Dokument vom Typ „Menü“.
Formular öffnen	Öffnet das ausgewählte Formular.
Script starten	Startet das ausgewählte Script.
Import starten	Startet einen Datenimport auf der Basis der angegebenen Importdefinition.
Export starten	Startet einen Datenexport auf der Basis der angegebenen Exportdefinition.

Code ausführen	Startet das im Dokument eingebettete Script. Als „Event ID“ können Sie einen freien Text festlegen, der innerhalb des Scriptes mit der Systemvariable **current.event** abgefragt werden kann. Auf diese Weise kann das Script auf das eingetretene Ereignis reagieren.
Auswahl definieren	Eingabefelder vom Typ „Auswahl“ stellen dem Anwender eine Reihe vordefinierter Optionen zur Verfügung. Mit der Aktion „Auswahl definieren“ geben Sie dem Benutzer die Möglichkeit zur Veränderung einer Auswahlliste.
Verknüpfung öffnen	Eine Verknüpfung ist eine Beziehung (Relation) zwischen zwei Formularen. Beim Ausführen dieser Aktion wird das angegebene Formular mit dem verknüpften Datensatz geöffnet. Die Funktionsweise von Verknüpfungen wird im Kapitel 4.4 näher erläutert.

8 Wählen Sie hier das zu startende Dokument aus oder vergeben Sie einen Namen für das Ereignis, falls Sie die Option „Code ausführen“ gewählt haben.

9 Geben Sie die minimale Berechtigungsstufe an, die ein Benutzer zum Ausführen der Aktion benötigt.

10 Optional: Wählen Sie eine Funktionstaste oder eine Tastenkombination zum Ausführen der Aktion. Der Shortcut „Druckbutton“ nimmt eine Sonderstellung ein: er überlagert die normale Satzdruck-Funktion von DataCool.

11 Diese Option bewirkt, dass der Datensatz **vor** der Ausführung der Aktion automatisch gespeichert wird. Sollte das Speichern fehlschlagen, so wird auch die Aktion nicht ausgeführt.

12 Diese Option bewirkt, dass der Datensatz **nach** der Ausführung der Aktion frisch aus der SQL-Tabelle geladen wird. Sollte die Aktion (z.B. ein Scriptlauf) den Datensatz verändert haben, so wird hierdurch die Änderung direkt am Bildschirm reflektiert.

13 Hinzufügen eines neuen Menüeintrags unterhalb der markierten Stelle. Neue Menüeinträge können auch mit der Taste Einfg hinzugefügt werden.

14 Löscht den markierten Menüeintrag aus dem Menübaum. Menüeinträge können auch mit der Taste Entf aus dem Baum entfernt werden.

15 Verschiebt den markierten Menüeintrag nach oben.

16 Verschiebt den markierten Menüeintrag nach unten.

17 Übernimmt die Änderungen und beendet den Menüdialog.

18 Bricht die Bearbeitung des Menüs ab. Alle Änderungen gehen verloren.

4.3 Änderung eines Formulars

Bei der Erstellung eines Formulars wird von DataCool automatisch eine passende SQL-Tabelle erzeugt. Dies ist ein Vorteil im Vergleich zu vielen anderen Systemen, bei denen Tabellenentwurf und Maskengestaltung getrennte Arbeitsschritte sind.

Wie ändere ich ein bestehendes Formular?

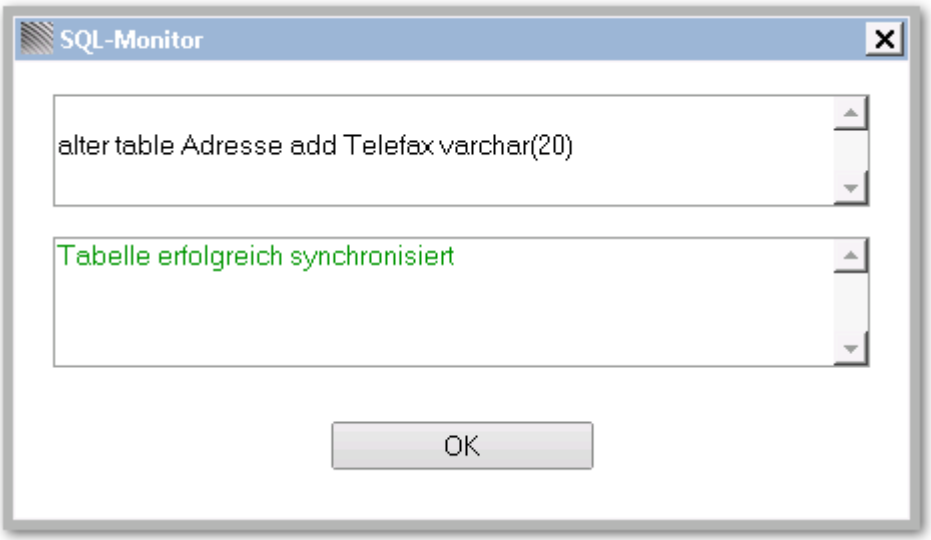
Klicken Sie mit der rechten Maustaste auf ein Dokument in der Formularliste des Hauptmenüs. Wählen Sie anschließend die Option „Bearbeiten“ aus dem Kontextmenü. Sie können Formulare auch mit der Taste F4 im Entwicklungsmodus öffnen.

Was passiert mit den eingegebenen Daten bei einer Formularänderung?

Änderungen an der Eingabemaske, die keine Felder betreffen, haben auch keinen Einfluss auf die zugehörige SQL-Tabelle. Wenn Sie also beispielsweise Änderungen an der Feldbeschriftung vornehmen oder eine Linie hinzufügen, so muss die zugrunde liegende SQL-Tabelle nicht angepasst werden.

Sobald Sie jedoch Felder hinzufügen, ändern oder löschen, muss auch die SQL-Tabelle an die neue Datenstruktur angepasst werden. Hierbei kommen grundsätzlich zwei unterschiedliche Techniken zum Einsatz. Enthält die SQL-Tabelle noch keine Daten, so wird die Tabelle komplett gelöscht und neu erzeugt. Sind bereits Daten erfasst worden, so wird die SQL-Tabelle mit Hilfe von Synchronisierungsbefehlen an die neue Formularstruktur angepasst.

Beispiel: Sie fügen ein Feld mit der Bezeichnung „Telefax“ zum Formular „Adresse“ hinzu. Bei Abspeichern synchronisiert DataCool die bestehende SQL-Tabelle mit dem Formular. Hierzu wird nachträglich die Spalte „Telefax“ zur SQL-Tabelle hinzugefügt. Dieser Vorgang wird im SQL-Monitorfenster dokumentiert:



Die Anpassung der SQL-Tabelle mit Hilfe von Synchronisierungsbefehlen erfolgt sehr schnell. Selbst wenn bereits große Datenmengen in der Tabelle gespeichert sind benötigt der SQL-Server kaum mehr als einige Sekunden. Dies ist ein großer Vorteil im Vergleich zu Datei-basierten Datenbanksystemen, die bei Änderungen einen zeitaufwändigen Reorganisationslauf benötigen.

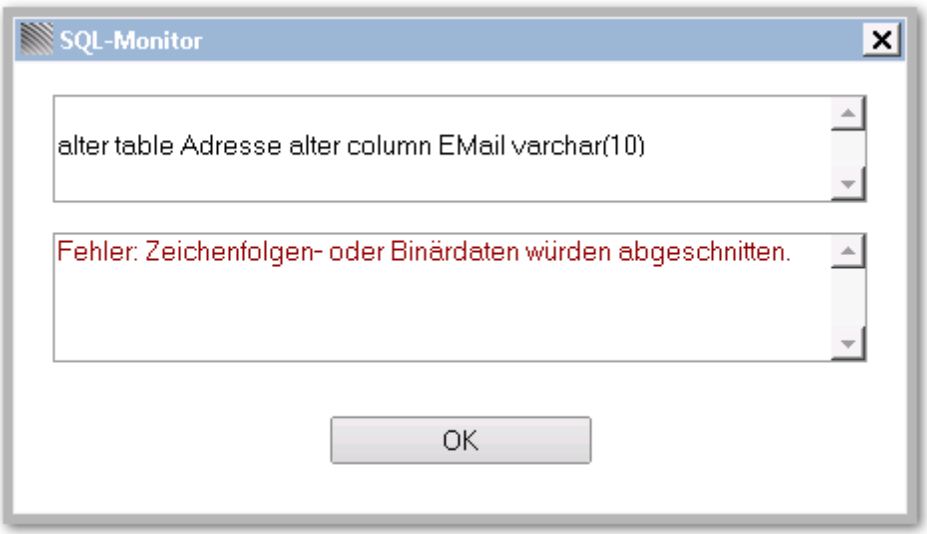
Viele Änderungen können Sie online ausführen – selbst wenn andere Benutzer gleichzeitig mit der Vorgängerversion des Formulars arbeiten. Dies gilt insbesondere dann, wenn Sie nur Felder hinzufügen oder Feldlängen erweitern. Andere Benutzer sollten die Maske verlassen, sobald Sie Felder entfernen oder verkürzen.

Fehlermeldungen bei der Synchronisation der SQL-Tabelle

Das Hinzufügen oder Löschen von Feldern ist generell unproblematisch. Änderungen an einer Felddefinition erfordern jedoch eine Konvertierung von Daten. So kann beispielswiese ein Zahlenfeld jederzeit in ein Textfeld gewandelt werden. Der umgekehrte Weg setzt jedoch voraus, dass alle erfassten Datensätze einen gültigen Zahlenwert im Textfeld besitzen.

Der SQL-Server führt die Synchronisationsbefehle nur dann aus, wenn jeder vorhandene Datensatz verlustfrei konvertiert werden kann. Die Speicherung der Formulardefinition und die Datenkonvertierung erfolgt innerhalb einer SQL-Transaktion. Diese bedeutet, dass die Änderungen an der Maske nur gespeichert werden, wenn auch die Anpassung der Daten fehlerfrei durchgeführt werden kann.

Beispiel: Sie versuchen das Textfeld „EMail“ von der ursprünglichen Länge (50 Zeichen) auf 10 Zeichen zu verkürzen. Die Tabelle enthält jedoch Datensätze mit E-Mail-Adressen, die länger als 10 Zeichen sind. Die Synchronisation scheitert, da das Abschneiden des Feldes zum Verlust von Daten führen würde. Sie müssen die Bearbeitung der Maske abbrechen ohne zu Speichern.

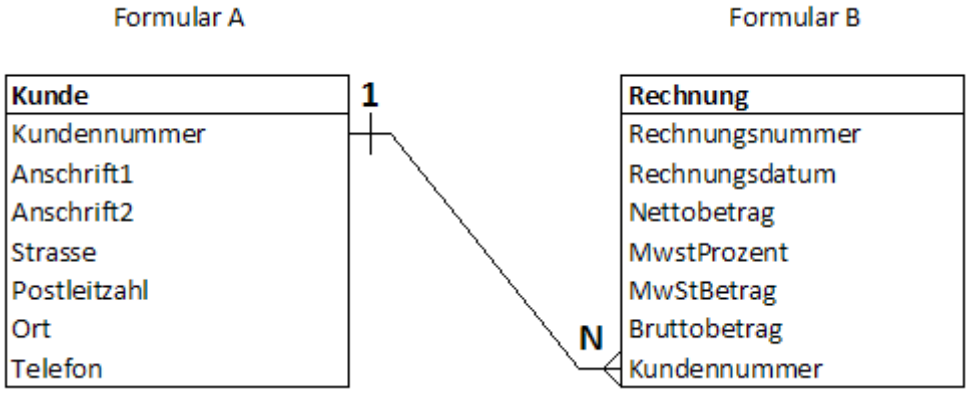


4.4 Verknüpfungen

Eine Verknüpfung ist eine Beziehung zwischen zwei Tabellen. Hierbei werden zusammengehörige Datensätze in den Tabellen anhand gleicher Schlüsselfelder identifiziert.

Beispiel:

Im Formular A werden Kundenadressen gespeichert. Das Formular B dient zur Speicherung von Rechnungen. Für jeden Kunden im Formular A kann es beliebig viele Rechnungen im Formular B geben. Jede Rechnung hat jedoch nur einen zugehörigen Kundendatensatz. Man spricht in diesem Fall von einer sog. 1:N-Beziehung.



Wie wird eine Verknüpfung erstellt?

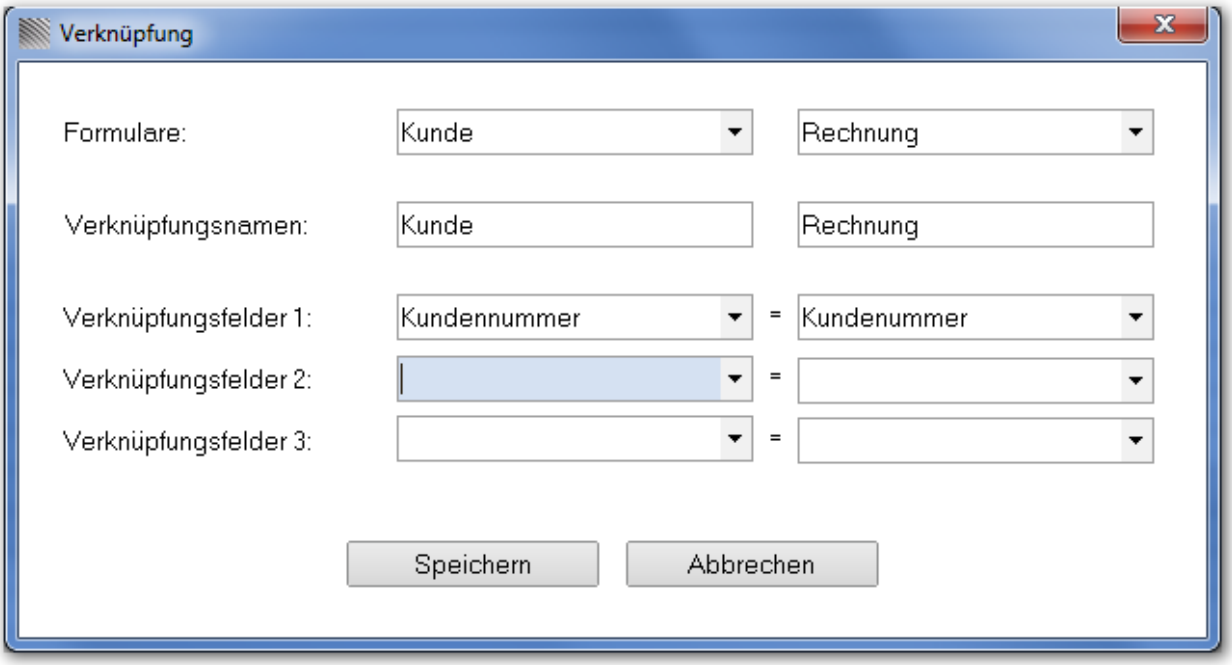
Als erstes müssen Sie die zu verknüpfenden Formulare erstellen. In jedem Formular muss es ein Schlüsselfeld geben, das als Verknüpfungsmerkmal dient. In obigem Beispiel dient hierzu das Feld Kundennummer. Die Felder können unterschiedliche Bezeichner haben, müssen aber vom gleichen Typ sein.

Beim Auflisten von verknüpften Datensätzen muss die Datenbank eine Suche anhand der Schlüsselfelder ausführen. Die **Schlüsselfelder** einer Verknüpfung sollten daher in beiden Formularen mit einem **Index** versehen werden.

Bitte wählen Sie die Option ***Verknüpfungen*** im Hauptmenü von DataCool. Klicken Sie anschließend auf den Button ***Neu*** oder drücken Sie die Taste Einfg um eine neue Verknüpfung zu erstellen.

Tipp: Sie können eine Verknüpfung auch über das Kontextmenü der Formularliste erstellen. Klicken Sie mit der rechten Maustaste auf eines der beiden Verknüpfungsformulare und wählen Sie **Verknüpfungen ⇨ Neue Verknüpfung**.

So definieren Sie die Verknüpfung aus dem vorangegangenen Beispiel:



Formulare: Wählen Sie die beiden Formulare aus, die miteinander verknüpft werden sollen.

Verknüpfungsnamen: Als Verknüpfungsname dient normalerweise der Formularname. Sie können jedoch mehr als nur eine Verknüpfung zwischen zwei Formularen definieren. In diesem Fall müssen Sie einen eindeutigen Bezeichner für Ableitungsformeln vergeben.

Verknüpfungsfelder 1: Wählen Sie die Schlüsselfelder in beiden Formularen aus. Die Felder müssen vom gleichen Typ sein und sollten indiziert werden.

Verknüpfungsfelder 2: Definieren Sie hier ggf. ein zweites Schlüsselfeldpaar. Eine Verknüpfung von Datensätzen findet nur statt, wenn alle Schlüsselfeldpaare übereinstimmen.

Verknüpfungsfelder 3: Definieren Sie hier ggf. ein drittes Schlüsselfeldpaar. Eine Verknüpfung von Datensätzen findet nur statt, wenn alle Schlüsselfeldpaare übereinstimmen.

Sie können auch Verknüpfungen ohne Schlüsselfelder definieren. Diese Art der Verknüpfung wird häufig für Konfigurationsformulare verwendet. Hier ist keine Selektion erforderlich, da es nur einen einzelnen Datensatz gibt.

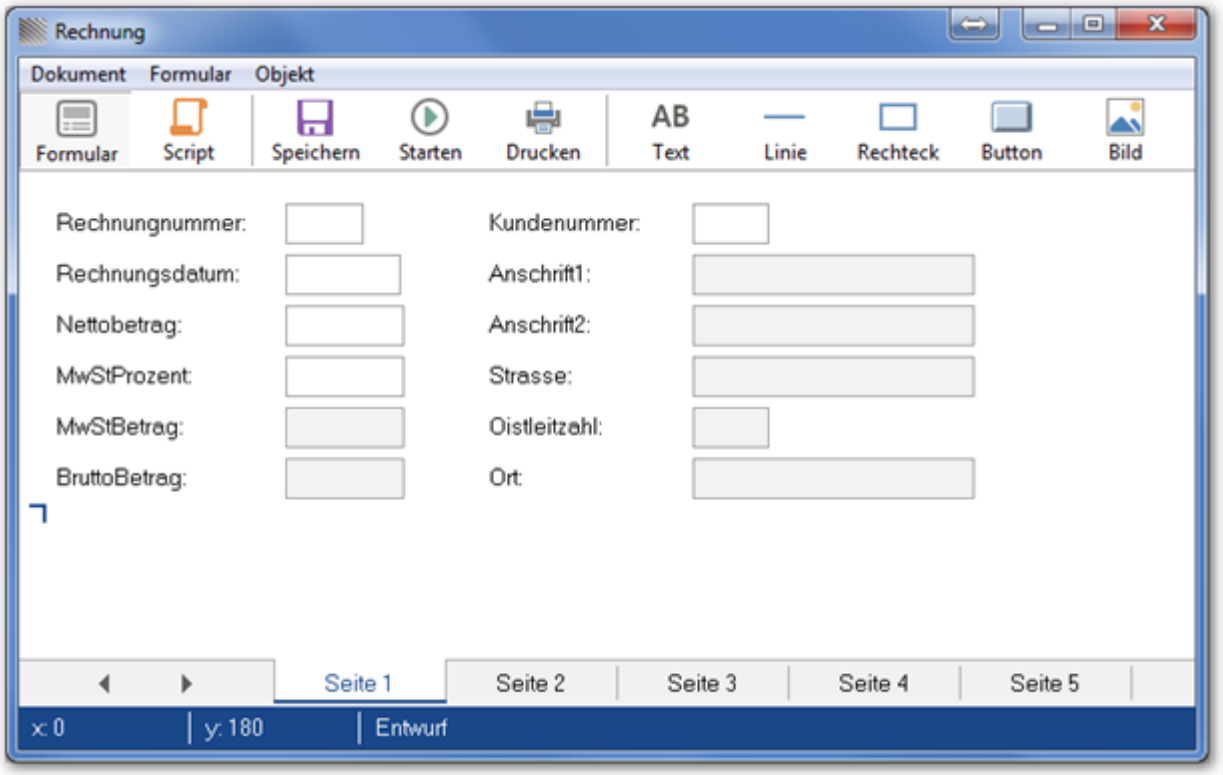
Nachdem die Verknüpfung erstellt wurde können Sie auf einfache Weise auf Informationen aus anderen Formularen zugreifen. Die Verwendung der Verknüpfung in Ableitungsformeln wird im nächsten Kapitel erläutert.

4.5 Ableitungsformeln

Eine Ableitung veranlasst DataCool, den Wert eines Feldes automatisch zu bestimmen. Die Berechnung des Feldwertes erfolgt mit Hilfe einer **Ableitungsformel**. Eine Formel kann aus Konstanten, Feldnamen, Systemvariablen, Funktionen oder Operatoren bestehen. Die Feldnamen können sich auf Felder des gleichen Formulars oder auf Felder aus einem verknüpften Formular beziehen. Es stehen Ihnen alle Funktionen zur Verfügung, die im Referenzteil beschrieben werden.

Beispiel:

Die nachfolgende Abbildung zeigt das in Kapitel 4.4 beschriebene Kundenformular. Die grau gefärbten Felder sind virtuell. Dies bedeutet, dass die Feldinformationen nicht der SQL-Tabelle gespeichert werden. Die Maske soll nun mit einigen Ableitungsformeln versehen werden. Hierzu rufen Sie die Eigenschaften des jeweiligen Feldes auf (rechte Maustaste ⇨ bearbeiten) und tragen die Formel im Register „Ableitung“ ein.



Einige Beispiele für sinnvolle Ableitungsformeln finden Sie auf der nächsten Seite.

Feldname	Ableitungsformel und Erläuterung
Rechnungsnummer	`sequence("Rechnung", "Rechnungsnummer")` Die Rechnungsnummer wird automatisch fortlaufend hochgezählt. Die Funktion „sequence“ ermittelt den höchsten Wert des Feldes „Rechnungsnummer“ im Formular „Rechnung“ und addiert die Zahl 1.
Rechnungsdatum	`current.date` Die Systemvariable „current.date“ liefert das aktuelle Systemdatum zurück.
MwStProzent	`19.0` Der Mehrwertsteuer-Prozentsatz wird mit einem Wert von 19,0 vorbelegt. Bitte beachten Sie: In Ableitungen und Scripten wird als Dezimaltrennzeichen stets ein Punkt verwendet.
MwStBetrag	`Nettobetrag * MwStProzent / 100` Der Mehrwertsteuerbetrag wird aus dem Nettobetrag und dem Mehrwertsteuer-Prozentsatz errechnet.
Bruttobetrag	`Nettobetrag + MwStBetrag` Der Bruttobetrag ergibt sich als Summe von Nettobetrag und Mehrwertsteuerbetrag.

Die virtuellen Anschriftsfelder werden mit Hilfe der Verknüpfung zum Formular Kunde ausgefüllt. Die Ableitungsformel lautet allgemein: **Verknüpfungsname.Feldname**.

Anschrift1	`Kunde.Anschrift1`
Anschrift2	`Kunde.Anschrift2`
Strasse	`Kunde.Strasse`
Postleitzahl	`Kunde.Postleitzahl`
Ort	`Kunde.Ort`

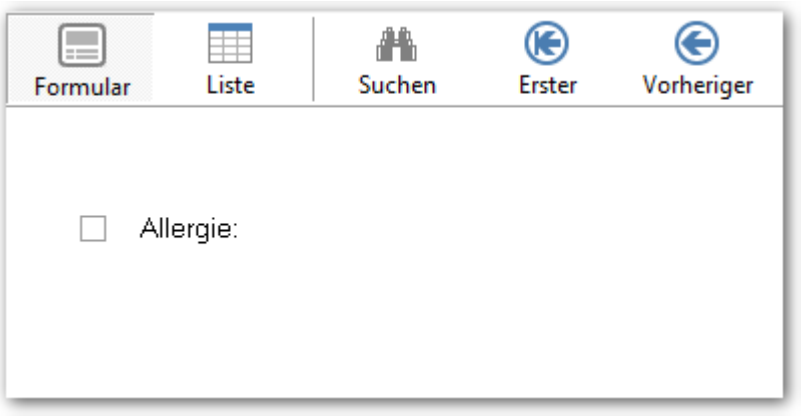
Beim Aufruf einer Rechnung wird die Kundenadresse stets neu aus der Kundenmaske geladen, da die Adressfelder virtuell sind. Auf diese Weise werden nachträgliche Änderungen an der Kundenadresse sofort in den Rechnungen reflektiert.

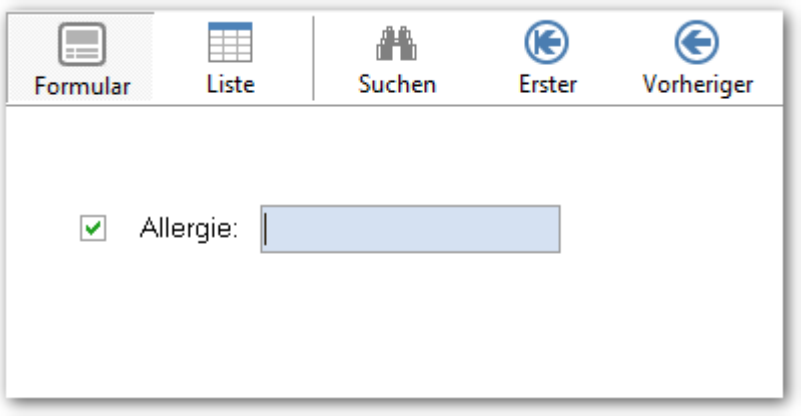
Wenn die Ableitungsformel eines Eingabefeldes den Wert **noexist** ergibt, dann wird das betroffene Feld ausgeblendet. Auf diese Weise können Formulare erstellt werden, die sich in Abhängigkeit der Benutzereingaben dynamisch verändern.

Beispiel:

```
if(Allergie="ja", null, noexist)
```

Die obige Ableitungsformel wird für das Feld Allergiename verwendet und bewirkt, dass die Eingabe eines Allergienamens nur dann abgefragt wird, wenn das Feld "Allergie" angekreuzt ist:





4.6 Ereignisverarbeitung

Jedes Formular besteht neben der Eingabemaske auch aus einem dazugehörigen Script. Dieses Script wird künftig als **Formularscript** bezeichnet. In der Entwurfsphase können Sie mit dem nebenstehenden Button zwischen Formular- und Script umschalten. Bei den meisten Formularen ist das Script leer. Sie können jedoch bei Bedarf ein Script definieren, das auf bestimmte Ereignisse reagiert. Mit Hilfe eines Formularscriptes können Sie beispielsweise komplizierte Plausibilitätsprüfungen oder Ableitungen realisieren. Das Formularscript kommt also immer dann zum Einsatz, wenn Sie mit Ableitungsformeln an Ihre Grenzen stoßen.

Das Ausführen eines Formularscriptes wird durch verschiedene Ereignisse ausgelöst. DataCool unterscheidet **Systemereignisse** und **Benutzerereignisse**. Die Systemereignisse sind fest vordefiniert und werden automatisch ausgelöst. Die Benutzerereignisse werden durch Buttons, Bildschaltflächen oder Menüs ausgelöst. Jedes Ereignis besitzt einen eindeutigen Bezeichner, den Sie mit der Systemvariable **current.event** im Script abfragen können.

current.event	**Systemereignisse**
load	Laden eines Dokumentes
check	Tritt vor dem Ausführen eines Scriptes auf. Auf diese Weise können diverse Prüfungen der Eingabewerte per Script vorgenommen werden. Bei eventuellen Fehlern kann das Script mit der Anweisung „error“ eine Fehlermeldung ausgeben und so den Start des Scriptes unterbinden.
run	Ausführen eines Scriptes
insert	Tritt **vor** dem Speichern eines neuen Datensatzes auf. Die Speicherung des Satzes kann unterbunden werden, wenn im Script die Systemvariable *current.check* auf *false* gesetzt wird.
postinsert	Tritt **nach** dem Speichern eines neuen Datensatzes auf. Die Speicherung des Satzes kann rückgängig gemacht werden, wenn im Script die Systemvariable *current.check* auf *false* gesetzt wird.
update	Tritt **vor** der Änderung eines bestehenden Datensatzes auf. Die Speicherung des Satzes kann unterbunden werden, wenn im Script die Systemvariable *current.check* auf *false* gesetzt wird.

postupdate	Tritt **nach** der Änderung eines bestehenden Datensatzes auf. Die Speicherung des Satzes kann rückgängig gemacht werden, wenn im Script die Systemvariable *current.check* auf *false* gesetzt wird.
delete	Tritt **vor** dem Löschen eines Datensatzes auf. Die Löschung des Satzes kann unterbunden werden, wenn im Script die Systemvariable *current.check* auf *false* gesetzt wird.
click_[Feldname]	Dieses Ereignis wird **vor** dem Aufruf eines SQL-Browsers ausgelöst. Die Ereigniskennung enthält den Namen des Auslöserfeldes.
browse_[Feldname]	Dieses Ereignis wird ausgelöst, wenn ein Wert mit Hilfe eines SQL-Browsers ausgewählt wurde. Das Ereignis tritt nicht ein, wenn der SQL-Browser ohne Auswahl eines Wertes geschlossen wird. Die Ereigniskennung enthält den Namen des Auslöserfeldes.

Den Ereignissen **insert**, **update** und **delete** kommt eine besondere Bedeutung zu. Diese Ereignisse treten vor Ausführung der jeweiligen Operation ein. Wenn das Script die Systemvariable current.check auf den Wert false setzt, dann wird die jeweilige Operation unterbunden. Mit anderen Worten: das Script kann also verhindern, dass der angezeigte Datensatz gespeichert oder gelöscht wird.

Beispiel: Bedingter Eingabezwang mit Hilfe eines Formularscriptes

```
if current.event="insert" or current.event="update" then
   if Form.Allergiker="ja" and Form.Allergie=null then
      error "Wogegen sind Sie allergisch?";
      current.check=false;
      focus "Allergie";
      exit script;
   end
end
```

Erläuterung des Scriptes

```
if current.event="insert" or current.event="update" then
```

Der nachfolgende Code wird nur ausgeführt, wenn ein Datensatz hinzugefügt oder geändert werden soll.

```
if Form.Allergiker="ja" and Form.Allergie=null then
```

Wenn das Formularfeld „Allergiker“ mit „ja“ ausgefüllt ist und das Formularfeld „Allergie“ keine Eingabe enthält, dann soll ein Hinweis erfolgen und der Datensatz soll nicht gespeichert werden.

```
message "Wogegen sind Sie allergisch?";
```

Nachfrage an den Benutzer.

```
current.check=false;
```

Wird die Systemvariable **current.check** auf false gesetzt, so wird der Datensatz nicht gespeichert.

```
focus "Allergie";
```

Der Cursor wird in das Feld „Allergie“ gesetzt.

```
exit script;
```

Der Scriptlauf wird beendet. Der Benutzer kann nun mit der Eingabe fortfahren und die fehlenden Angaben ergänzen.

Error-Befehl

Formularscripte werden häufig eingesetzt um Benutzereingaben auf Plausibilität zu prüfen. Tritt ein Fehler auf, so sind meistens die nachfolgenden Schritte erforderlich:

- Ausgabe einer Fehlermeldung für den Benutzer
- Setzen der Systemvariable current.check auf „false“
- Positionierung des Cursors im fehlerhaften Eingabefeld
- Abbruch des Scriptlaufes

Alle oben genannten Funktionen sind beim Befehl **error** zusammengefasst worden. Das vorangegangene Beispiel lässt sich daher auch verkürzt schreiben:

```
if current.event="insert" or current.event="update" then

   if Form.Allergiker="ja" and Form.Allergie=null then

      error "Wogegen sind Sie allergisch?", "Allergie";

   end

end
```

Zugriff auf Formularfelder

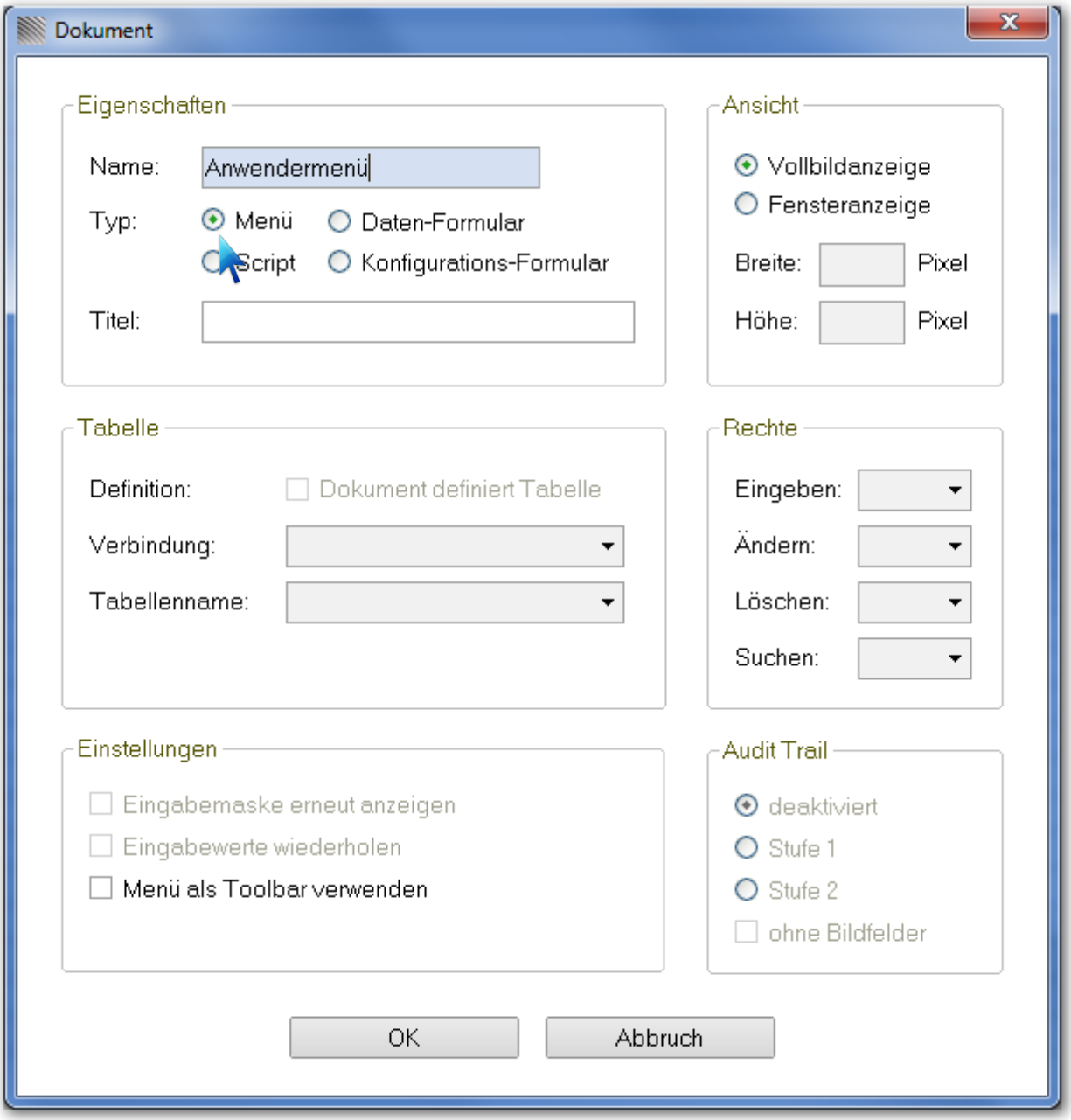
Innerhalb eines Formularscriptes können Sie auf Felder aus der Eingabemaske zugreifen. Das Script kann Feldwerte auslesen oder verändern. Verwenden Sie hierzu den Bezeichner **Form.Feldname**.

5 Menügestaltung

5.1 Standardmenü

Mit Hilfe von Menüs können Sie für den Anwender Ihres Programmes eine Benutzeroberfläche erstellen. Bei DataCool ist das Menü eine Sonderform des Formulars. Ein Menü besitzt im Gegensatz zum Formular keine SQL-Tabelle.

Bitte wählen Sie die Option ***Formulare*** im Hauptmenü von DataCool. Klicken Sie anschließend auf den Button ***Neu*** oder drücken Sie die Taste Einfg um ein neues Dokument zu erstellen. Im nachfolgenden Fenster wählen Sie nun den Dokumententyp „Menü“:



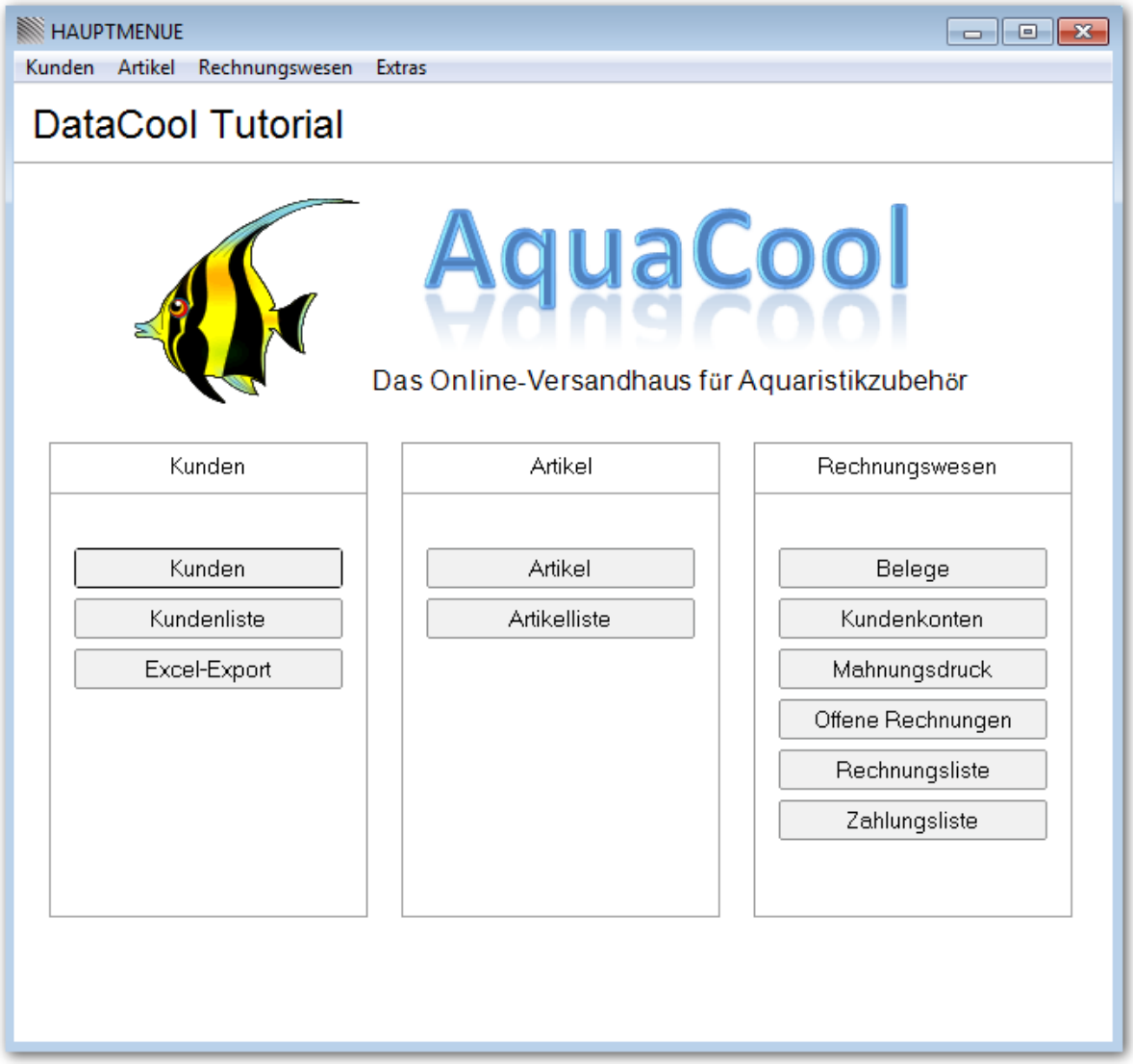
Die Eingabe im Feld **Titel** wird zur Laufzeit des Menüs in großer Schrift angezeigt.

Ein Menü kann aus folgenden Elementen bestehen:

- fester Text
- feste Bilder
- Linien und Rechtecke
- Standard-Schaltflächen (Buttons)
- Bildschaltflächen mit hinterlegten Aktionen
- Virtuelle Felder mit Ableitungsformeln (z.B. „current.user“)
- Subformulare

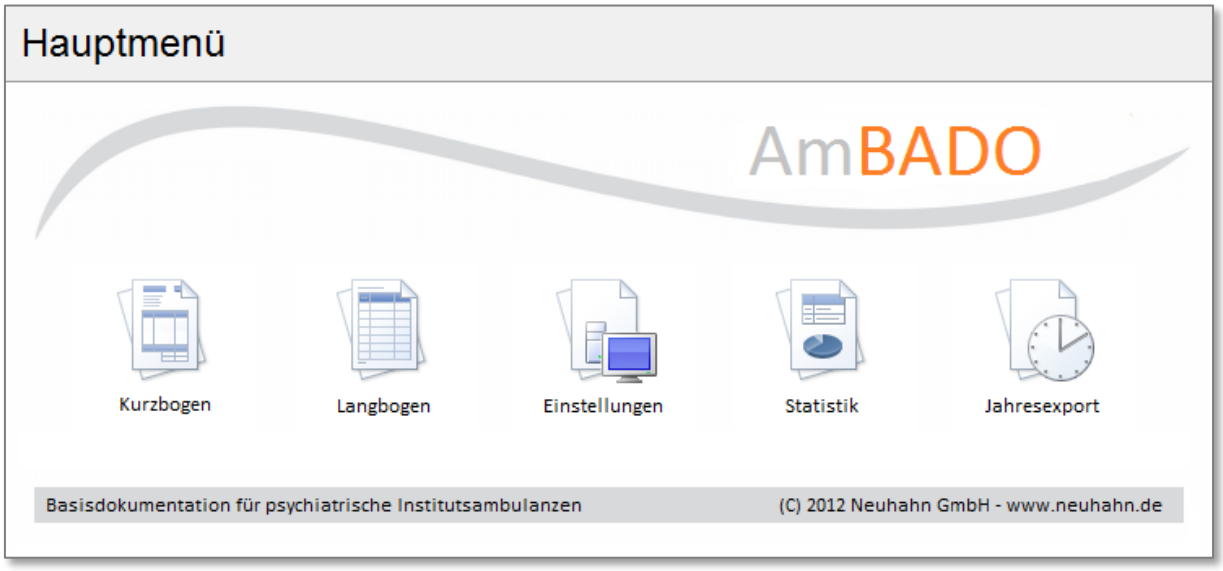
Menüs werden auf die gleiche Art und Weise erstellt wie Formulare. Die Vorgehensweise wird in Kapitel 4.1 ausführlich beschrieben.

Nachfolgend sehen Sie ein Beispiel für ein fertiges Menü mit Standard-Schaltflächen. Dieses Beispiel ist aus dem DataCool Tutorial entnommen:



Alternativ zu den Standard-Schaltflächen können Sie auch Icons für das Menü verwenden. Nachdem Sie die Icons in das Menü eingebunden haben können Sie mit der rechten Maustaste auf die Symbole klicken und dort jeweils die benötigte Aktion hinterlegen.

Beispiel für ein fertiges Menü unter Verwendung von Bildschaltflächen:



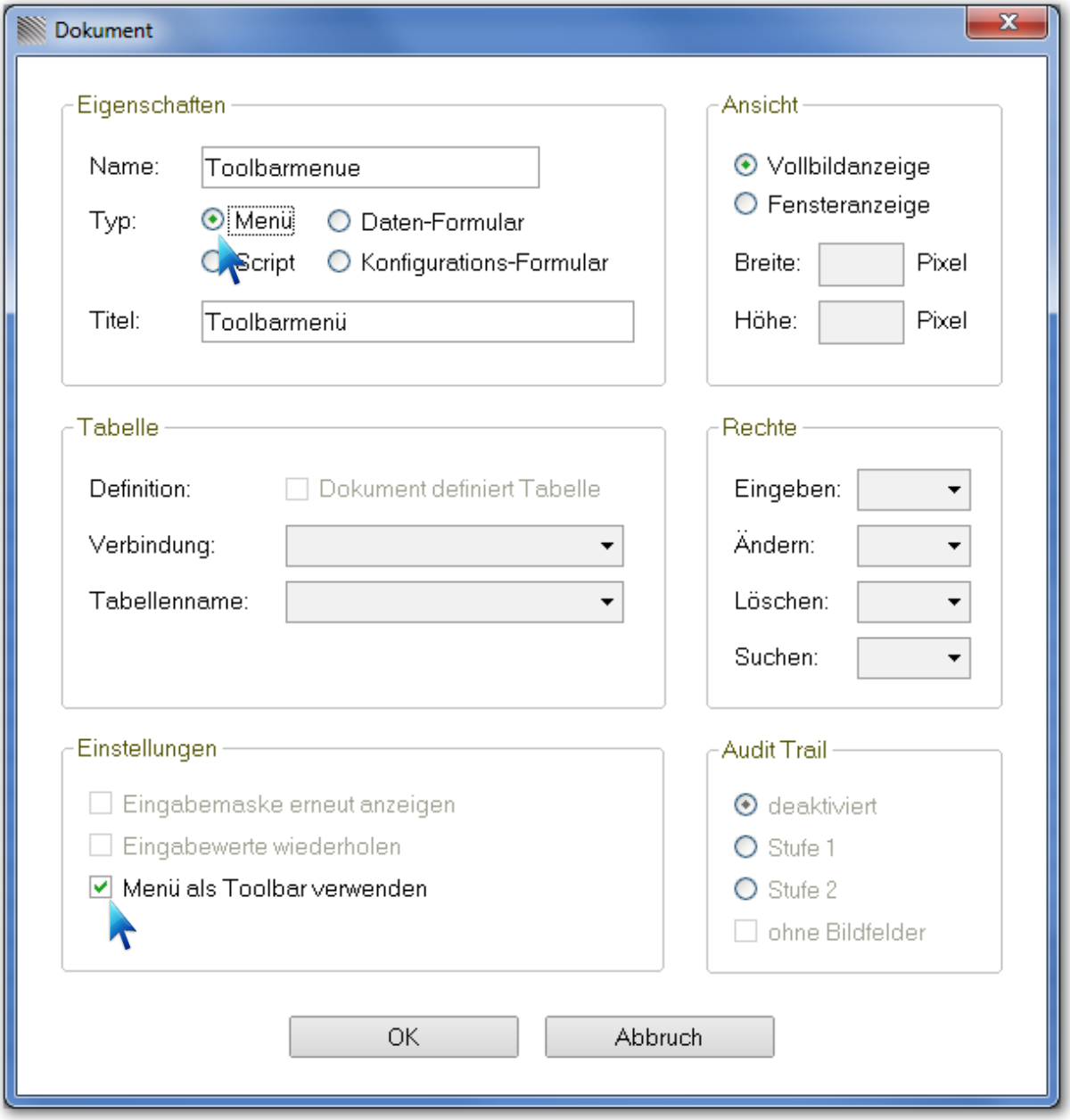
5.2 Toolbarmenü

Das Toolbarmenü ist eine Sonderform des Menüs. Es ist am besten zu vergleichen mit der linken Menüleiste von Microsoft Outlook. Sie können mit Hilfe der Toolbar dynamisch zwischen mehreren Formularen wechseln. Hierbei bleiben die jeweils angezeigten Datensätze auf dem Bildschirm stehen.

Das nachfolgende Beispiel erläutert die Erstellung eines Toolbar-Menüs:

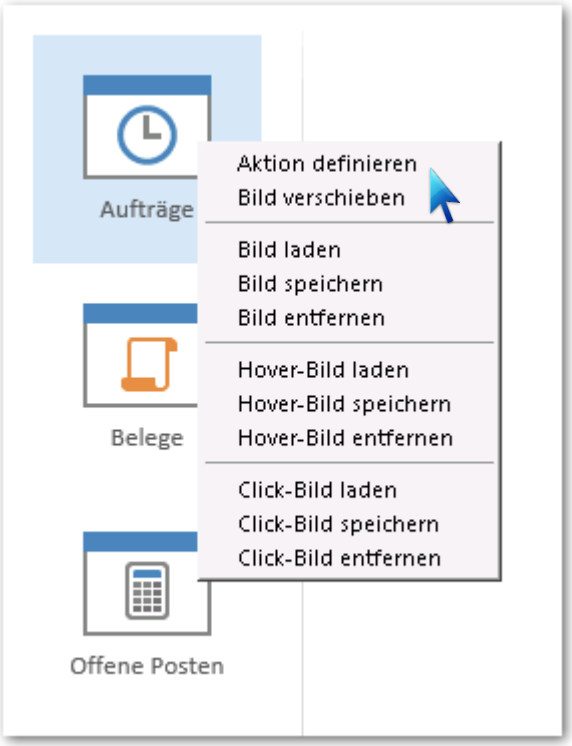
Schritt 1:

Wählen Sie beim Anlegen des Menüs die Option „**Menü als Toolbar verwenden**“:



Schritt 2:

Erstellen Sie ein Menü mit grafischen Buttons. Bitte achten Sie bei der Wahl der Icons darauf, dass diese alle gleich groß sind.

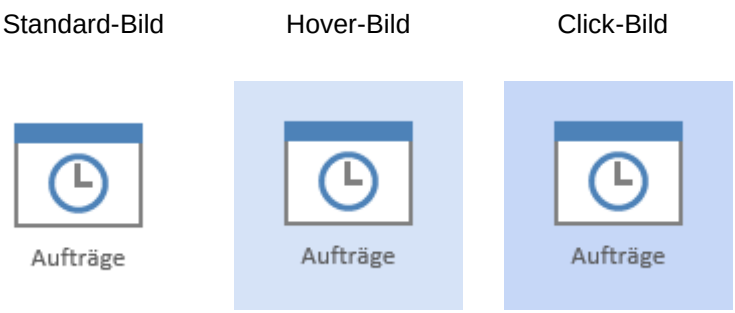


Klicken Sie anschließend auf jedes Icon mit der rechten Maustaste und wählen Sie die Option „Aktion definieren“ aus dem Kontextmenü. Legen Sie auf diese Weise für jedes Icon fest, welches Formular geöffnet werden soll.

Die Aktion des ersten Icons (links oben) wird beim Öffnen des Menüs automatisch aktiviert.

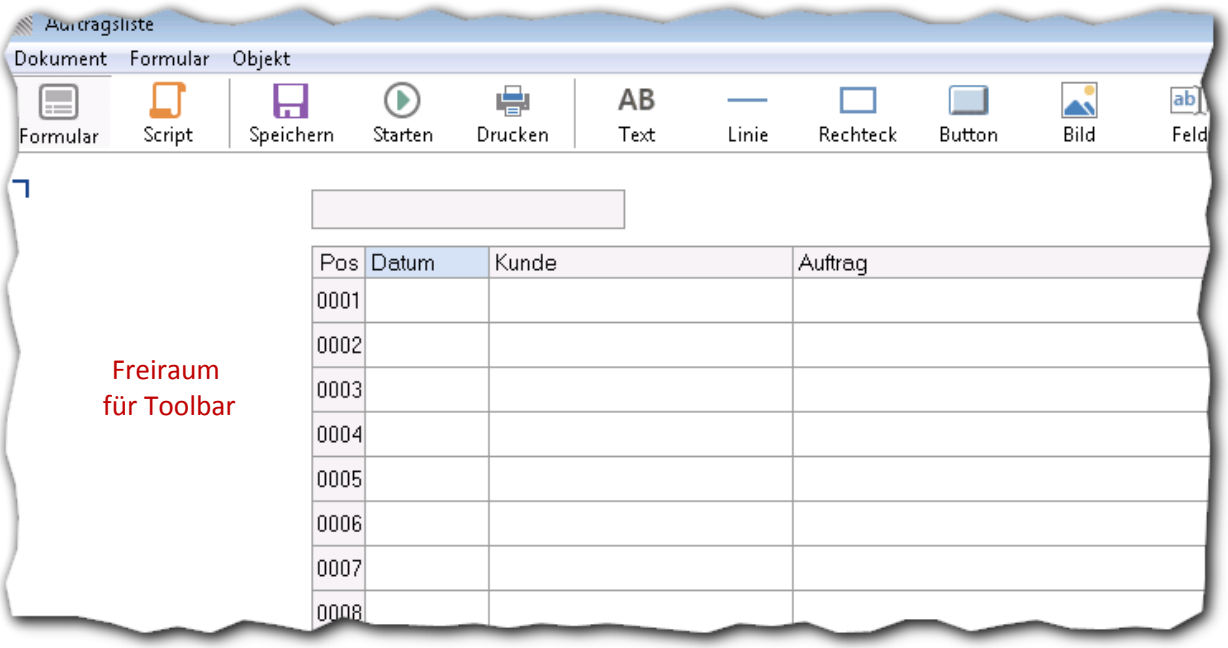
NEU

Ab der Version 2014 können Sie zusätzlich ein **Hover-Bild** und ein **Click-Bild** definieren. Das Hover-Bild erscheint wenn der Mauszeiger über das Bild bewegt wird. Das Klick-Bild erscheint wenn das Bild angeklickt wird. Für ein Toolbar-Menü benötigen Sie also jeweils drei Bilder je Button um einen professionellen Look zu erzielen.



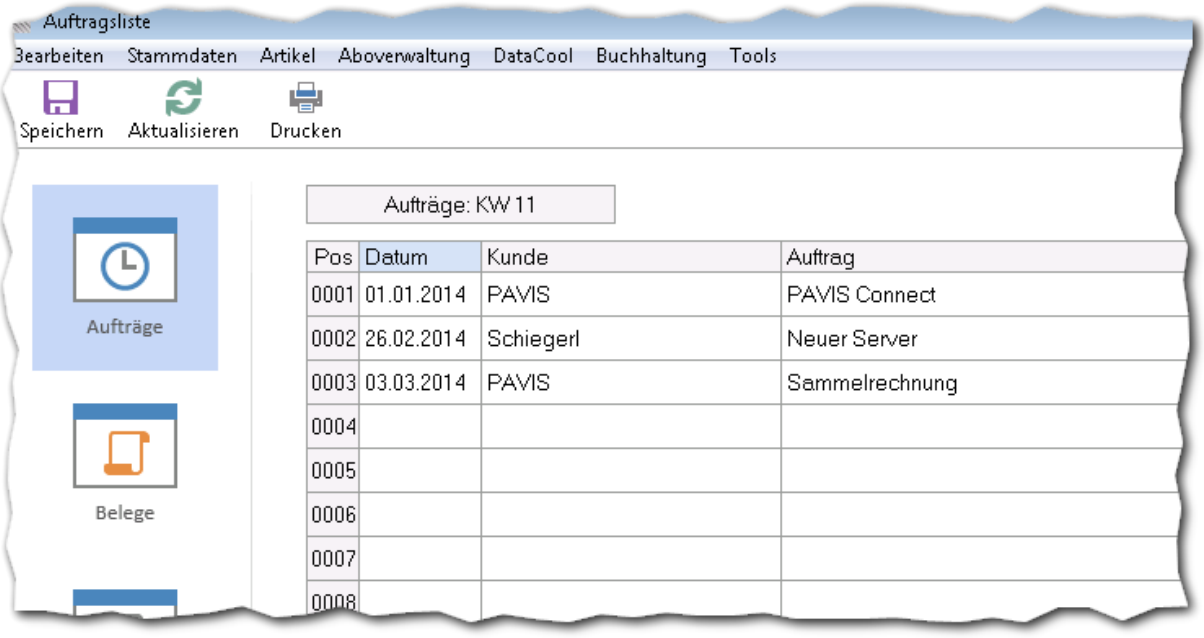
Schritt 3:

Erstellen Sie die benötigten Datenformulare und achten Sie darauf, dass an der geplanten Position der Toolbar-Buttons ein Freiraum bleibt.



Ergebnis:

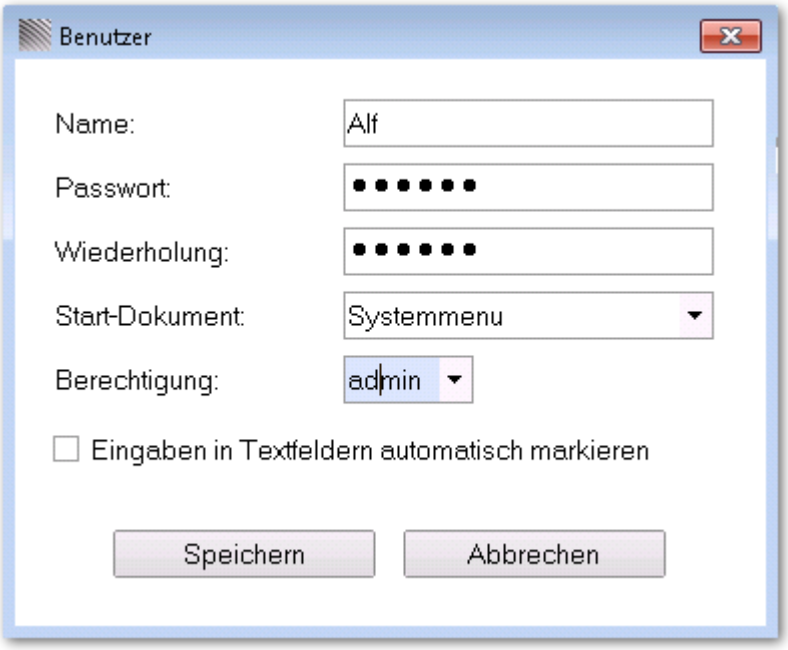
Beim Start des Toolbarmenüs wird die erste Option links oben automatisch aktiviert. Im Falle unseres Beispiels ist dies eine Auftragsliste. Die Toolbar selbst wird dann bei jedem Formular automatisch eingeblendet:



6 Benutzerverwaltung

6.1 Benutzer definieren und ändern

Bitte wählen Sie die Option ***Benutzer*** im Hauptmenü. Klicken Sie anschließend auf den Button ***Neu*** oder drücken Sie die Taste Einfg um ein neues Benutzerkonto zu erstellen. Legen Sie nun die Benutzereigenschaften fest:



Name: Login-Name des Benutzers. Dieser Name kann in Ableitungsformeln oder in Scripten mit der Systemvariable current.user abgefragt werden.

Passwort: Kennwort des Benutzers.

Wiederholung: Bitte geben Sie das Kennwort zur Sicherheit erneut ein.

Start-Dokument: Bestimmen Sie, welches Dokument nach der Anmeldung bei DataCool als erstes gestartet werden soll. Das Startdokument kann ein Menü, ein Formular oder ein Script sein. Der erste Eintrag „Systemmenü“ bedeutet, dass DataCool mit der Programmieroberfläche startet.

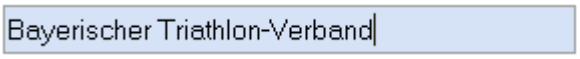
Berechtigung: Wählen Sie eine der Berechtigungsstufen: „niedrig“, „mittel“, „hoch“ oder „admin“. Für die meisten Anwender empfiehlt sich die Stufe „mittel“. Mit dieser Berechtigungsstufe können Sie in den meisten Formularen Daten eingeben, ändern oder löschen. Die höchste Berechtigungsstufe „admin“ ermöglicht das Abspeichern von SQL-Filtern in Formularen.

Eingabemethode: Die Option „**Eingaben in Textfeldern automatisch markieren**“ legt die bevorzugte Eingabemethode für diesen Benutzer fest:

Option deaktiviert (Standard):

Wenn ein Eingabefeld den Fokus bekommt, dann wird der Cursor ans Ende gesetzt. Dies bewirkt, dass neu eingegebene Werte am Ende angehängt werden. Bei einer Neueingabe muss das Feld zunächst mit F6 oder Entf geleert werden.

Beispiel:



Option aktiviert:

Wenn ein Eingabefeld den Fokus bekommt, dann werden vorhandene Daten ggf. automatisch markiert. Dies bewirkt, dass neu eingegebene Werte die vorhandenen Eingaben ersetzen. Bei Änderungen muss jedoch zunächst der Cursor bewegt werden um die Markierung zu entfernen:

Beispiel:

6.2 Benutzerrechte zuweisen

DataCool unterscheidet vier Berechtigungsstufen, die den Zugriff auf diverse Programmfunktionen steuern:

niedrig ⇨ mittel ⇨ hoch ⇨ admin

Bei der Entwicklung Ihrer Anwendung können Sie an verschiedenen Stellen die Berechtigungsstufe festlegen, die ein Benutzer zum Ausführen einer Aktion mindestens besitzen muss:

Formulare:	In den Formulareigenschaften können Sie die Berechtigung für folgende Aktionen festlegen: - Eingeben - Ändern - Löschen - Suchen
Subformulare:	In der Subformulardefinition können Sie die Berechtigung für folgende Aktionen festlegen: - Anzeigen von Subdatensätzen - Einfügen von Subdatensätzen - Löschen von Subdatensätzen - Duplizieren von Subdatensätzen
Buttons:	Berechtigung zum Ausführen der Aktion.
Bild-Schaltflächen:	Berechtigung zum Ausführen der Aktion.
Menü-Optionen:	Berechtigung zum Ausführen der Aktion

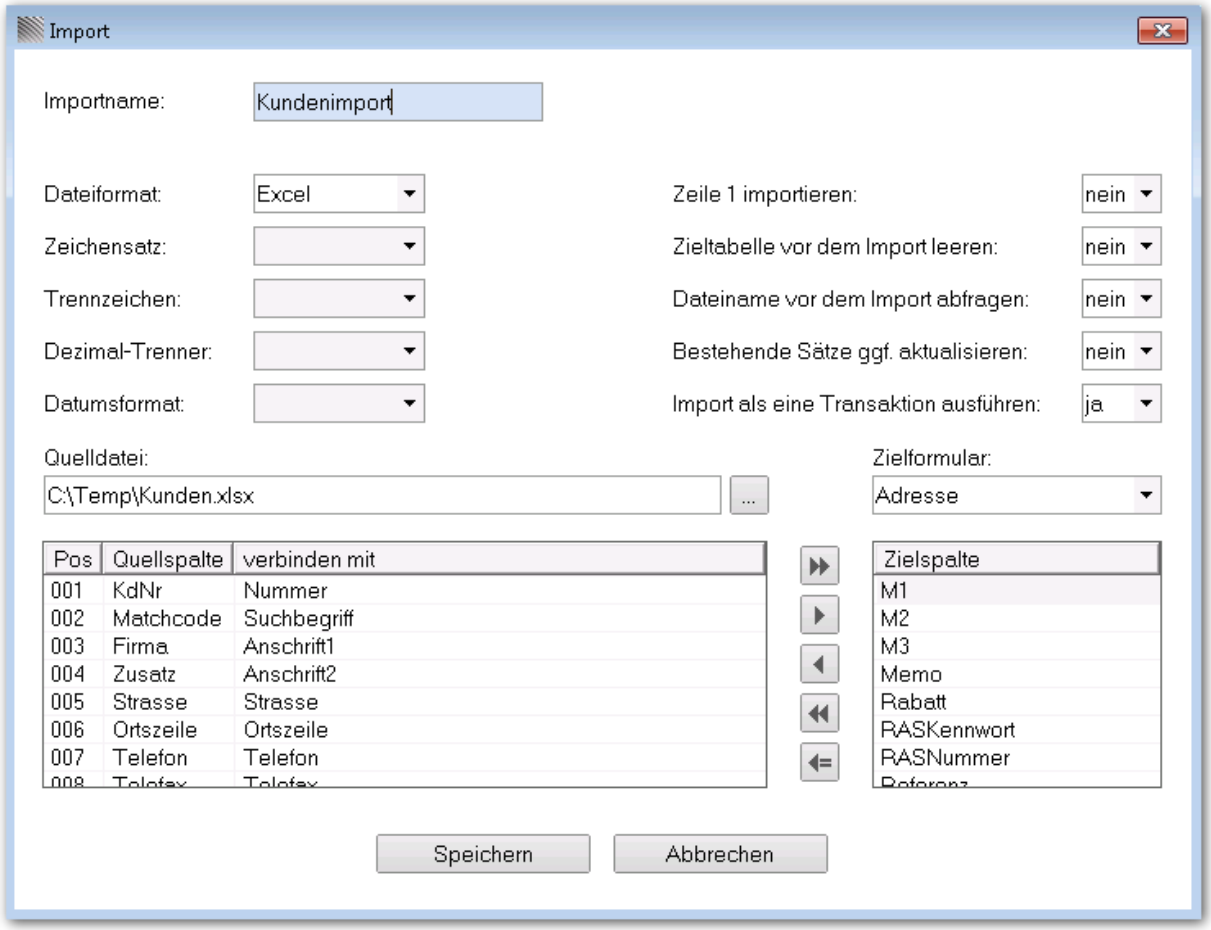
Sie können den Zugang zu bestimmten Dokumenten auch unabhängig von den Benutzerrechten einstellen. Legen Sie hierzu einfach verschiedene Menü-Dokumente für bestimmte Benutzergruppen an. Weisen Sie anschließend jedem Benutzer sein passendes Menü als Startdokument zu.

7 Datenaustausch

7.1 Datenimport

DataCool besitzt eine komfortable Schnittstelle zum Einlesen aller gebräuchlichen Dateiformate. Zum Einlesen von Daten in eine SQL-Tabelle benötigen Sie eine **Importdefinition**. Eine Importdefinition beschreibt den genauen Aufbau der Quelldatei und die Zuordnung zu den einzelnen Datenfeldern der Zieltabelle.

Zur Erstellung einer neuen Importdefinition klicken Sie auf den Button ***Import*** im Hauptmenü von DataCool. Klicken Sie anschließend auf den Button ***Neu*** oder drücken Sie die Taste Einfg. Es öffnet sich der Dialog zur Erstellung einer Importdefinition:



Importname: Vergeben Sie eine eindeutige Bezeichnung für die Importdefinition. Die Importdefinition kann unter Angabe des Namens wie folgt gestartet werden:

- über einen Button
- über eine Menüoption
- über eine Bildschaltfläche
- mit Hilfe des Befehls „import“ der Scriptsprache

Dateiformat: Geben Sie das Dateiformat der Quelldatei an. Folgende Formate werden unterstützt:

Text

Das Format „Text“ ist das am weitesten verbreitete Dateiformat zum Austausch von Daten. Häufig wird auch die Bezeichnung **CSV-Datei** für dieses Format verwendet. Eine Textdatei besteht aus einer beliebigen Anzahl von Zeilen. Jede Zeile in der Quelldatei entspricht genau einem Datensatz. Die einzelnen Felder einer Zeile werden durch ein **Feldtrennzeichen** von einander getrennt. Aufgrund des Feldtrennzeichens ist es kein Problem, wenn die Feldinhalte unterschiedlich lang sind. Man spricht daher auch von einer Textdatei mit variabler Länge.

Excel

Daten aus der Tabellenkalkulation Microsoft Excel lassen sich besonders einfach einlesen. Die Importfunktion setzt allerdings voraus, das Excel 2003 oder höher auf dem Rechner installiert ist.

DataEase 4.x

DataCool verfügt über eine spezielle Schnittstelle, die das Einlesen von Formulardaten aus dem Datenbanksystem DataEase ermöglicht. Es wird ausschließlich die Version 4.x für DOS unterstützt. Für den Import benötigen Sie stets zwei Dateien: die dba-Datei mit der Formulardefinition und die dbm-Datei mit den Daten.

XML

XML ist die Abkürzung von e**X**tensible **M**arkup **L**anguage. Das XML-Format findet besonders im Internet weite Verbreitung. Es handelt sich um eine Sonderform einer Textdatei, bei der die Feldinhalte von einer Anfangs- und Endekennung umschlossen werden. Diese Kennung wird auch als öffnendes bzw. schließendes Tag bezeichnet:

<Feldname> Feldinhalt </Feldname>

Durch die Verschachtelung von Tags ist das Format sehr gut zur Abbildung von hierarchischen Datenstrukturen geeignet. Ein weiterer Vorteil besteht in der Möglichkeit Text- und Binärdaten in einer Datei zu übertragen. DataCool nutzt diese Technik für Formulare mit Bildfeldern.

Das nachfolgende Beispiel zeigt eine einfache XML-Datei mit zwei Datensätzen. Jeder Datensatz wird von den Tags <record> und </record> umschlossen. Innerhalb eines Datensatzes sind die einzelnen Werte zwischen Tags mit den Feldnamen eingebettet.

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<!-- DataCool Export 'Artikel' -->
<export>
   <record>
      <BestNr> q2610a </BestNr>
      <Bezeichnung> Toner </Bezeichnung>
   </record>
   <record>
      <BestNr> 51645a </BestNr>
      <Bezeichnung> Tinte </Bezeichnung>
   </record>
</export>
```

Zeichensatz: Geben Sie den Dateiursprung (DOS oder Windows) an. Diese Einstellung ist nur für Textdateien relevant. Dateien, die mit dem DOS-Zeichensatz erzeugt wurden, werden von DataCool beim Import in den Windows-Zeichensatz konvertiert.

Trennzeichen: Beim Dateiformat Text müssen Sie das Feldtrennzeichen festlegen. Wählen Sie eines der gebräuchlichen Trennzeichen aus der Liste oder geben Sie ein beliebiges Symbol als Trennzeichen ein. Die Auswahl „Tab“ bedeutet, dass die Felder mit einem Tabulator (Zeichencode $09) voneinander getrennt sind.

Dezimal-Trenner: Bitte legen Sie fest, ob die Datei ein Komma oder einen Punkt als Dezimaltrennzeichen verwendet. Diese Einstellung ist nur für die Formate Text und XML relevant.

Datumsformat: Bestimmen Sie die Reihenfolge der Angaben Tag, Monat und Jahr. Legen Sie außerdem fest, ob die Jahreszahl 4-stellig oder 2-stellig angegeben wird. 2-Stellige Jahreszahlen werden von DataCool beim Import automatisch in 4-stellige Jahreszahlen gewandelt. Hierbei wird das Startjahr aus der Systemkonfiguration zugrundegelegt. Das Trennzeichen zwischen Tag, Monat und Jahr muss kein Punkt sein. Jedes beliebige oder kein Trennzeichen ist möglich.

TT.MM.JJ, MM.TT.JJ, JJ.MM.TT
TT.MM.JJJJ, MM.TT.JJJJ, JJJJ.MM.TT

Zeile 1 importieren: Diese Einstellung ist nur für die Formate Text oder Excel relevant. Wenn die erste Zeile der Importdatei keinen Datensatz sondern die Feldbezeichnungen als Überschrift enthält, dann müssen die hier „nein“ eintragen.

Zieltabelle leeren: Wenn Sie hier „ja“ eintragen, dann wird die Zieltabelle vor dem Import mit dem SQL-Befehl „truncate table“ geleert. Es erfolgt eine Sicherheitsabfrage, falls die Tabelle bereits Daten enthält. Wird der Import über ein Script ausgeführt, so unterbleibt die Rückfrage damit das Script ohne Benutzereingriff ablaufen kann.

Dateiname abfragen: Wenn Sie hier „ja“ eintragen, dann wird vor dem Import ein kleiner Dateiexplorer geöffnet. Bei der Einstellung „nein“ wird von einem festen Dateinamen ausgegangen.

Sätze aktualisieren: Diese Option ist nur dann relevant, wenn das Zielformular eindeutige Felder besitzt. Anhand der Eindeutigkeitsregel wird bestimmt, ob der zu importierende Datensatz in der Zieltabelle bereits existiert oder nicht.

Bestehende Sätze aktualisieren = „ja“

Ist der Importdatensatz in der Zieltabelle bereits vorhanden, so wird der Datensatz in der Zieltabelle aktualisiert.

Bestehende Sätze aktualisieren = „nein“

Ist der Importdatensatz in der Zieltabelle bereits vorhanden, so wird der Importdatensatz verworfen.

Transaktion: Wenn Sie den Import als eine Transaktion ausführen, dann wird die Importdatei ganz oder gar nicht eingelesen. Tritt während des Einlesevorganges ein Fehler auf (z.B. ein ungültiger Datumswert), so wird der Import insgesamt verworfen.

Wenn Sie den Import nicht als Transaktion ausführen, dann werden nur die fehlerhaften Datensätze der Importdatei übersprungen. Alle korrekten Datensätze werden trotzdem übernommen.

Vorteile beim Import mit Transaktion:

- schnellerer Importvorgang
- konkrete Meldung beim Auftreten des ersten Fehlers

Quelldatei: Wählen Sie mit dem Dateiexplorer eine Quelldatei aus. Unmittelbar nach der Auswahl wird die erste Zeile der Importdatei in die **Zuordnungstabelle** geladen.

Zielformular: Wählen Sie das Zielformular für die importierten Daten aus. Mit Hilfe der Buttons können Sie nun eine Zuordnung zwischen den Quellspalten der Importdatei und den Zielspalten der SQL-Tabelle vornehmen. Auf diese Weise können Sie bestimmen welche Spalten in welche Felder importiert werden sollen. Die Reihenfolge der Felder in der Importdatei muss also nicht mit der Reihenfolge der Felder in der Tabelle übereinstimmen.

Funktionen der Zuordnungstabelle:

 Alle Verbindungen lösen

 Markierte Verbindung lösen

 Verbindung für markierte Spalte erstellen

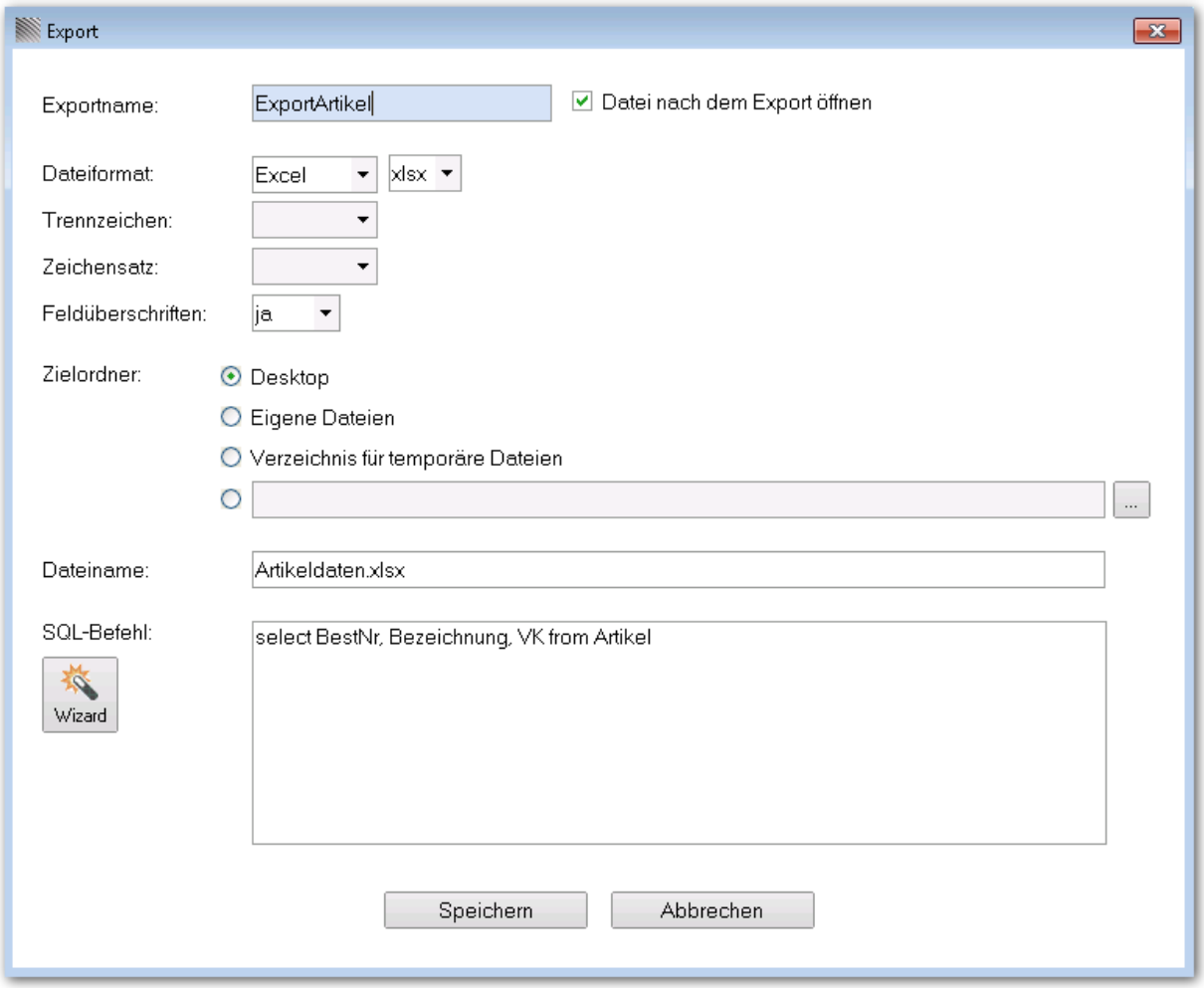
 Alle Spalten verbinden (nach Reihenfolge)

 Alle Spalten verbinden (nach Feldbezeichnung)

7.2 Datenexport

DataCool besitzt eine komfortable Schnittstelle zum Auslesen von Daten in den wichtigsten Formaten. Bevor Sie mit dem Export beginnen können, müssen Sie eine **Exportdefinition** erstellen. Die Exportdefinition beschreibt u.a. die Auswahl und die Reihenfolge der zu auszugebenden Datenfelder.

Zur Erstellung einer neuen Exportdefinition klicken Sie auf den Button ***Export*** im Hauptmenü. Klicken Sie anschließend auf den Button ***Neu*** oder drücken Sie die Taste Einfg. Es öffnet sich der Dialog zur Erstellung einerExportdefinition:



Exportname: Vergeben Sie eine eindeutige Bezeichnung für die Exportdefinition. Die Exportdefinition kann unter Angabe des Namens wie folgt gestartet werden:

- über einen Button
- über eine Menüoption
- über eine Bildschaltfläche
- mit Hilfe des Befehls „export“ der Scriptsprache

Die Option „Datei nach dem Export öffnen“ bewirkt, dass die frisch erstellte Exportdatei automatisch geöffnet wird. Hierzu wird das jeweils geeignete Programm verwendet (Excel, Texteditor oder Webbrowser).

Dateiformat: Wählen Sie ein Format für die Exportdatei. Folgende Formate werden unterstützt:

Text

Das Format „Text“ ist das am weitesten verbreitete Dateiformat zum Austausch von Daten. Häufig wird auch die Bezeichnung **CSV-Datei** für dieses Format verwendet. Eine Textdatei besteht aus einer beliebigen Anzahl von Zeilen. Jede Zeile in der Datei entspricht genau einem Datensatz. Die einzelnen Felder einer Zeile werden durch ein **Feldtrennzeichen** von einander getrennt. Aufgrund des Feldtrennzeichens ist es kein Problem, wenn die Feldinhalte unterschiedlich lang sind. Man spricht daher auch von einer Textdatei mit variabler Länge.

Excel

DataCool kann Ihre gespeicherten Daten direkt als Excel-Tabelle ausgeben. Diese Funktion setzt allerdings voraus, das Excel 2003 oder höher auf dem Rechner installiert ist

Ab DataCool 2014 können Sie bestimmen ob das alte Excel-Format von Office 2003 (.xls) oder das neue Excel-Format ab Office 2007 (.xlsx) verwendet werden soll. Die Verwendung des neuen Formates setzt natürlich voraus dass Excel 2007 oder höher installiert ist.

HTML

HTML steht für **H**yper**T**ext **M**arkup **L**anguage. Dieses Format wird für die Erstellung von Internetseiten verwendet. Mit DataCool können Sie Ihre Daten in Form einer einfachen HTML-Tabelle exportieren. Die erzeugte Exportdatei können Sie per Doppelklick direkt in Ihrem Browser öffnen.

XML

XML ist die Abkürzung von e**X**tensible **M**arkup **L**anguage. Das XML-Format findet besonders im Internet weite Verbreitung. Es handelt sich um eine Sonderform einer Textdatei, bei der die Feldinhalte von einer Anfangs- und Endekennung umschlossen werden. Diese Kennung wird auch als öffnendes bzw. schließendes Tag bezeichnet:

<Feldname> Feldinhalt </Feldname>

Durch die Verschachtelung von Tags ist das Format sehr gut zur Abbildung von hierarchischen Datenstrukturen geeignet. Ein weiterer Vorteil besteht in der Möglichkeit Text- und Binärdaten in einer Datei zu übertragen. DataCool nutzt diese Technik für Formulare mit Bildfeldern.

Das nachfolgende Beispiel zeigt eine einfache XML-Datei mit zwei Datensätzen. Jeder Datensatz ist in das Tag „record“ eingebettet. Innerhalb eines Datensatzes sind die einzelnen Werte zwischen Tags mit dem Feldnamen eingebettet.

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<!-- DataCool Export 'Artikel' -->
<export>
   <record>
      <BestNr> q2610a </BestNr>
      <Bezeichnung> Toner </Bezeichnung>
   </record>
   <record>
      <BestNr> 51645a </BestNr>
      <Bezeichnung> Tinte </Bezeichnung>
   </record>
</export>
```

Trennzeichen: Beim Dateiformat Text müssen Sie das Feldtrennzeichen festlegen. Wählen Sie eines der gebräuchlichen Trennzeichen aus der Liste oder geben Sie ein beliebiges Symbol als Trennzeichen ein. Die Auswahl „Tab“ bedeutet, dass die Felder mit einem Tabulator (Zeichencode $09) voneinander getrennt werden.

Zeichensatz: Geben Sie den gewünschten Zeichensatz für die Exportdatei an (DOS oder Windows). Diese Einstellung ist nur für Textdateien relevant..

Feldüberschriften: Bestimmen Sie, ob die erste Zeile der Exportdatei Feldbezeichner enthalten soll oder nicht.

Zielordner: Geben Sie den Zielordner für die Exportdatei an.

Ab der Version 2014 können Sie auch eines der folgenden Systemverzeichnisse als Ziel für den Export wählen:

- Desktop
- Eigene Dateien
- Verzeichnis für temporäre Dateien

Dateiname: Bestimmen Sie den Dateinamen für die Exportdatei. Bestehende Dateien werden von DataCool beim Export ohne Rückfrage überschrieben.

SQL-Befehl: Geben Sie einen SQL-Befehl zur Selektion der Exportdaten ein. Anhand des select-Befehles können Sie folgende Angaben bestimmen: Tabelle, Feldliste, Sortierreihenfolge.

Beispiel: Der nachfolgende Befehl exportiert die Felder „BestNr“ und „Bezeichnung“ aus der SQL-Tabelle Artikel:

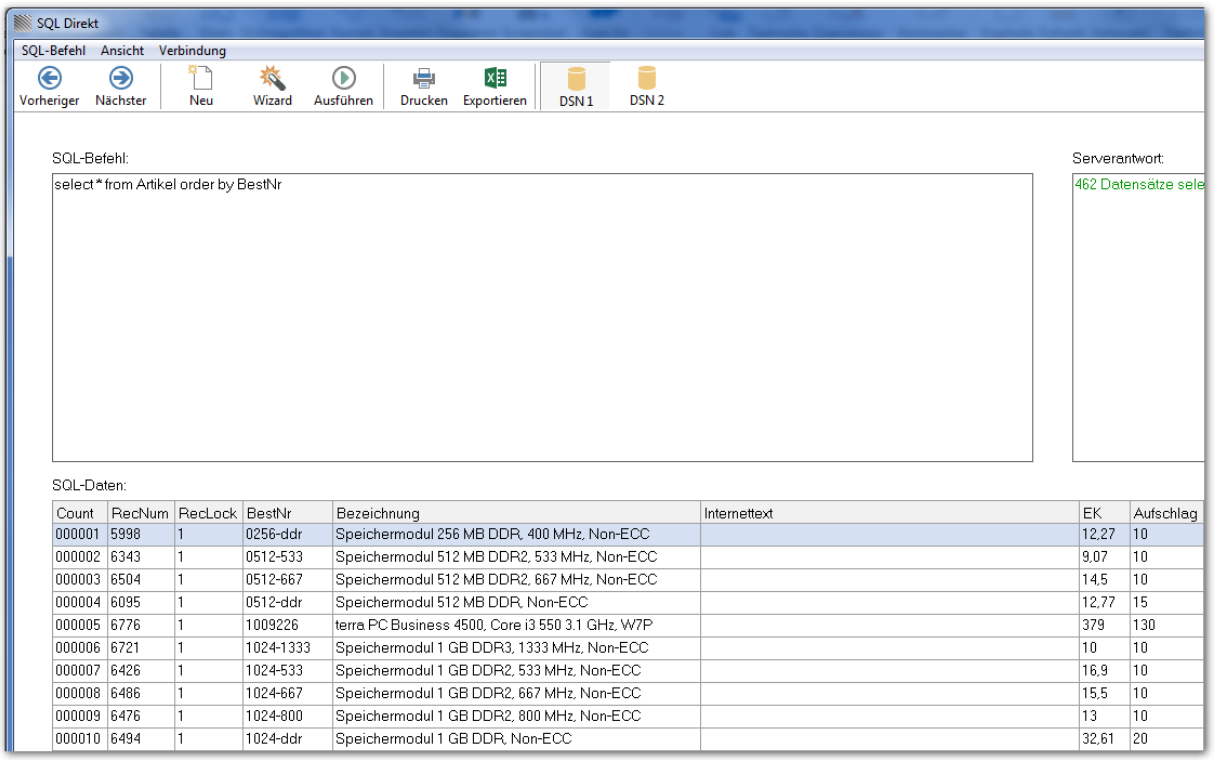
```
select BestNr, Bezeichnung from Artikel
```

8 Hilfsmittel

8.1 SQL Direkt

SQL Direkt ist ein einfaches Dialogfenster zur direkten Kommunikation mit dem Datenbankserver. Sie finden dieses Tool mit Hauptmenü unter **Tools ⇨ SQL Direkt**.

Mit Hilfe von SQL Direkt können Sie beliebige SQL-Befehle direkt an der Server senden. Bei der Verwendung des Befehls select werden die ausgewählten Daten in tabellarischer Form angezeigt:



Count	RecNum	RecLock	BestNr	Bezeichnung	Internettext	EK	Aufschlag
000001	5998	1	0256-ddr	Speichermodul 256 MB DDR, 400 MHz, Non-ECC		12,27	10
000002	6343	1	0512-533	Speichermodul 512 MB DDR2, 533 MHz, Non-ECC		9,07	10
000003	6504	1	0512-667	Speichermodul 512 MB DDR2, 667 MHz, Non-ECC		14,5	10
000004	6095	1	0512-ddr	Speichermodul 512 MB DDR, Non-ECC		12,77	15
000005	6776	1	1009226	terra PC Business 4500, Core i3 550 3.1 GHz, W7P		379	130
000006	6721	1	1024-1333	Speichermodul 1 GB DDR3, 1333 MHz, Non-ECC		10	10
000007	6426	1	1024-533	Speichermodul 1 GB DDR2, 533 MHz, Non-ECC		16,9	10
000008	6486	1	1024-667	Speichermodul 1 GB DDR2, 667 MHz, Non-ECC		15,5	10
000009	6476	1	1024-800	Speichermodul 1 GB DDR2, 800 MHz, Non-ECC		13	10
000010	6494	1	1024-ddr	Speichermodul 1 GB DDR, Non-ECC		32,61	20

Menü „SQL-Befehl“:

NEU

Vorheriger (Shift-F3): DataCool 2020 speichert die letzten 20 SQL-Befehle in einer Historytabelle ab. Mit den Buttons „Vorheriger“ und „Nächster“ können Sie einen SQL-Befehl wiederfinden und ggf. erneut ausführen.

Nächster (F3): DataCool 2020 speichert die letzten 20 SQL-Befehle in einer Historytabelle ab. Mit den Buttons „Vorheriger“ und „Nächster“ können Sie einen SQL-Befehl wiederfinden und ggf. erneut ausführen.

Neu (F5): Löscht die Anzeige für die Eingabe eines neuen Befehls.

Wizard: Ruft den SQL-Wizard auf. Der SQL-Wizard ist ein Hilfefenster, das die Erstellung von select-Befehlen für ungeübte Benutzer erleichtert.

Ausführen (F2): Führt den SQL-Befehl aus. Rückmeldungen werden im Feld „Serverantwort“ angezeigt. Bei der Ausführung von select-Befehlen werden die ausgewählten Daten in die darunterliegende Tabelle geladen.

NEU

Ab der Version 2014 zeigt das Feld „Serverantwort“ auch die Anzahl der selektierten oder geänderten Datensätze an. Darüberhinaus ist es nun möglich einzelne Werte aus der Ergebnistabelle in die Zwischenablage zu kopieren. Klicken Sie hierzu mit der rechten Maustaste auf die gewünschte Tabellenzelle:

SQL-Daten:

Count	RecNum	RecLock	BestNr	Bezeichnung
000001	5967	1	pk-01	Patchkabel, CAT 6, 1.0 Meter
000002	5968	1	pk-02	Patchkabel, CAT 6, 2.0 Meter
000003	5969	1	pk-03	Patchkabel, CAT 6, 3.0 Meter
000004	5970	1	pk-05	.0 Meter
000005	5971	1	pk-10	Patchkabel, CAT 6, 10.0 Meter
000006	5972	1	pk-15	Patchkabel, CAT 6, 15.0 Meter

kopieren

Drucken: Druckt die angezeigten Ergebnisse aus.

Exportieren: Die angezeigte Datentabelle wird an Microsoft Excel übergeben. Diese Funktion setzt voraus, dass Excel auf dem PC installiert ist. Der Excel-Export enthält stets alle selektierten Daten, obwohl die Tabellenanzeige auf dem Bildschirm auf 1.000 oder 5.000 Datensätze beschränkt ist.

Timeout: Für die Ausführung von SQL-Befehlen gilt normalerweise ein Timeout von 90 Sekunden. Wenn DataCool innerhalb dieser Zeit keine Rückantwort vom Server bekommt, dann wird die Ausführung des Befehles abgebrochen.

NEU

In Einzelfällen kann die Ausführung von SQL-Befehlen auch länger dauern, wenn z.B. mit einem update-Befehl eine große Anzahl von Datensätzen verändert wird. In diesem Fall haben Sie nun die Möglichkeit den Timeout-Wert von 90 Sekunden auf 300 Sekunden (5 Minuten) hochzusetzen.

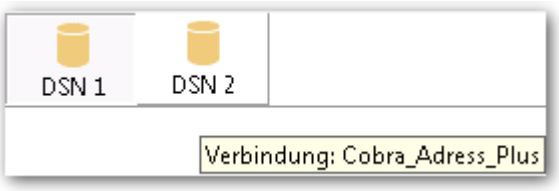
Ansicht:

1000 Sätze: Beschränkt den Ladevorgang auf 1.000 Datensätze.

5000 Sätze: Beschränkt den Ladevorgang auf 5.000 Datensätze.

Verbindungen:

Falls Sie Verbindungen zu anderen SQL-Datenbanken definiert haben (siehe Kapitel 3.8), so können Sie in der Toolbar zwischen diesen Verbindungen umschalten. Die SQL-Befehle werden dann auf die jeweils ausgewähle Datenbank angewendet. Der Name der aktiven Verbindung wird in der Statusbar angezeigt.



8.2 Dokumente exportieren und importieren

Von Zeit zu Zeit kann es erforderlich werden ein DataCool Formular oder ein Script von einer Anwendung in eine andere Anwendung zu überspielen. Zu diesem Zweck stehen Ihnen eine Import- und eine Export-Funktion für Dokumente zur Verfügung.

DataCool verwendet beim Import/Export von Dokumenten das XML-Format. Die Struktur der verwendeten XML-Dateien ist offen und transparent. Auf diese Weise haben Sie eine zukunftssichere Lösung, da Sie auch mit Drittprogrammen ggf. auf die Dokumentendefinition zugreifen können.

Dokument exportieren

Markieren Sie das gewünschte Formular oder Script und wählen Sie die Option **Dokument ⇨ exportieren**. Wählen Sie anschließend den Zielordner für die Exportdatei aus. Der Name des Dokumentes wird automatisch als Dateiname verwendet.

Dokument importieren

Wählen Sie die Option **Dokument ⇨ importieren** aus dem Hauptmenü von DataCool. Wählen Sie anschließend mit dem Explorer die XML-Datei aus. Bitte beachten Sie, dass noch kein gleichnamiges Dokument in der Applikation existieren darf. Beim Import von Formularen wird die zugehörige SQL-Tabelle automatisch mit erstellt.

8.3 Formulare aus anderen Quellen erstellen

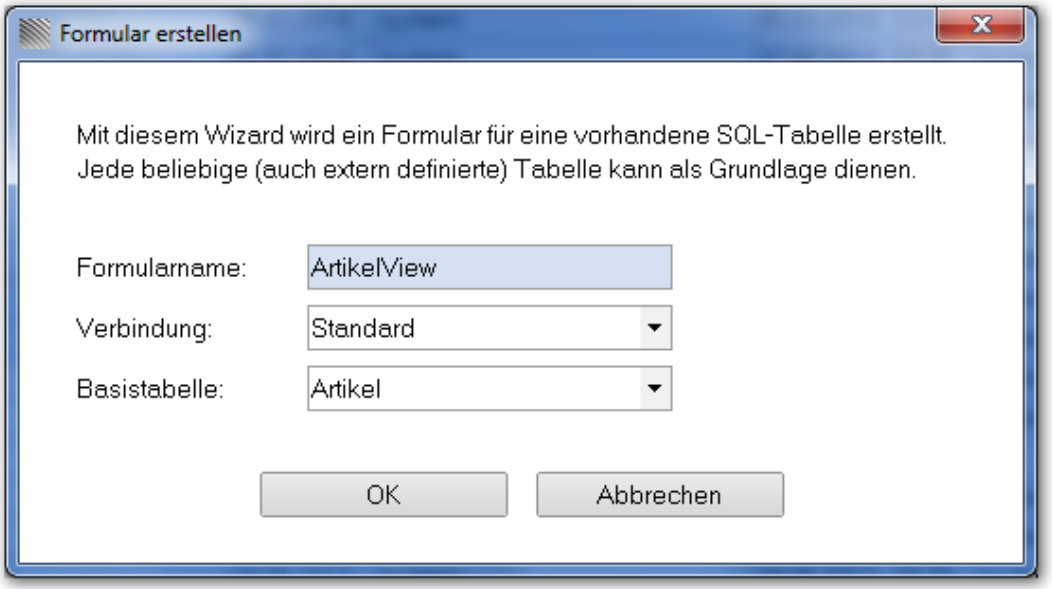
DataCool verfügt über drei Hilfsprogramme zur automatischen Erzeugung von Formularen.

Formular erstellen (aus SQL-Tabelle)

Mit diesem Assistenten erzeugen Sie ein Formular auf der Basis einer bestehenden SQL-Tabelle. Die zugrundeliegende Tabelle kann auch eine Tabelle sein, die mit einer anderen Anwendung erstellt wurde. Das Ergebnis des Assistenten ist ein einfaches Formular mit Feldbezeichnern und Feldern. Die Feldnamen und Feldeigenschaften werden aus den Spalten der SQL-Tabelle abgeleitet.

Das automatisch erzeugte Formular ist ein View. Dies bedeutet, dass Änderungen am Formular nicht zu einer Anpassung der zugehörigen Tabelle führen. Wenn Sie das automatisch generierte Formular löschen, so bleibt die Basistabelle erhalten.

Tools ⇨ Formular erstellen (aus SQL-Tabelle)



Formularname: Name des neuen Formulars (frei wählbar).

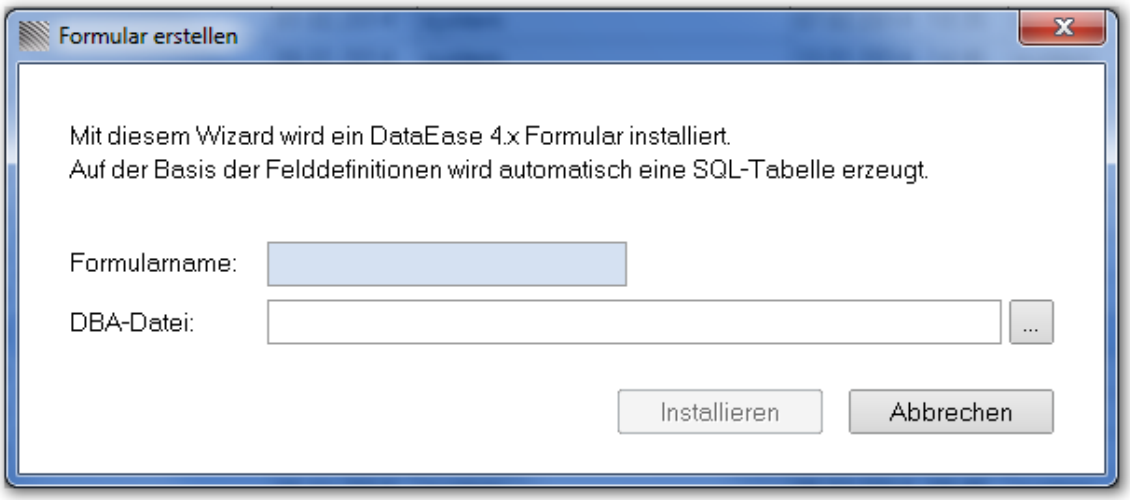
Verbindung: Wählen Sie die Standardverbindung oder die Bezeichnung einer fremden Datenquelle, die Sie zuvor im Menü Konfiguration ⇨ Verbindungen eingerichtet haben.

Basistabelle: Wählen Sie die Tabelle aus, für die das Formular erstellt werden soll.

Formular erstellen (aus DataEase 4.x)

Mit Hilfe des Assistenten können Sie Formulare aus DataEase übernehmen. Sie benötigen hierzu die Formulardefinitionsdatei mit der Endung „dba“. Der Wizard ist nur zu DataEase DOS Version 4.x kompatibel.

Tools ⇨ Formular erstellen (aus DataEase 4.x)



Formularname: Name des neuen Formulars (frei wählbar).

DBA-Datei: Wählen Sie die Formulardefinitionsdatei aus dem Applikationsverzeichnis Ihrer DataEase Anwendung.

DataCool verwendet die Zeilen- und Spaltenposition aus der DOS Maske zur Positionierung der Felder in der Windows Maske. Dies funktioniert in der Regel recht gut. Sie müssen das neue Formular aber eventuell etwas nachbearbeiten.

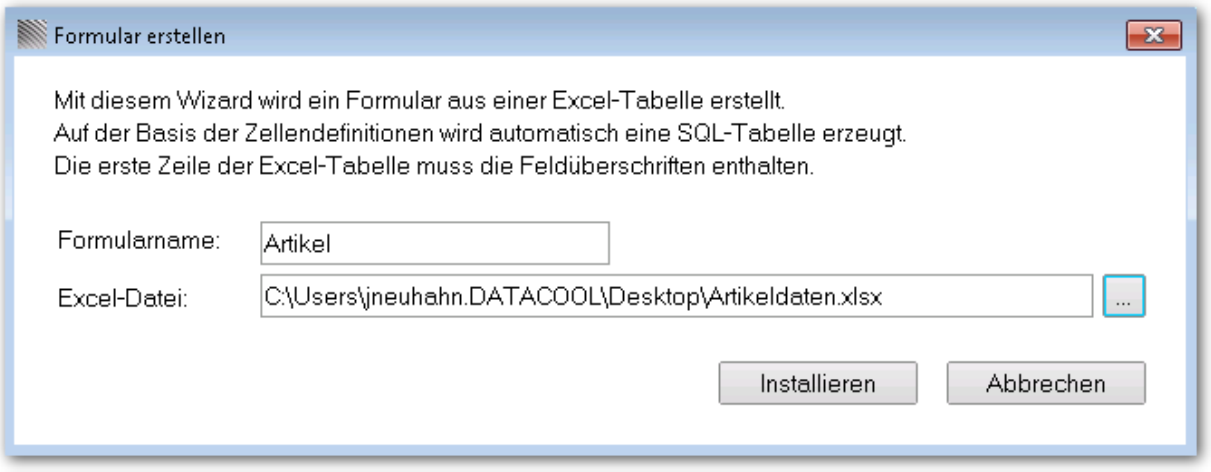
Die Feldeigenschaften (Feldtyp, Feldlänge, etc.) werden bei der Konvertierung exakt übernommen. Bitte beachten Sie jedoch, dass die Ableitungsformeln verloren gehen. Eine Übernahme ist nicht möglich, da DataCool nicht mit der Scriptsprache DQL von DataEase arbeitet.

Nach der Installation des Formulars müssen Sie ggf. noch eine Überspielung der Daten vornehmen. Verfahren Sie hierzu wie im Kapitel 7.1 (Datenimport) beschrieben.

Formular erstellen (aus Excel-Tabelle)

Ab Version 2014 gibt es einen neuen Assistenten zur Erstellung eines Formulares auf der Basis einer Excel-Tabelle. Diese Option ist besonders nützlich wenn Sie eine fremde Excel-Tabelle erhalten und diese für einen Einmalimport einlesen möchten. Bisher mussten Sie manuell ein passendes Hilfsformular erstellen.

Tools ⇨ Formular erstellen (aus Excel-Tabelle)



Bitte beachten Sie, dass die erste Zeile der Excel-Tabelle Spaltenüberschriften enthalten muss. Diese Spaltenüberschriften werden im DataCool Formular als Feldbezeichner verwendet. Die Datentypen der einzelnen Spalten bestimmen die Feldeigenschaften des neuen Formulares.

8.4 Globale Suche

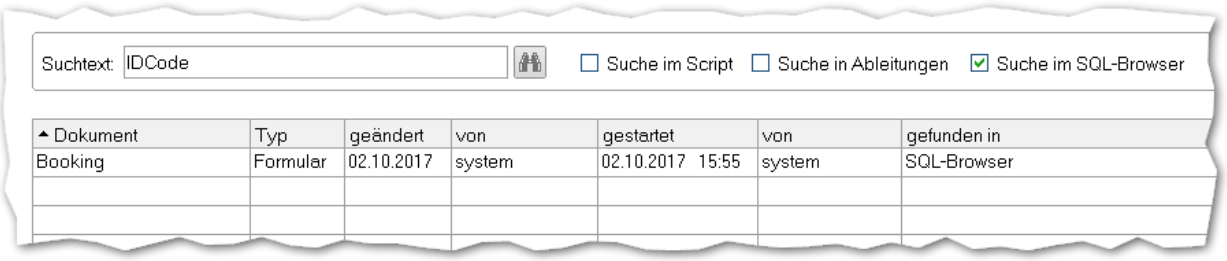
Ab der Version 2014 gibt es ein praktisches Tool für die globale Suche in Scripten, Ableitungsformeln und SQL-Browserdefinitionen. Dieses Tool ist besonders dann nützlich, wenn Sie beispielsweise ein Feld umbenannt haben und den gesamten Programmcode an die neue Bezeichnung anpassen müssen.

So geht's:

Wählen Sie die Option „Globale Suche" in der linken Menüleiste des Hauptmenüs. Geben Sie nun den Suchbegriff ein und legen Sie fest wo gesucht werden soll:

- Suche im Scripttext
- Suche in Ableitungsformeln
- Suche im SQL-Befehl für Auswahlbuttons

Klicken Sie anschließend auf das Symbol mit dem Fernglas um die Suche zu starten: Danach wird Ihnen die Trefferliste angezeigt:



In der rechten Spalte sehen Sie, ob der Suchbegriff im Script, in einer Ableitungsformel oder in einer SQL-Browserdefinition gefunden wurde.

Sie können einen Treffer in der Liste markieren und das Dokument direkt zur Bearbeitung aufrufen. Wenn es sich um ein Script handelt, dann können Sie im Scripttext mit der Taste F3 zu den Fundstellen springen.

9 Netzwerkinstallation

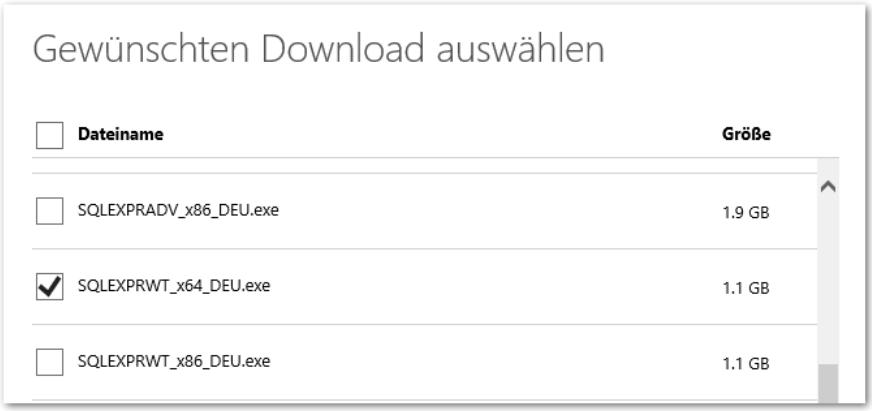
9.1 Installation SQL Server und Management Studio

Beim Einzelplatz-Setup wird von DataCool automatisch die „LocalDB“ installiert. Dies ist eine spezielle Version des „SQL Server Express“, die wenig Ressourcen beansprucht. Diese Version ist jedoch nicht netzwerkfähig. Falls Sie mit DataCool im Mehrplatzbetrieb arbeiten möchten müssen Sie den SQL Server Express installieren. Das nachfolgende Beispiel zeigt die Installation von SQL Server Express 2012. Die Version DataCool 2020 ist bis zur Version 2017 des SQL-Servers freigegeben.

Schritt 1:

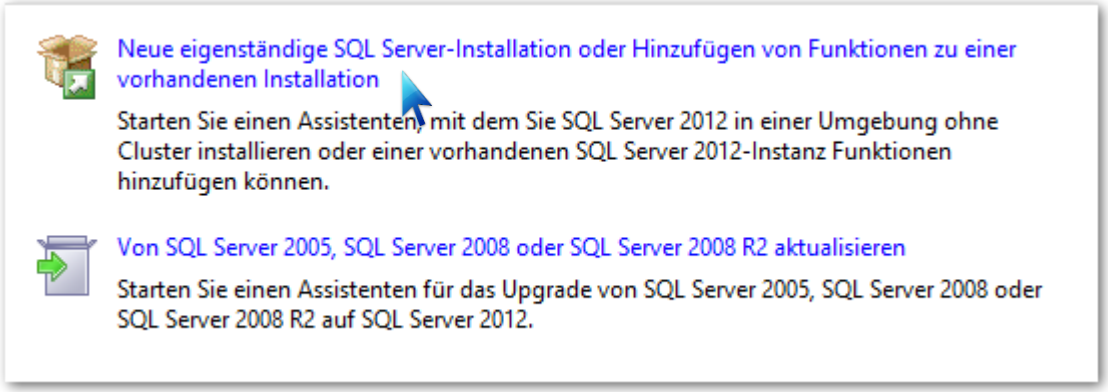
Laden Sie die Setup-Dateien vom Microsoft Download Center herunter:

Wählen Sie hierzu die Version „SQLEXPRWT_**x86**_DEU.exe“ für 32 Bit oder „SQLEXPRWT_**x64**_DEU.exe“ für 64 Bit.



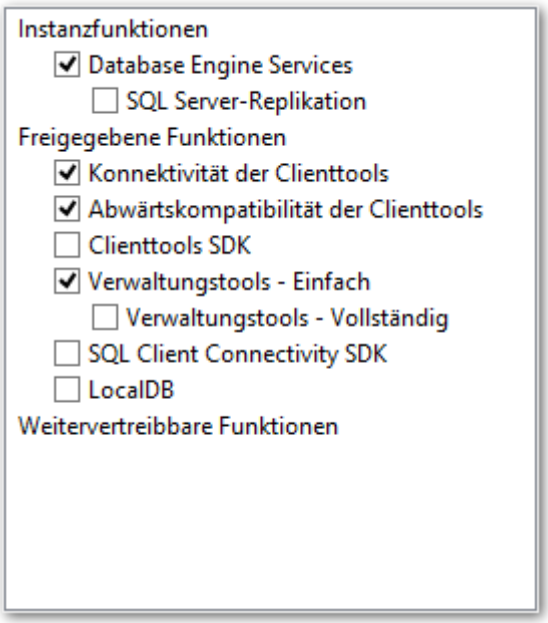
Schritt 2:

Starten Sie die Installation von „SQL Server Express 2012“ und wählen Sie den Punkt „Neue eigenständige SQL Server-Installation“:



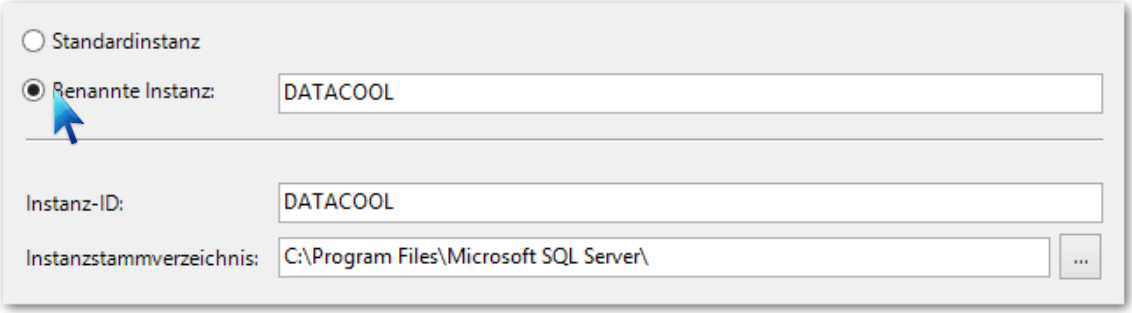
Funktionsauswahl:

Es genügt wenn Sie die nachfolgend angekreuzten Instanzfunktionen auswählen. Die restlichen Komponenten werden nicht benötigt.



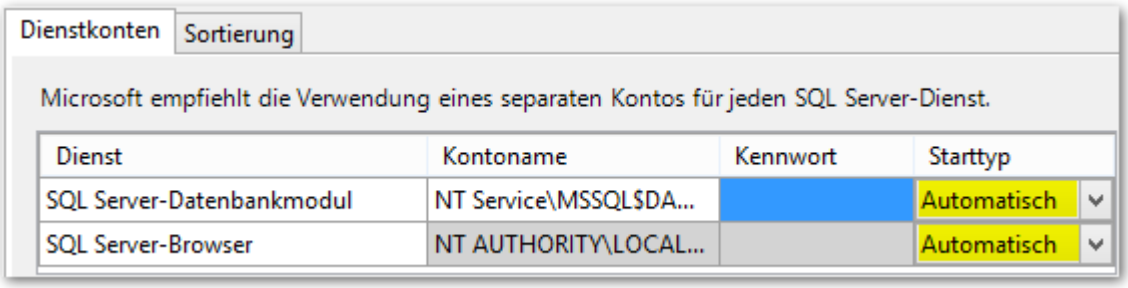
Instanzkonfiguration:

Erstellen Sie eine benannte Instanz mit dem Namen „DATACOOL“.



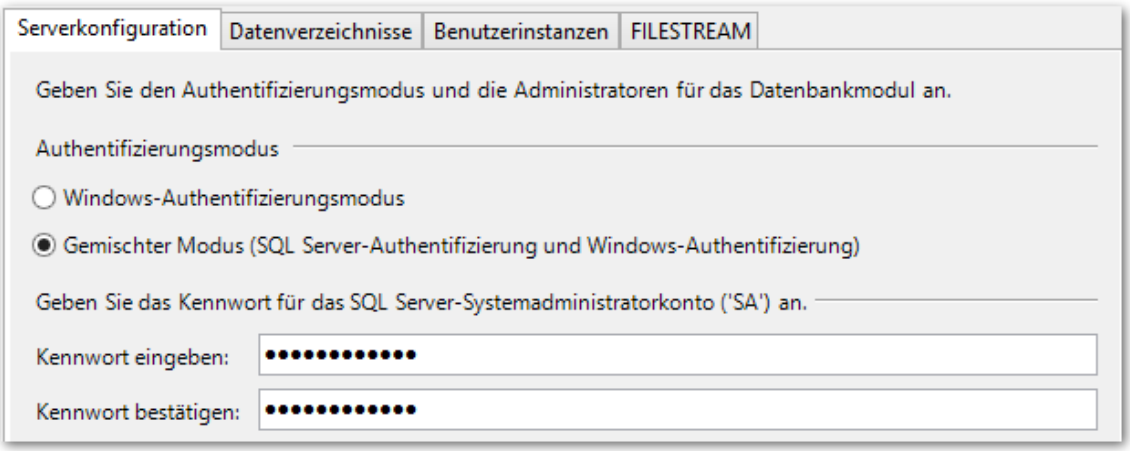
Dienstkonten:

Übernehmen Sie die vorgeschlagenen Werte für den Kontonamen und stellen Sie den Starttyp für beide Dienste auf „**automatisch**“:



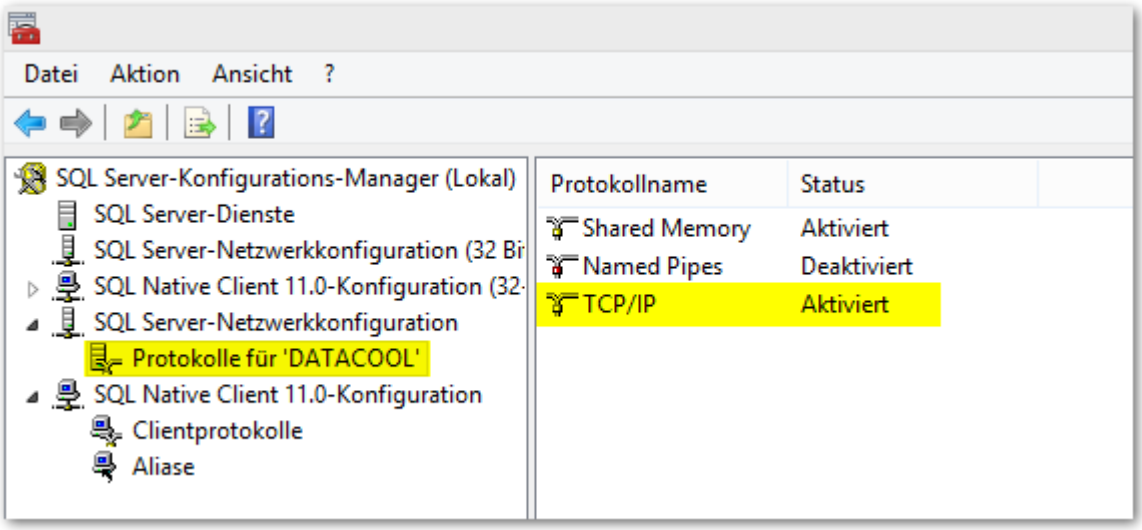
Authentifizierungsmodus:

Wählen Sie „**Gemischter Modus**“ und vergeben Sie ein Kennwort für den SQL Server-Systemadministrator (SA):



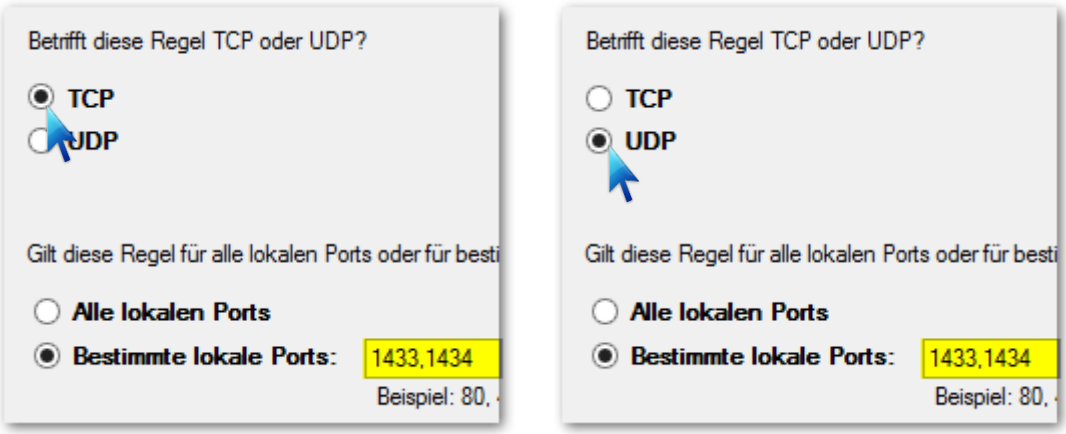
Schritt 3:

Starten den „SQL Server Configuration Manager“ und aktivieren Sie das Netzwerkprotokoll „TCP/IP“ für die benannte Instanz von DataCool:



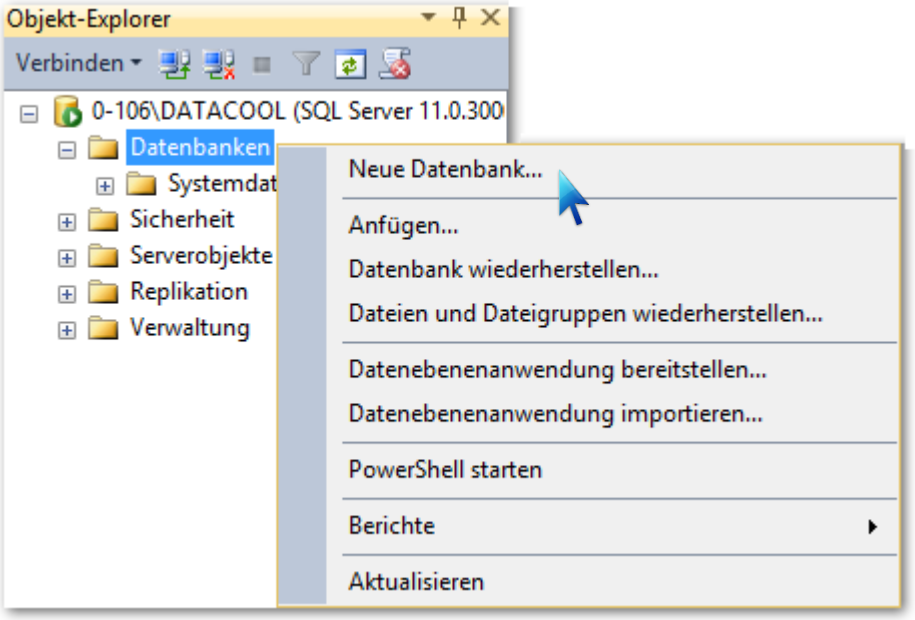
Schritt 4:

Konfigurieren Sie die Windows-Firewall, damit die Client-Computer mit dem SQL Server kommunizieren können. Definieren Sie hierzu zwei eingehende Regeln, die den Datenverkehr der Protokolle TCP und UDP auf den Ports 1433 und 1434 zulassen:

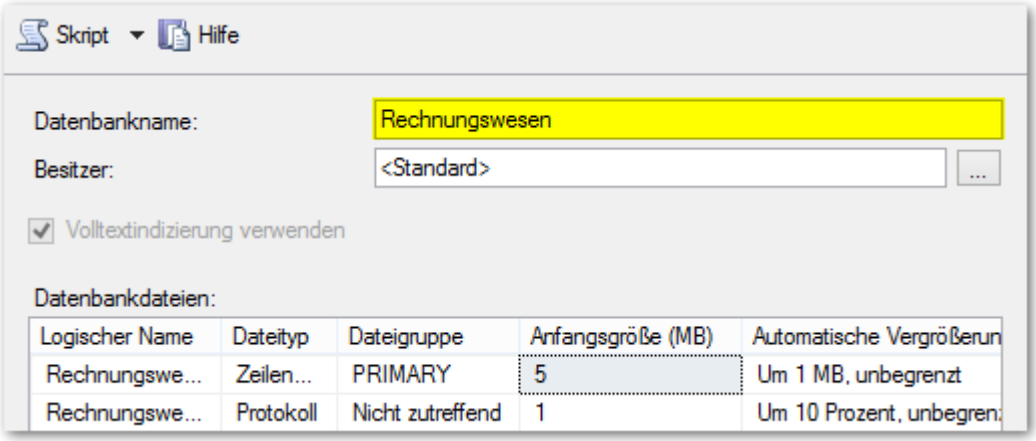


9.2 Erstellen einer neuen Datenbank

Starten Sie das „SQL Server Management Studio“ und klicken Sie mit der rechten Maustaste auf den Punkt „Datenbanken“ um eine neue Datenbank zu erstellen:



Vergeben Sie einen Namen für die neue Datenbank. Der SQL-Server schlägt jeweils einen Dateinamen für die Datenbank und das Transaktionsprotokoll vor. Im Normalfall können Sie die vorgeschlagenen Speicherorte übernehmen.

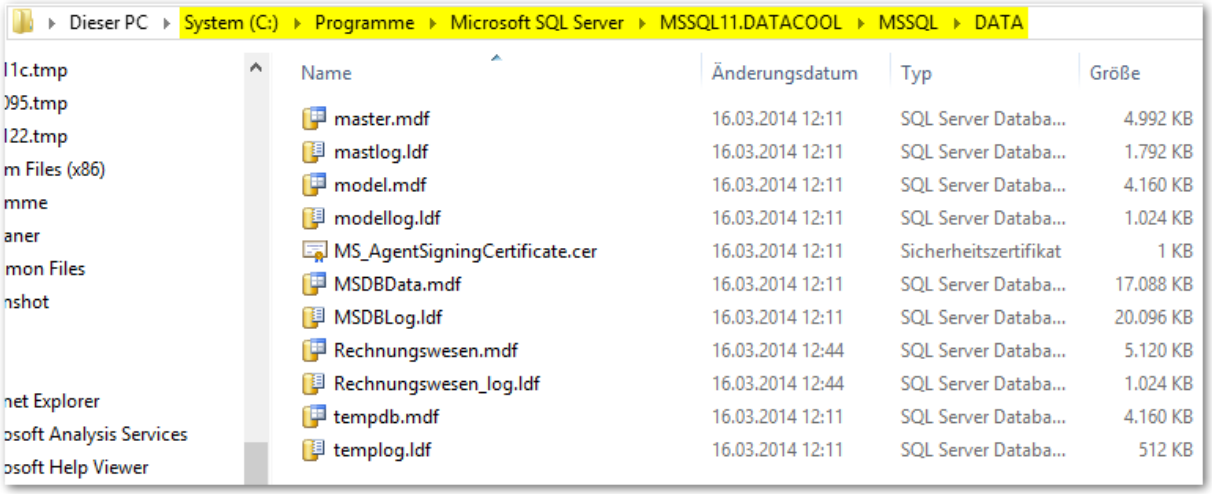


Klicken Sie auf der linken Seite auf den Punkt „Optionen“ und kontrollieren Sie das eingestellte **Wiederherstellungsmodell** der neuen Datenbank. Hier sollte unbedingt der Wert „**Einfach**“ eingestellt werden, da die Transaktionsprotokolle sonst enorm viel Speicherplatz belegen und die Perfomance des Systems beieinträchtigen.

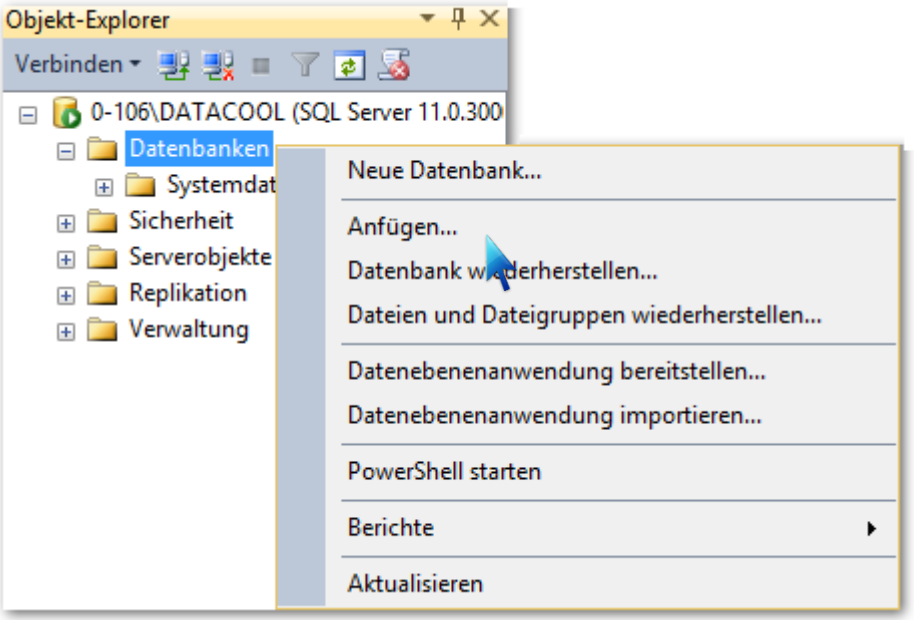
Wenn alle Einstellungen geprüft sind klicken Sie auf OK um die neue Datenbank anzulegen.

9.3 Anhängen einer bestehenden Datenbank

Kopieren Sie zunächst die vorhandene Datenbank (.mdf) und das zugehörige Transaktionsprotokoll (.ldf) in das Datenverzeichnis des SQL Servers:



Starten Sie danach das „SQL Server Management Studio“ und klicken Sie mit der rechten Maustaste auf den Punkt „Datenbanken“ um die vorhandene Datenbank anzufügen:

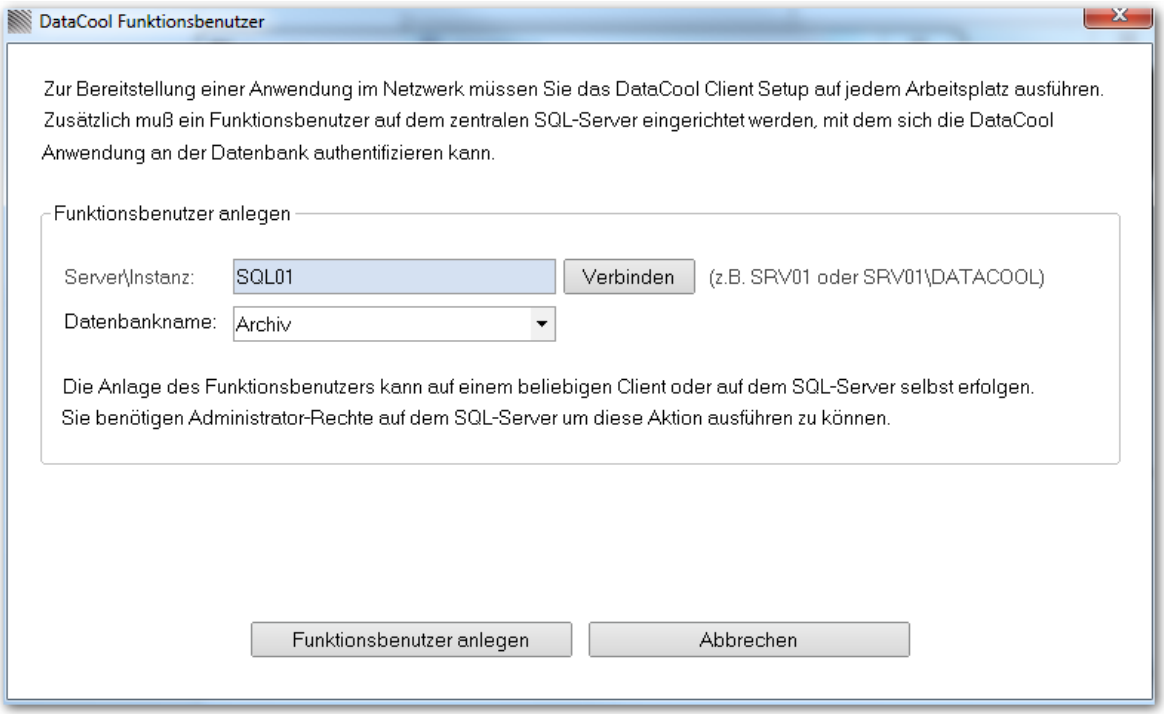


Im nachfolgenden Dialog klicken Sie dann auf Hinzufügen um die Datenbankdateien auszuwählen.

9.4 Funktionsbenutzer und Programmverknüpfung

Funktionsbenutzer

Im Netzbetrieb wird ein sog. Funktionsbenutzer für DataCool auf dem SQL-Server benötigt. Mit Hilfe dieses Benutzers authentifiziert sich die DataCool Anwendung an der SQL-Datenbank. Die Anlage des Funktionsbenutzers kann auf einem beliebigen Client oder auf dem SQL-Server selbst erfolgen. Es ist jedoch erforderlich, dass der angemeldete Windows-Benutzer SYSADMIN-Rechte auf dem SQL-Server besitzt. Es empfiehlt sich daher die nachfolgenden Schritte durchzuführen, während Sie als Domänen-Administrator auf einem Client angemeldet sind.



Arbeitsschritte:

- Starten Sie das Programm DataCool.exe und klicken Sie auf den Link „Funktionsbenutzer“.

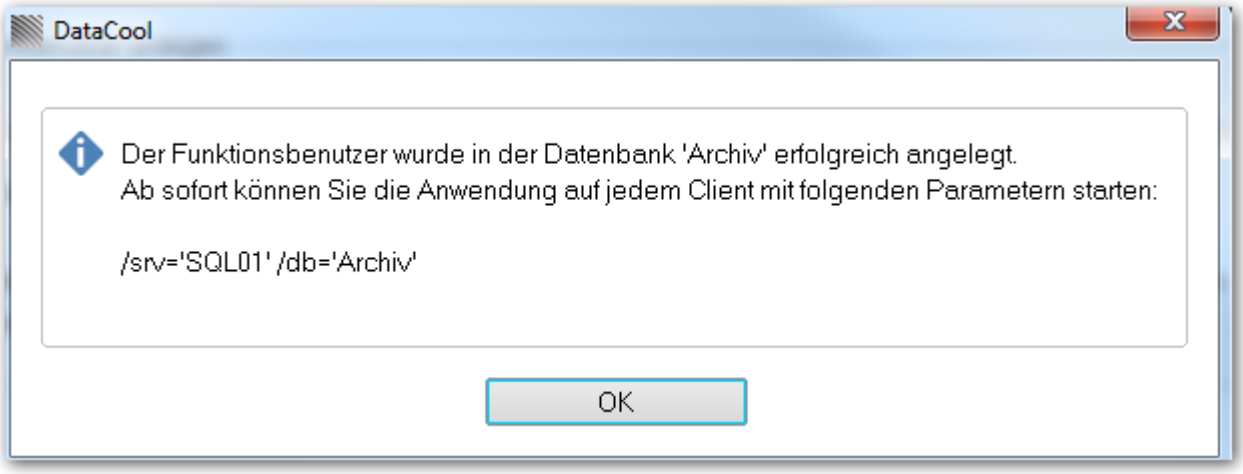
- Geben Sie den Namen des Servers an und hängen Sie ggf. einen Instanznamen an, falls Sie eine benannte Instanz des SQL-Servers installiert haben (z.B: SQL01 für einen Server ohne Instanznamen oder SQL01\DATACOOL für eine benannte Instanz auf dem Server SQL01).

- Klicken Sie auf den Button „Verbinden“. Es wird eine Auflistung aller Datenbanken auf dem Server geladen, sofern der angemeldete Domänenbenutzer über ausreichende Rechte auf dem SQL-Server verfügt.

- Wählen Sie die gewünschte Datenbank aus, die Sie im Netzbetrieb mit DataCool nutzen möchten

- Klicken Sie auf „Funktionsbenutzer anlegen“.

Die Anlage des Funktionsbenutzers wird mit folgender Meldung bestätigt:



Programmverknüpfung:

Für den Zugriff auf die DataCool-Datenbank im Netzbetrieb müssen Sie nun nur noch auf den einzelnen Clients das Client-Setup ausführen und beim Start von DataCool die angezeigten Parameter übergeben. Der Parameter **/srv** teilt dem DataCool-Programm mit auf welchem Server die Datanbank liegt. Der Parameter **/db** ist optional wenn Sie auch die Datenbank vorgeben möchten. Wenn dieser Parameter fehlt, dann erhält der Anwender einen Dialog zur Auswahl aller verfügbaren Datenbanken. Weitere Einzelheiten zu den Befehlszeilenparametern erfahren Sie in Kapitel 3.4.

9.5 Hinweise zur Datensicherung

DataCool speichert sowohl die Programmstruktur als auch die Anwendungsdaten in der SQL-Datenbank. Die SQL-Datenbank besteht aus zwei Dateien:

- der Datenbankdatei mit der Endung MDF
- dem Transaktionsprotokoll mit der Endung LDF

In diesen beiden Dateien sind alle Informationen einer DataCool-Anwendung enthalten:

- Formulare inklusive Bilder
- Scripte
- Verknüpfungen
- Benutzer
- Importdefinitionen
- Exportdefinitionen
- Anwendungstabellen
- Anwendungsdaten inklusive Bildfelder

Offline-Sicherung

Bei einem einfachen Sicherungsmodell genügt es die beiden o.g. Dateien auf ein anderes Medium zu kopieren. Zu diesem Zweck muss die Datenbank gestoppt werden, da die Dateien andernfalls geöffnet sind und sich nicht kopieren lassen.

Sie können die zu sichernde Datenbank entweder im SQL Server Management Studio offline schalten oder mit Hilfe von Befehlen in der Eingabeaufforderung stoppen:

```
net stop mssql$datacool
```

Danach können Sie die Dateien kopieren und die Datenbankdienste wieder starten:

```
net start mssql$datacool
```

Die Datenbankdateien befinden sich in der Regel im folgenden Verzeichnis auf der Festplatte des Servers:

C:\Programme\Microsoft SQL Server\MSSQL11.DATACOOL\MSSQL\Data

Hinweis: Beim SQL Server Express Local DB befinden sich die Datenbankdateien stattdessen im Profilverzeichnis des Benutzers.

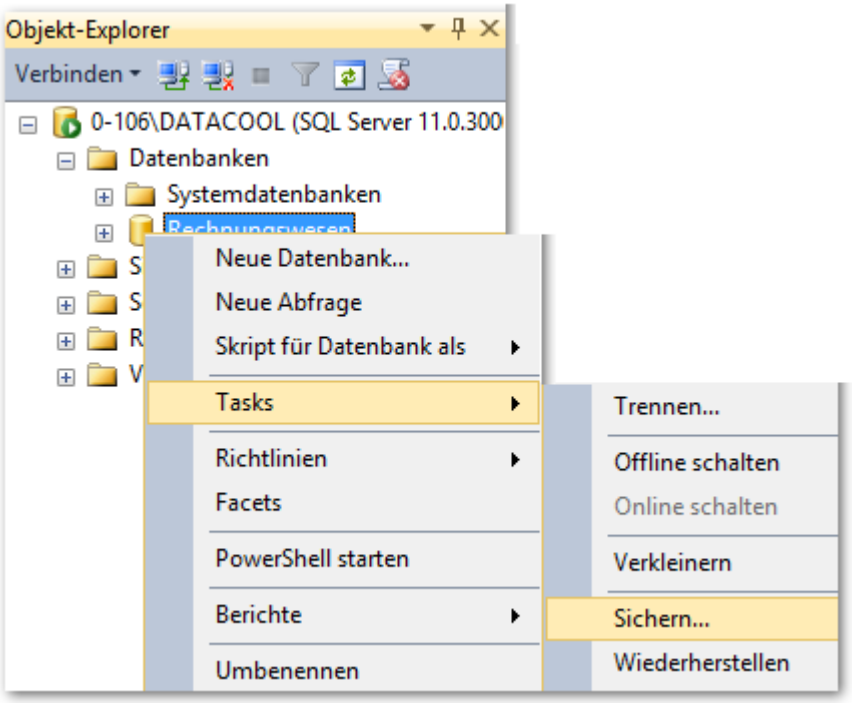
Online-Sicherung

Für eine Online-Sicherung Ihrer Datenbanken gibt es prinzipiell zwei Möglichkeiten:

- Erstellung einer VSS-Kopiesicherung:

  Verwenden Sie eine Sicherungssoftware, die den Volume Shadow Service (VSS) unterstützt. Hierbei wird eine laufende Datenbank während der Sicherung kurz gestoppt um einen konsistenten Zustand der Sicherung zu ermöglichen. Nur so ist gewährleistet, dass die Sicherungsdateien später erfolgreich wiederhergestellt werden können. Sie können VSS-Sicherungen beispielsweise mit dem Windows Server Backup oder dem Sicherungsprogramm Acronis Backup & Recovery erstellen.

- Online-Sicherung mit dem Management Studio



Bei der Onlinesicherung wird eine Sicherungsdatei mit der Endung „.bak“ erstellt. Diese Datei kann mit der Option „Wiederherstellung“ im Management Studio für ein Restore verwendet werden.

In der kostenpflichtigen Vollversion des SQL-Servers gibt es das Tool „SQL Server Agent“. Mit diesem Tool können Onlinesicherungen zeitgesteuert automatisch durchgeführt werden. In der kostenfreien Version „SQL Server Express“ ist diese Möglichkeit nicht vorgesehen.

10 DataCool Scriptsprache (SQL Basic)

10.1 Aufbau der Scriptsprache

Die DataCool Scriptsprache verbindet die mengenorientierte Abfragesprache **SQL** mit der einfachen Programmiersprache **BASIC**.

BASIC steht für „**B**eginner's **A**ll-purpose **S**ymbolic **I**nstruction **C**ode“ und wurde in seiner Urform bereits 1963 entwickelt.

Ausgehend von dem einfachen BASIC Befehlssatz wurde unsere Scriptsprache um viele nützliche Zusatzfunktionen wie etwa der Druck von Barcodes oder das Versenden von E-Mails erweitert.

Im Kapitel 10.4 können Sie sich einen Überblick über die Leistungsfähigkeit der DataCool Scriptsprache verschaffen.

Im Kapitel 10.5 finden Sie den Referenzteil, der alle Befehle ausführlich erläutert.

10.2 Der Script-Editor

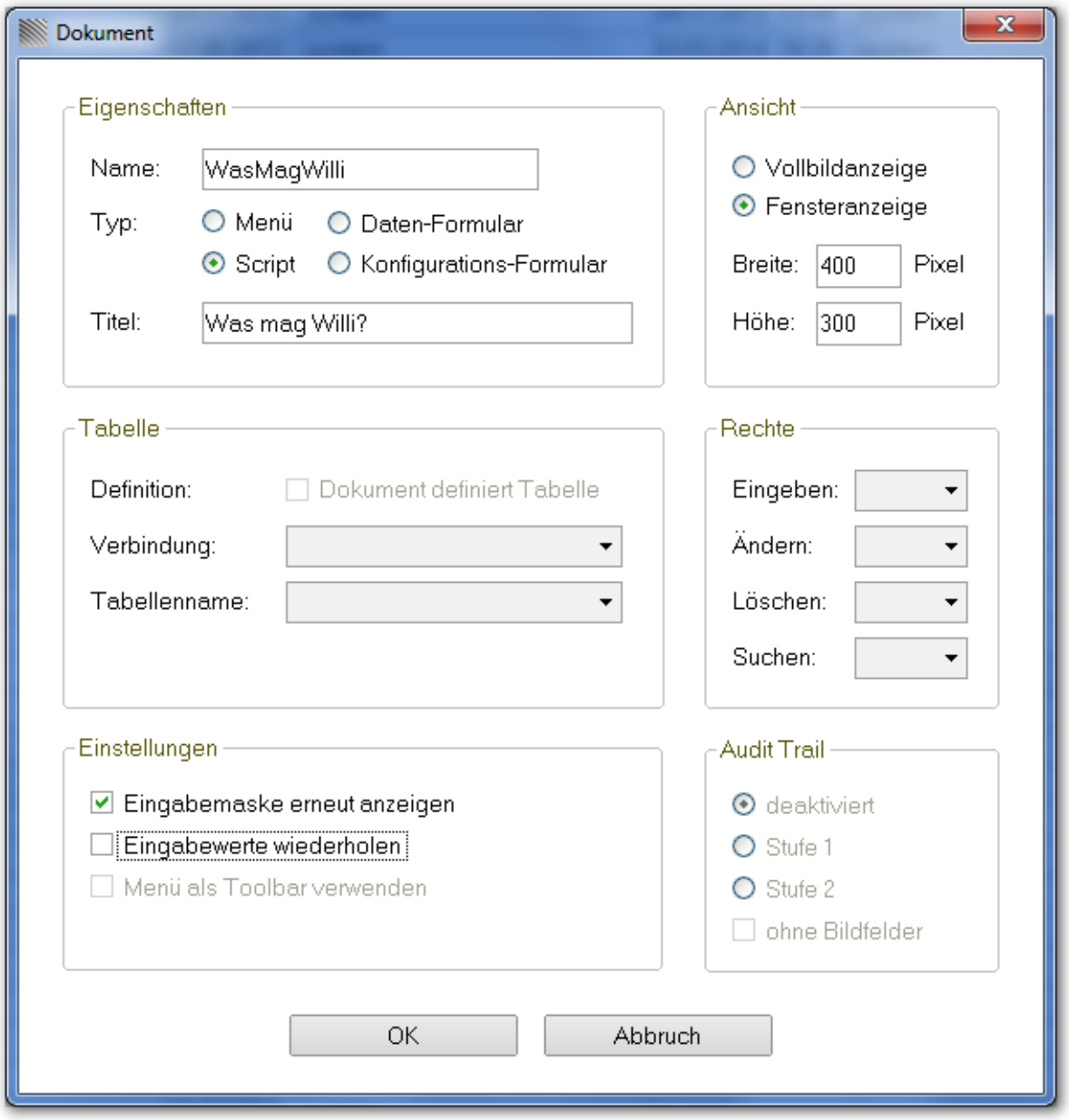
Die Handhabung des Script-Editors soll an einer einfachen Programmieraufgabe demonstriert werden.

Aufgabe

Wir programmieren das Spiel „Was mag Willi“. Das Script soll den Benutzer dazu auffordern, einen Gegenstand zu benennen. Sobald der Benutzer den Gegenstand eingegeben hat soll das Script dem Benutzer mitteilen, ob Willi den Gegenstand mag oder nicht. Dahinter steht ein einfaches Prinzip: Willi mag alle Gegenstände, in denen ein „ll“ vorkommt.

Lösung

Klicken Sie im Hauptmenü von DataCool auf den Button ***Scripte***. Wählen Sie anschließend die Option ***Neu*** oder drücken Sie die Taste Einfg. Es öffnet sich der Dialog zur Festlegung der Scripteigenschaften.



Name: Geben Sie einen Namen für das Script an. Ein Script darf nicht denselben Namen besitzen wie ein Formular. Leerzeichen und Sonderzeichen sind nicht erlaubt. Wir nennen unser Script „WasMagWilli“.

Typ: Der Dokumententyp ist „Script“.

Titel: Der Titel wird beim Start des Scriptes in der Eingabemaske in großer Schrift angezeigt.

Ansicht: Wählen Sie zwischen Vollbild- und Fensteranzeige. Die Einstellung ist nur für Scripte relevant, die eine Eingabemaske besitzen. In unserem Fall können Sie die Vorgabe von 400 x 300 Pixel übernehmen.

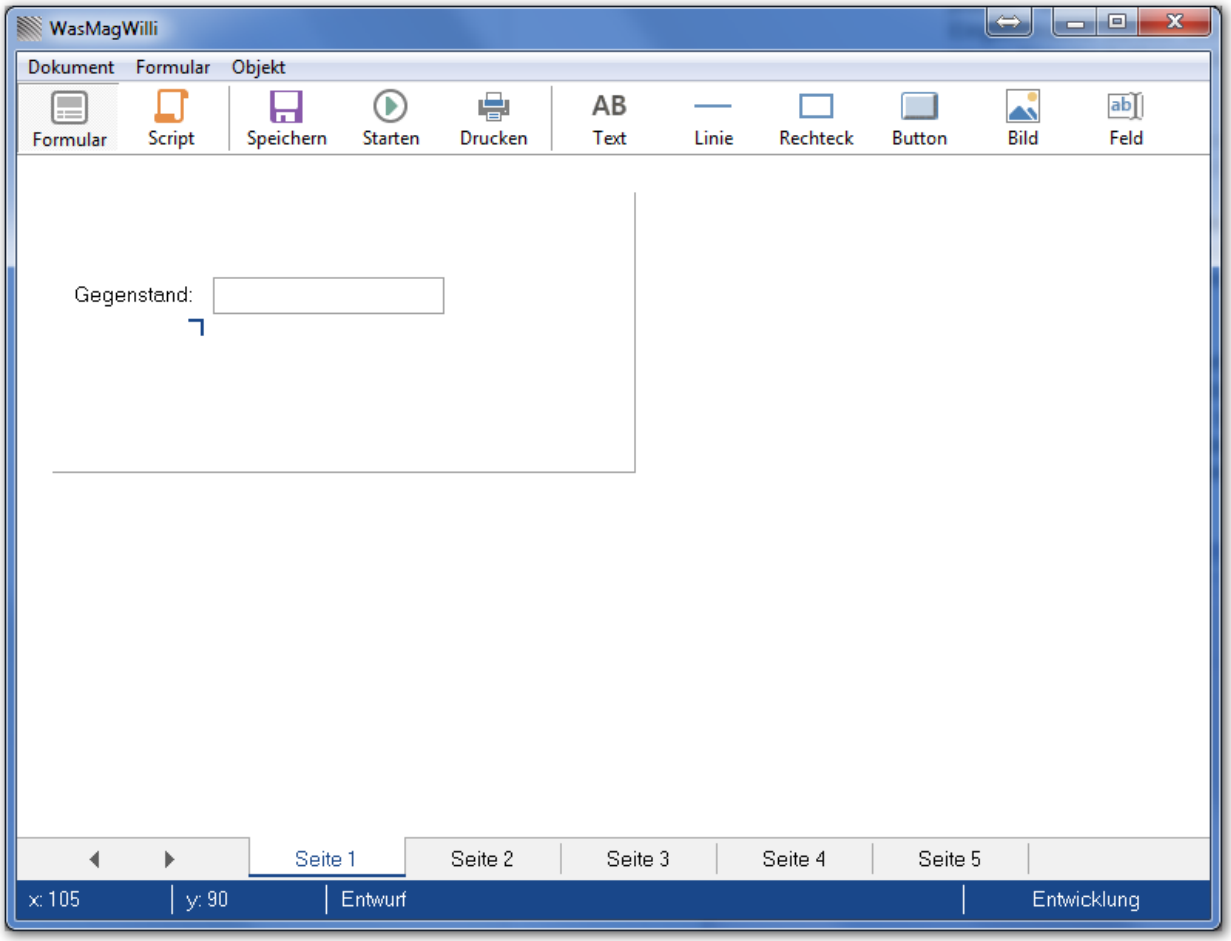
Script-Einstellungen:

Die Option „**Eingabemaske erneut anzeigen**“ ist nur für Scripte mit Eingabeformular relevant. Normalerweise wird das Script nach seiner Ausführung beendet. Ist die Option jedoch aktiviert, so springt das Script wieder zur Eingabemaske zurück. Der Benutzer kann dann mit einer neuen Eingabe fortfahren oder auf Abbruch klicken um das Script zu beenden.

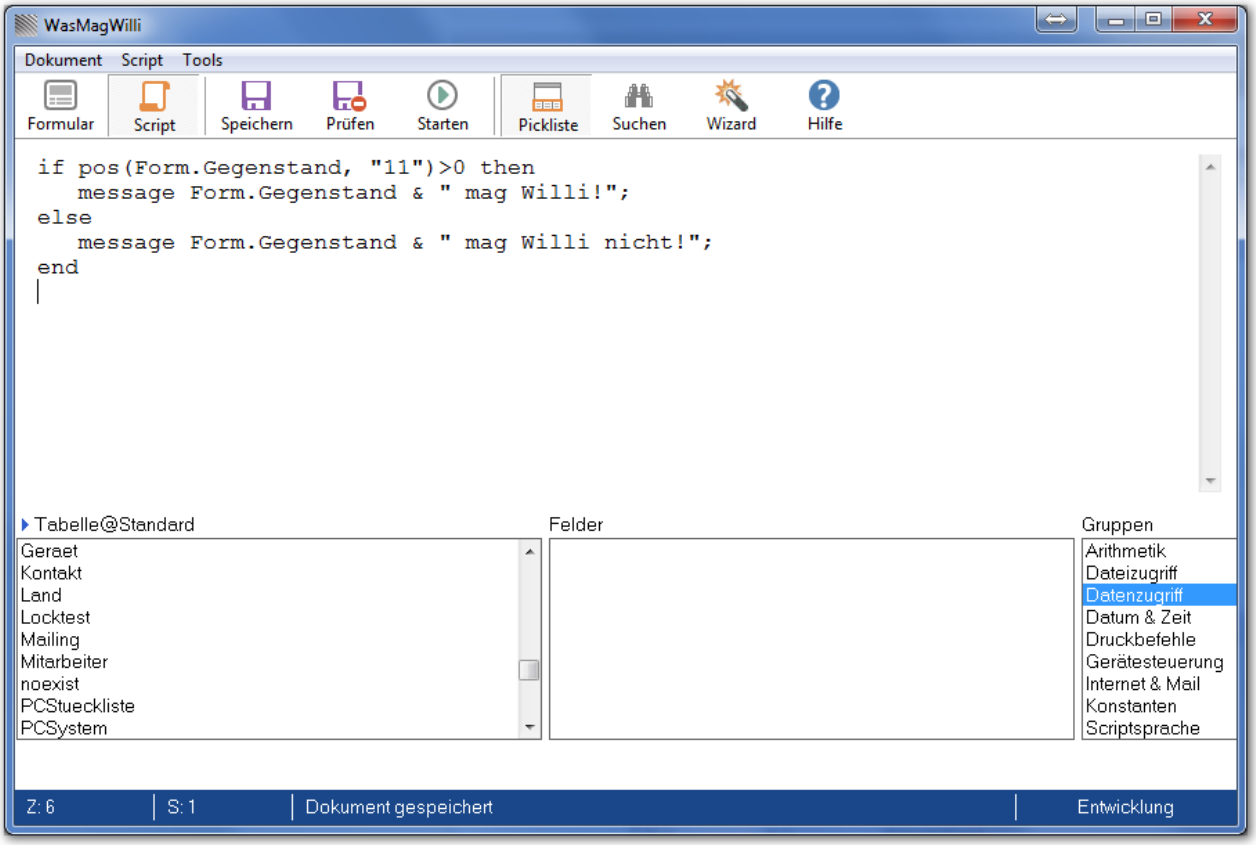
In unserem Beispiel haben wird diese Option aktiviert, damit der Benutzer beliebig viele Gegenstände vorschlagen kann.

Die Option „**Eingabewerte wiederholen**“ bestimmt, dass bei wiederholter Ausführung des Scriptes die zuletzt eingegebenen Werte automatisch vorgeschlagen werden.

Nachdem Sie den Button ***OK*** angeklickt haben befinden Sie sich automatisch im Script-Editor. Da unser Beispiel-Script vom Benutzer eine Eingabe erwarten soll, müssen wir jedoch zuerst ein Eingabeformular für das Script erstellen. Klicken Sie hierzu auf den Button ***Formular*** in der Toolbar. Erstellen Sie ein einfaches Formular mit einem Textfeld namens „Gegenstand“. Die grauen Begrenzungslinien deuten die spätere Größe des Formulars zur Laufzeit an.



Nun können wir mit der Erstellung des Programmes beginnen. Klicken Sie hierzu auf den Button ***Script*** in der Toolbar:



Zur besseren Lesbarkeit hier nochmals der Programmcode im Klartext:

```
if pos(Form.Gegenstand, "ll")>0 then
   message Form.Gegenstand & " mag Willi!";
else
   message Form.Gegenstand & " mag Willi nicht!";
end
```

Die pos-Funktion sucht nach der Zeichenfolge „ll“ im Feld „Gegenstand“. Auf die Werte der Eingabemaske können Sie mit dem Bezeichner **Form.Feldname** zugreifen. Wenn die pos-Funktion die Zeichenfolge „ll“ im Gegenstand findet, so erfolgt die Meldung, dass Willi den Gegenstand mag. Andernfalls erfolgt die Meldung, dass Willi den Gegenstand nicht mag.

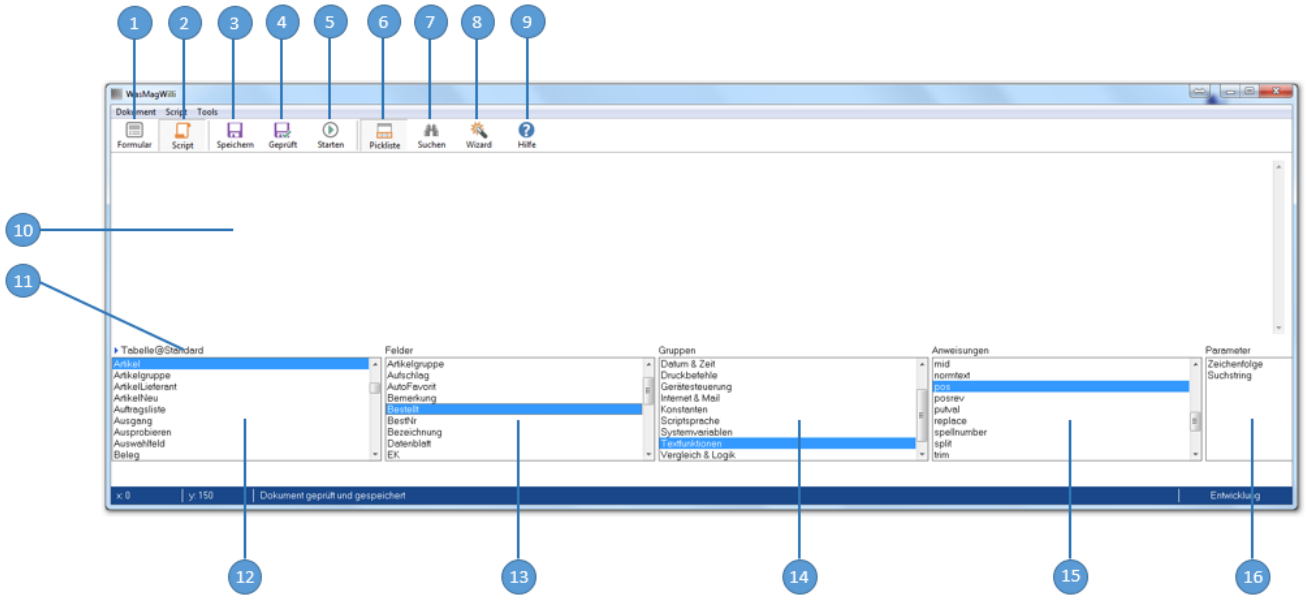
Nachdem Sie das Script eingegeben haben klicken Sie auf den Button „**Prüfen**“. DataCool überprüft die Syntax des Scriptes. Wenn alles korrekt ist, dann wird das Script abgespeichert und die Beschriftung des Buttons ändert sich in „**Geprüft**“. Im Falle eines Fehlers wird die falsche Programmzeile farbig markiert und das Script wird nicht gespeichert.

Wenn das Script fehlerfrei geprüft wurde, dann können Sie auf ***Start*** klicken um es auszuführen:

Das Script prüft nun ob ein „ll“ im Gegenstand vorkommt und gibt eine Meldung aus:

Zugegeben, dieses Beispiel hat noch nicht viel mit Datenverarbeitung zu tun. Dennoch haben Sie nun die ersten Handgriffe im Umgang mit dem Script-Editor kennengelernt.

Die Funktionen des Script-Editors im Detail:



1 Anzeige / Bearbeitung des Script-Formulars.

2 Anzeige / Bearbeitung des Scriptes (Programmcode).

3 Speicherung des Scriptes **ohne Prüfung**.

4 Speicherung des Scriptes **mit Prüfung**. Tritt ein Fehler auf, so wird die fehlerhafte Programmzeile rot markiert und das Script wird nicht gespeichert. Ist das Script fehlerfrei, so wird es gespeichert.

5 Start des Scriptes.

6 Schaltet die Pickliste aus oder ein. Die Pickliste ist eine Ansammlung von Auswahllisten im unteren Bereich des Editorfensters. Die Auswahllisten helfen dem Programmierer beispielsweise bei der Bestimmung von Feld- oder Tabellennamen.

7 Funktion zum Suchen und Ersetzen von Scripttext.

8 Aufruf des Script-Wizards. Der Script Wizard ist ein Listengenerator, der auf der Basis eines Dialoges automatisch Scriptcode erzeugt.

9 Aufruf der Online-Befehlsreferenz im Internet.

10 Editor-Bereich zur Bearbeitung des Programmcodes.

11 Umschaltung der Tabellenanzeige zwischen mehreren Verbindungen.

12 Auswahlliste für SQL-Tabellen. Per Doppelklick wird der Tabellenname in den Scripttext übernommen. Wird die Tabelle nur einmal angeklickt, so wird die nebenstehende Feldliste geladen.

13 Auswahlliste für Feldnamen. Per Doppelklick wird der Feldname in den Scripttext übernommen.

Neu ab Version 2014. wenn Sie mit dem Mauszeiger über einen Feldnamen fahren, dann wird eine Sprechblase mit dem Feldtyp angezeigt. Bei Auswahlfeldern sehen Sie außerdem eine Auflistung der Optionen.

14 Auswahlliste für Befehlsgruppen. Per Doppelklick wird die nebenstehende Anweisungsliste geladen.

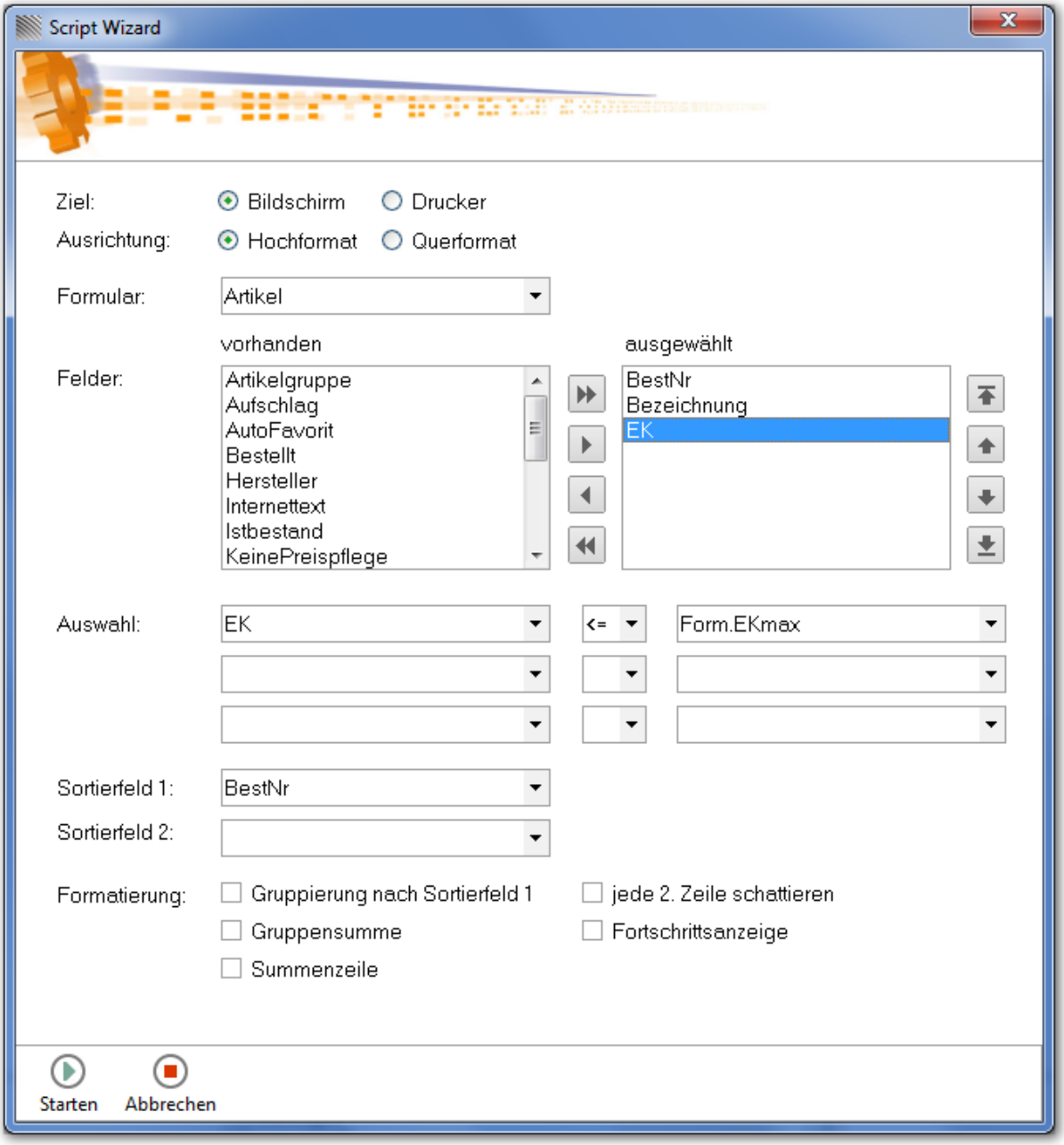
15 Auswahlliste für Funktionen und Befehle. Per Doppelklick wird die Beschreibung in der Online-Referenz im Internet aufgerufen. Wird eine Funktion nur einmal angeklickt, so wird die nebenstehende Parameterliste geladen.

10.3 Der Script-Wizard

Die Entwicklung von Anwendungen besteht zu einem guten Teil aus der Programmierung von Listen und Auswertungen. Zu diesem Zweck geben wir Ihnen ein wertvolles Werkzeug zur Hand: den **Script-Wizard**.

Der Script Wizard ist eine Art Listengenerator. Sie brauchen nicht mehr zu tun als einige Fragen zu beantworten. Der Wizard erzeugt aufgrund Ihrer Angaben ein fertiges Script. Sie können dieses Script sofort verwenden oder für Ihre Zwecke individuell anpassen. Auf diese Weise behalten Sie stets die volle Kontrolle über das Ergebnis. Der erzeugte Programmcode ist darüberhinaus gut zur Erlernung der Scriptsprache geeignet.

Der Script-Wizard ersetzt den Programmcode durch das automatisch generierte Script. Verwenden Sie den Wizard daher **nur für neue Scripte**.



Ziel: Wählen Sie das Ausgabeziel für die Liste: Bildschirm oder Drucker. Für die Bildschirmanzeige wird ein PDF Reader benötigt.

Ausrichtung: Bestimmen Sie ob die Liste im Hoch- oder Querformat angezeigt werden soll.

Formular: Wählen Sie das Formular, dessen Daten aufgelistet werden sollen.

Felder: Wählen Sie die Datenfelder, die in die Liste übernommen werden sollen. Mit Hilfe der Pfeil-Buttons können Sie die Auswahl leicht verändern oder die Reihenfolge der gelisteten Felder anpassen.

Auswahl. Geben Sie bei Bedarf ein Selektionskriterium an. Sie können auf der rechten Seite des Ausdrucks einen konstanten Wert oder ein Feld aus der Eingabemaske verwenden. Im letzteren Fall muss das Feld natürlich bereits existieren, bevor Sie den Wizard starten.

Sortierfelder: Geben Sie bis zu zwei Sortiermerkmale für die Liste an.

Formatierung:

Gruppierung nach Sortierfeld 1:

Es wird nach dem Sortierfeld 1 gruppiert. Dies bedeutet, dass bei einem Gruppenwechsel eine Gruppenüberschrift ausgegeben wird.

Gruppensumme:

Wenn Sie die Option „Gruppierung nach Sortierfeld 1“ aktiviert haben, dann können Sie hier eine Gruppensumme hinzufügen.

Summenzeile:

Bestimmen Sie, ob am Ende der Liste eine Summenzeile erscheinen soll.

Jede 2. Zeile schattieren:

Hinterlegen Sie jede zweite Datenzeile mit einem grauen Schatten, um die Lesbarkeit der Liste zu erhöhen. Diese Option steht nicht in Verbindung mit gruppierten Listen zur Verfügung.

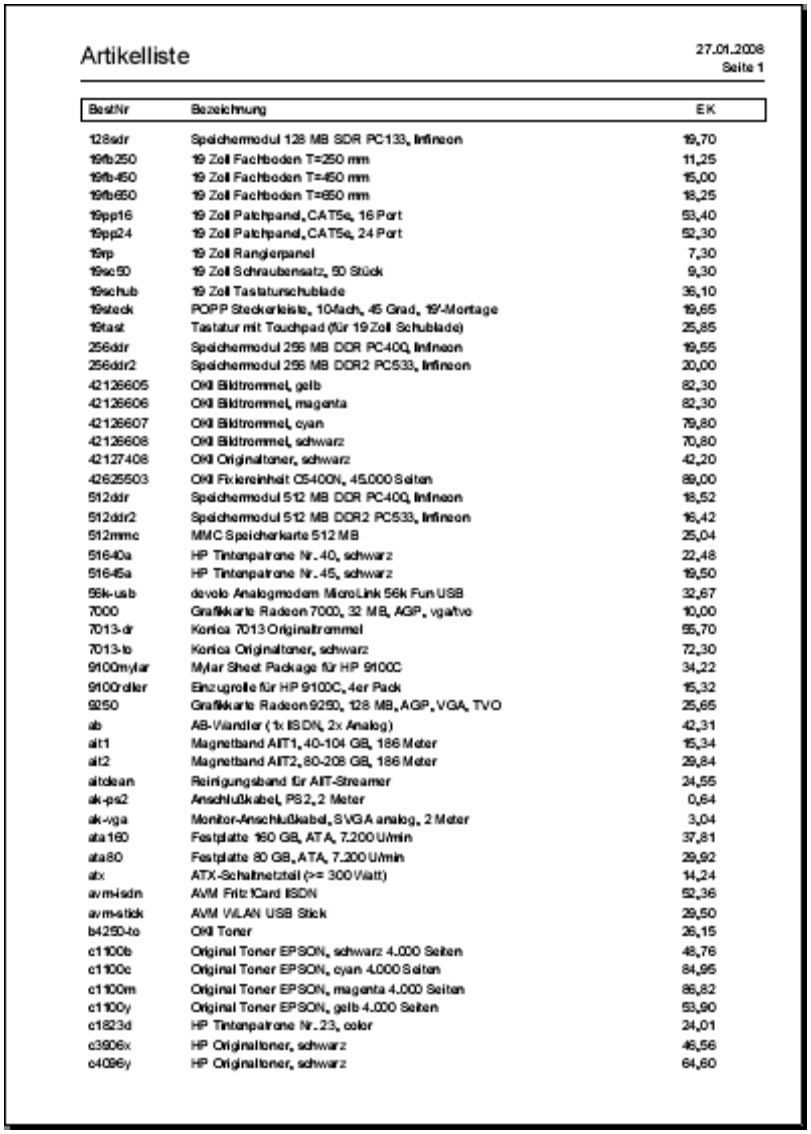
Fortschrittsanzeige:

Bestimmen Sie, ob bei der Ausgabe der Liste ein Fortschrittsbalken angezeigt werden soll.

Das folgende Beispielscript erzeugt eine Artikelliste. Es werden alle Artikel gelistet, deren Einkaufspreis (EK) einen vom Benutzer eingegebenen Wert (EKmax) nicht übersteigt:

```
dim L as number 3.0;      ' Länge des Druckbereichs in mm
dim B as number 3.0;      ' Breite des Druckbereichs in mm
dim P() as number 3.0;    ' Tabulatoren

L=275;
B=180;

P=array(2, 29, 167);

startjob screen, "", a4, 20, 10, 1;

select * from Artikel where EK<=@Form.EKmax order by BestNr;

   ' Seitenumbruch
   if L-current.ypos<current.height then
      newpage;
   end

   ' Seitenkopf
   if current.ypos=0 then
      font "Arial", "", 18;
      print "Artikelliste", left;
      font "Arial", "", 10;
      print current.date, right, B;
      newline;
      print "Seite " & current.page, right, B;
      newline;
      hline B;
      newline;
      rectangle B, 6;
      rmove 2, 1;
      print "BestNr",      left,  P(1);
      print "Bezeichnung", left,  P(2);
      print "EK",          right, P(3);
      newline;
      rmove 0, 3;
   end

   ' Datensatz
   print BestNr,      left,  P(1);
   print Bezeichnung, left,  P(2);
   print EK,          right, P(3);
   newline;

next

endjob;
```

Die fertige Liste sieht folgendermaßen aus:



Artikelliste

27.01.2006
Seite 1

BestNr	Bezeichnung	EK
128sdr	Speichermodul 128 MB SDR PC133, Infineon	19,70
19fb250	19 Zoll Fachboden T=250 mm	11,25
19fb450	19 Zoll Fachboden T=450 mm	15,00
19fb650	19 Zoll Fachboden T=650 mm	18,25
19pp16	19 Zoll Patchpanel, CAT5e, 16 Port	53,40
19pp24	19 Zoll Patchpanel, CAT5e, 24 Port	52,30
19rp	19 Zoll Rangierpanel	7,30
19sc50	19 Zoll Schraubensatz, 50 Stück	9,30
19schub	19 Zoll Tastaturschublade	36,10
19steck	POPP Steckerleiste, 10fach, 45 Grad, 19'-Montage	19,65
19tast	Tastatur mit Touchpad (für 19 Zoll Schublade)	25,85
256ddr	Speichermodul 256 MB DDR PC400, Infineon	19,55
256ddr2	Speichermodul 256 MB DDR2 PC533, Infineon	20,00
42126605	OKI Bildtrommel, gelb	82,30
42126606	OKI Bildtrommel, magenta	82,30
42126607	OKI Bildtrommel, cyan	79,80
42126608	OKI Bildtrommel, schwarz	70,80
42127408	OKI Originaltoner, schwarz	42,20
42625503	OKI Fixiereinheit C5400N, 45.000 Seiten	89,00
512ddr	Speichermodul 512 MB DDR PC400, Infineon	18,52
512ddr2	Speichermodul 512 MB DDR2 PC533, Infineon	16,42
512mmc	MMC Speicherkarte 512 MB	25,04
51640a	HP Tintenpatrone Nr. 40, schwarz	22,48
51645a	HP Tintenpatrone Nr. 45, schwarz	19,50
56k-usb	devolo Analogmodem MicroLink 56k Fun USB	32,67
7000	Grafikkarte Radeon 7000, 32 MB, AGP, vga/tvo	10,00
7013-dr	Konica 7013 Originaltrommel	55,70
7013-to	Konica Originaltoner, schwarz	72,30
9100mylar	Mylar Sheet Package für HP 9100C	34,22
9100roller	Einzugrolle für HP 9100C, 4er Pack	15,32
9250	Grafikkarte Radeon 9250, 128 MB, AGP, VGA, TVO	25,65
ab	AB-Wandler (1x ISDN, 2x Analog)	42,31
ait1	Magnetband AIT1, 40-104 GB, 186 Meter	15,34
ait2	Magnetband AIT2, 80-208 GB, 186 Meter	29,84
aitclean	Reinigungsband für AIT-Streamer	24,55
ak-ps2	Anschlußkabel, PS2, 2 Meter	0,64
ak-vga	Monitor-Anschlußkabel, SVGA analog, 2 Meter	3,04
ata160	Festplatte 160 GB, ATA, 7.200 U/min	37,81
ata80	Festplatte 80 GB, ATA, 7.200 U/min	29,92
atx	ATX-Schaltnetzteil (>= 300 Watt)	14,24
avm-isdn	AVM Fritz!Card ISDN	52,36
avm-stick	AVM WLAN USB Stick	29,50
b4250-to	OKI Toner	26,15
c1100b	Original Toner EPSON, schwarz 4.000 Seiten	48,76
c1100c	Original Toner EPSON, cyan 4.000 Seiten	84,95
c1100m	Original Toner EPSON, magenta 4.000 Seiten	86,82
c1100y	Original Toner EPSON, gelb 4.000 Seiten	53,90
c1823d	HP Tintenpatrone Nr. 23, color	24,01
c3906x	HP Originaltoner, schwarz	46,56
c4096y	HP Originaltoner, schwarz	64,60

Erläuterungen zum Script:

```
dim L as number 3.0;      ' Länge des Druckbereichs in mm
dim B as number 3.0;      ' Breite des Druckbereichs in mm
dim P() as number 3.0;    ' Tabulatoren
```

Als erstes werden drei Variablen mit der Anweisung **dim** reserviert. Bei den Variablen L und B handelt es sich um Ganzzahlen der Länge 3. Die Variable P ist ein Datenfeld (Array) aus Ganzzahlen der Länge 3. Das Datenfeld P wird zur Festlegung der Tabstopp-Positionen für die Datenfelder verwendet.

```
L=275;
```

Die Variable L wird mit einem Wert von 275 vorbelegt. Dies entspricht der Länge des bedruckbaren Bereiches in mm.

```
B=180;
```

Die Variable B wird mit einem Wert von 180 vorbelegt. Dies entspricht der Breite des bedruckbaren Bereiches in mm.

```
P=array(2, 29, 167);
```

Das Datenfeld P wird mit den Werten 2, 29 und 167 vorbelegt. Dies entspricht der x-Position, an der die Felder BestNr, Bezeichnung und EK gedruckt werden sollen.

```
startjob screen, "", a4, 20, 10, 1;
```

Der Befehl **startjob** startet einen Druckauftrag. Die Parameter legen folgendes fest: Ausgabeziel: **Bildschirm**, Name des Druckauftrags: keiner, Papierformat: **A4**, linker Rand: **20 mm**, oberer Rand: **10 mm**, Exemplare: **1**.

```
select * from Artikel where EK<=@Form.EKmax order by BestNr;
```

Es werden alle Artikeldatensätze geladen, bei denen der EK kleiner oder gleich dem Maximal-EK aus dem Eingabeformular ist. Bitte beachten das Zeichen @ in der where-Klausel. Es bewirkt eine Ersetzung des Feldnamens durch den Wert aus der Eingabemaske. Zu jedem select-Befehl gehört eine next-Anweisung. Der Programmcode zwischen diesen beiden Anweisungen wird für jeden selektierten Datensatz einmal durchlaufen.

```
' Seitenumbruch
if L-current.ypos<current.height then
   newpage;
end
```

Die Systemvariable **current.ypos** gibt die aktuelle Cursorposition in y-Richtung auf dem Papier zurück. Der Ausdruck L-current.ypos ergibt somit den verbleibenden Rest auf dem Papier. Ist dieser kleiner als **current.height** (die Höhe einer Druckzeile), so ist ein Seitenumbruch erforderlich. Der Seitenumbruch wird mit Hilfe des Befehls **newpage** ausgelöst.

```
if current.ypos=0 then
```

Zu Beginn der Liste (wenn der erste Datensatz ausgeben wird) oder nach einem Seitenwechsel befindet sich der Cursor an der y-Position 0. In dieser Situation muss der Seitenkopf der Liste ausgegeben werden bevor der nächste Datensatz gedruckt werden kann.

```
font "Arial", "", 18;
print "Artikelliste", left;
font "Arial", "", 10;
print current.date, right, B;
newline;
print "Seite " & current.page, right, B;
newline;
hline B;
newline;
```

Mit den obigen Befehlen wird der Titel der Liste gedruckt. Der Schriftzug „Artikelliste“ wird mit der Schrift Arial 18 ausgeben. Danach wird die Schriftgröße auf 10 Punkt umgestellt und das aktuelle Datum rechtsbündig ausgegeben. Mit **newline** wird ein Zeilenwechsel ausgeführt. Unterhalb des Datums wird die aktuelle Seitenzahl mit der Systemvariable **current.page** ausgegeben. Der Befehl **hline** zeichnet eine horizontale Trennlinie mit der Breite B.

```
rectangle B, 6;
rmove 2, 1;
print "BestNr",      left,  P(1);
print "Bezeichnung", left,  P(2);
print "EK",          right, P(3);
newline;
rmove 0, 3;
```

Mit den obigen Befehlen wird die Spaltenüberschrift erzeugt. Der Befehl r**ectangle** zeichnet ein Rechteck mit der Breite B und der Höhe 6 mm. Die Anweisung **rmove** bewegt der Cursor relativ zur linken oberen Ecke des Rechtecks um 2 mm nach rechts und 1 mm nach unten. Die nachfolgenden **print**-Anweisungen drucken jeweils einen Spaltentitel an der Tabposition P(x). Der Text „BestNr“ und „Bezeichnung“ wird linksbündig ausgerichtet. Der Text „EK“ wird rechtsbündig ausgerichtet. Mit **newline** und **rmove** wird der Cursor an die Startposition für den nachfolgenden Datensatz gebracht.

```
end
```

Die **end**-Anweisung gehört zu der **if**-Bedingung für den Seitenkopf. Der Programmcode zwischen **if** und **end** wird nur dann ausgeführt, wenn die Bedingung der **if**-Anweisung erfüllt ist.

```
' Datensatz
print BestNr,      left,  P(1);
print Bezeichnung, left,  P(2);
print EK,          right, P(3);
newline
```

Die **print**-Befehle dienen zur Ausgabe der Datenfelder BestNr, Bezeichnung und EK aus der SQL-Tabelle. Die Ausgabe erfolgt an der jeweiligen Tabposition. Nach jedem Datensatz wird die Zeile mit dem Befehl **newline** weitergeschaltet.

```
next
```

Die **next**-Anweisung gehört zum **select**-Befehl weiter oben. Der Programmcode zwischen beiden Anweisungen wird für jeden selektierten Datensatz einmal ausgeführt. Die Befehle **select** und **next** bilden daher eine Schleifenanweisung.

```
endjob;
```

Mit der Anweisung **endjob** wird der Druckauftrag beendet. Diese Anweisung darf nicht fehlen, da sonst keine Ausgabe erfolgt.

10.4 Befehlsübersicht

Die Befehlsübersicht listet alle DataCool Sprachelemente nach Funktionsgruppen auf. Dies soll Ihnen das Auffinden einer benötigten Funktion oder eines Befehls erleichtern. Neue oder geänderte Sprachelemente ab Version 2014 sind rot markiert.

Arithmetik

+	floor	random
-	labvalue	round
*	log	sequence
/	count	sin
abs	mod	sqrt
ceil	occurence	sum
cos	power	tan
count	price	

Dateizugriff

close	savepicture	writehtml
copyfile	selectfile	writexml
create	unzip	xmlclose
createdir	wordclose	xmleof
deletefile	wordopen	xmlopen
getdir	wordprint	xmlread
eof	wordreplace	xmlvalue
open	wordsave	zip
read	write	
renamefile	writebase64	

Datenzugriff

commit	rollback	truncate
delete	select	update
fetch	sqldirect	
insert	transaction	

Datum & Zeit

age	m2time	spellweekday
date	minutes	time2h
datediff	month	time2m
day	setyear	timediff
formatdate	shiftday	timespan
h2time	shiftmonth	weekday
hours	shiftyear	year
lastday	spellmonth	yearweek

Druckbefehle

addfont
align
barcode
colorbar
endjob
font
height
hline
line
move
newline
newpage
picture
print
rectangle
rmove
shadow
source
startjob
tab
vline

Gerätesteuerung

closecom
opencom
readcom
secpos
writecom

Internet & Mail

connect
disconnect
gethtml
getrest
getsoap
sendmail
startbrowser
startmailer

Konstanten

noexist
Allgemein
Ausgabeziel
Ausrichtung
Barcodetyp
Berechtigung
Farbangaben
Infofenster
Papierformate

Scriptsprache

array
case
choice
choicenum
clearform
closewin
dim
error
exit
export
focus
if-Anweisung
if-Funktion
import
info
input
inputbulk
message
opendoc
openwin
progress
public
query
repeat
run
start
while

Systemvariablen

current.check	current.level	current.tempdir
current.date	current.page	current.time
current.desktopdir	current.progdir	current.user
current.dsn	current.reccount	current.userdir
current.event	current.reclock	current.xpos
current.file	current.recnum	current.ypos
current.form	current.recpos	
current.height	current.station	

Textfunktionen

““	lastword	split
&	len	trim
asc	lower	upper
chr	md5	utf2win
decrypt	memolines	vatcheck
dos2win	memosize	vatsample
encrypt	mid	win2dos
fill	normtext	win2utf
first	pos	win2xml
firstword	posrev	xml2win
formatnumber	putval	
getval	replace	
last	spellnumber	

Vergleich & Logik

()	=	<>
<	>	and
<=	>=	or

10.5 Referenzteil

Der nachfolgende Referenzteil listet zuerst alle Operatoren auf. Danach folgen die DataCool Befehle in alphabetischer Reihenfolge (unabhängig von der Zuordnung zu einer Funktionsgruppe). Der Referenzteil ist somit als Nachschlagewerk geeignet.

+ (Addition)

Typ: Mathematischer Operator

Syntax: *Zahl1 + Zahl2*

Rückgabewert: Eine Zahl

Beschreibung: Addiert **Zahl1** und **Zahl2**.

Sofern erforderlich, werden die Operanden in einen Zahlenwert konvertiert.
Ist die Konvertierung nicht möglich, so wird mit dem Wert 0 gerechnet.
In mathematischen Ausdrücken gilt die Regel "Punkt vor Strich".

Beispiel:

```
2 + 5 * 2
```

```
12
```

- (Subtraktion)

Typ: Mathematischer Operator

Syntax: *Zahl1 - Zahl2*

Rückgabewert: Eine Zahl

Beschreibung: Subtrahiert **Zahl2** von **Zahl1**.

Sofern erforderlich, werden die Operanden in einen Zahlenwert konvertiert. Ist die Konvertierung nicht möglich, so wird mit dem Wert 0 gerechnet. In mathematischen Ausdrücken gilt die Regel "Punkt vor Strich".

Beispiel:

```
20 - 5 * 2
```

```
10
```

* (Multiplikation)

Typ: Mathematischer Operator

Syntax: *Zahl1 * Zahl2*

Rückgabewert: Eine Zahl

Beschreibung: Multipliziert **Zahl1** mit **Zahl2**.

Sofern erforderlich, werden die Operanden in einen Zahlenwert konvertiert.
Ist die Konvertierung nicht möglich, so wird mit dem Wert 0 gerechnet.
In mathematischen Ausdrücken gilt die Regel "Punkt vor Strich".

Beispiel:

```
2 + 5 * 2
```

```
12
```

/ (Division)

Typ: Mathematischer Operator

Syntax: *Zahl1* / *Zahl2*

Rückgabewert: Eine Zahl

Beschreibung: Dividiert **Zahl1** durch **Zahl2**.

Sofern erforderlich, werden die Operanden in einen Zahlenwert konvertiert.
Ist die Konvertierung nicht möglich, so wird mit dem Wert 0 gerechnet.
In mathematischen Ausdrücken gilt die Regel "Punkt vor Strich".

Beispiel:

```
2 + 6 / 2
```

```
5
```

" " (Stringkonstante)

Typ: Interpunktionssymbol

Syntax: "*Textkonstante*"

Beschreibung: Anführungszeichen werden dazu verwendet, eine **Texkonstante** einzugrenzen. Innerhalb der Texkonstante können folgende Sonderzeichen vorkommen:

\" = Anführungszeichen
\t = Tabulator (Hex: 09)
\n = Zeilenschaltung (Hex: 0D0A)
\\ = Backslash (Hex: 5C)

Beispiel:

```
"Das ist die Dokumentation \n von \"DataCool\""
```

```
Das ist die Dokumentation
von "DataCool"
```

& (Verkettung)

Typ: Operator

Syntax: *Zeichenfolge1* & *Zeichenfolge2*

Rückgabewert: Eine Zeichenfolge

Beschreibung: Der Operator verkettet **Zeichenfolge1** mit **Zeichenfolge2** zu einer langen Zeichenfolge.

Beispiel:

```
"Data" & "Cool"
```

```
DataCool
```

() (Klammerung)

Typ: Interpunktionssymbol

Syntax: (*Ausdruck*)

Beschreibung: Die runden Klammern werden als Trennungszeichen für Funktionsparameter sowie zur Bestimmung der Reihenfolge in mathematischen und logischen Operationen benutzt.

siehe auch: and, or

Beispiel:

```
(7 - 2) * (10 / 5)
```

```
10
```

< (kleiner)

Typ: Vergleichsoperator

Syntax: *Ausdruck1* < *Ausdruck2*

Beschreibung: Der Operator wird zum Vergleich zweier Ausdrücke verwendet.

Ist der **Ausdruck1** kleiner als **Ausdruck2**, so ist das Ergebnis *true*, ansonsten *false*.

Beispiel:

```
2 < 7
```

true

<= (kleiner gleich)

Typ: Vergleichsoperator

Syntax: *Ausdruck1* <= *Ausdruck2*

Beschreibung: Der Operator wird zum Vergleich zweier Ausdrücke verwendet.

Ist der **Ausdruck1** kleiner gleich **Ausdruck2**, so ist das Ergebnis *true*, ansonsten *false*.

Beispiel:

```
7 <= 7
```

true

= (gleich)

Typ: Vergleichsoperator

Syntax: *Ausdruck1* = *Ausdruck2*

Beschreibung: Der Operator wird zum Vergleich zweier Ausdrücke verwendet.

Sind **Ausdruck1** und **Ausdruck2** gleich, so ist das Ergebnis *true*, ansonsten *false*.

Beispiel:


7 = 2 + 5


true

> (größer)

Typ: Vergleichsoperator

Syntax: *Ausdruck1 > Ausdruck2*

Beschreibung: Der Operator wird zum Vergleich zweier Ausdrücke verwendet.

Ist der **Ausdruck1** größer als **Ausdruck2**, so ist das Ergebnis *true*, ansonsten *false*.

Beispiel:


7 > 2


true

>= (größer gleich)

Typ: Vergleichsoperator

Syntax: *Ausdruck1 >= Ausdruck2*

Beschreibung: Der Operator wird zum Vergleich zweier Ausdrücke verwendet.

Ist der **Ausdruck1** größer gleich **Ausdruck2**, so ist das Ergebnis *true*, ansonsten *false*.

Beispiel:

```
7 >= 2
```

true

<> (ungleich)

Typ: Vergleichsoperator

Syntax: *Ausdruck1* <> *Ausdruck2*

Beschreibung: Der Operator wird zum Vergleich zweier Ausdrücke verwendet.

Sind **Ausdruck1** und **Ausdruck2** verschieden, so ist das Ergebnis *true*, ansonsten *false*.

Beispiel:

```
2 <> 7
```

true

and (logisches und)

Typ: Logischer Operator

Syntax: *Ausdruck1* and *Ausdruck2*

Beschreibung: Der Operator verbindet zwei verschiedene logische Ausdrücke.

Haben beide Ausdrücke den Wert *true*, so ist das Ergebnis *true*. In allen anderen Fällen ist das Ergebnis *false*.

Wenn die Operatoren *and* und *or* kombiniert werden, müssen die Kriterien in runde Klammern gesetzt werden, damit der Sinnzusammenhang ersichtlich wird.

siehe auch: (), or

Beispiel:

```
2 < 7 and 8 < 7
```

false

or (logisches oder)

Typ: Logischer Operator

Syntax: *Ausdruck1* or *Ausdruck2*

Beschreibung: Der Operator verbindet zwei verschiedene logische Ausdrücke.

Hat *Ausdruck1* oder *Ausdruck2* den Wert *true*, so ist das Ergebnis *true*. Hat keiner der beiden Ausdrücke den Wert *true*, so ist das Ergebnis *false*.

Wenn die Operatoren *and* und *or* kombiniert werden, müssen die Kriterien in runde Klammern gesetzt werden, damit der Sinnzusammenhang ersichtlich wird.

siehe auch: (), and

Beispiel:

```
2 < 7 or 8 < 7
```

true

abs

Typ: Funktion

Syntax: *abs(Zahl)*

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion bestimmt den Absolutbetrag der **Zahl**.
Der Absolutbetrag ist der vorzeichenlose (positive) Wert.

Beispiel:

```
abs(-10)
10
```

```
abs(10)
10
```

addfont

Typ: Druckbefehl

Syntax: addfont *Schriftart, Schriftdatei*

Beschreibung: Bei dem Befehl **startjob** können die Ausgabeziele *printer*, *screen* und *pdf* festgelegt werden. Bei der Ausgabe auf dem Drucker stehen hierbei alle auf dem Rechner installierten Schriftarten zur Verfügung. Die Ausgabeziele *screen* und *pdf* hingegen unterstützen nur die Schriftarten Arial, Times und Courier. Mit dem Befehl **addfont** können weitere Schriftarten in die PDF-Datei eingebettet werden. Auf diese Weise stehen beliebige Schriften im PDF-Dokument zur Verfügung. Der mit **addfont** eingebundene Truetype-Font muss auf dem Rechner installiert sein, auf dem das Druckscript läuft. Die PDF-Datei selbst kann dann auf jedem beliebigen Gerät angezeigt werden, auch wenn der Font nicht auf dem Anzeige-Gerät installiert ist.

Schriftart	Name der Schriftart, so wie er in Anwendungsprogrammen (z.B. MS Word) dargestellt wird. Sie können den korrekten Schriftnamen in der Systemsteuerung unter Schriftarten abrufen, wenn Sie eine Schrift per Doppelklick öffnen.
Schriftdatei	Dateiname der Truetype-Schrift inklusive Pfadangabe. Es werden die Dateiformate TTF und OTF unterstützt.

siehe auch: font

Beispiel:

```
startjob screen;

addfont "Calibri Light",
        "c:\\windows\\fonts\\calibril.ttf";

font "Calibri Light", "", 11;

print "Hello World!";

endjob;
```

verfügbar ab Build 1177 (10/2018)

age

Typ: Funktion

Syntax: age(*Geburtsdatum [, Stichtag]*)

Rückgabewert: Eine Ganzzahl

Beschreibung: Die Funktion errechnet das Alter eines Menschen aus dem **Geburtsdatum**. Fehlt die optionale Angabe **Stichtag**, so wird das Alter am heutigen Tag ermittelt.

Beispiel:


```
age(24.12.1980, 24.12.2000)
20
```


```
age(Geburtsdatum)
Alter am heutigen Tag
```

align

Typ: Druckbefehl

Syntax: align *Ausrichtung*

Beschreibung: Dieser Befehl legt die Ausrichtung für nachfolgende *print* Befehle fest. Folgende Werte sind möglich:

left	linksbündig
center	mittig
right	rechtsbündig

siehe auch: print

Beispiel:

```
align center;
```

Allgemein (Konstanten)

Typ: Konstante

Werte:

Name	Wert	Bedeutung
null	null	undefiniert (nicht ausgefüllt)
false	0	logischer Wert: falsch
true	1	logischer Wert: wahr
Zip	1	E-Mail Anlage komprimieren
NoZip	2	E-Mail Anlage nicht komprimieren
Checksum	1	Barcode wird mit Prüfsumme gedruckt
NoChecksum	2	Barcode wird ohne Prüfsumme gedruckt

array

Typ: Funktion

Syntax: array(*Wert1, Wert2, ...*)

Rückgabewert: Ein Array

Beschreibung: Die Funktion *array* wird benutzt, um ein Datenfeld (array) mit Werten zu belegen. Ein Aufruf der Funktion ohne Parameter bewirkt eine Initialisierung des Datenfeldes.

siehe auch: dim

Beispiel:

```
dim Farbe() as text 10;
Farbe = array("rot", "grün", "blau");
i=0;
while i
   i=i+1;
   message Farbe(i);
wend
```

asc

Typ: Funktion

Syntax: asc(*Zeichenfolge*)

Rückgabewert: Eine Ganzzahl

Beschreibung: Die Funktion gibt den ASCII-Code des ersten Buchstabens der **Zeichenfolge** zurück. Eine leere **Zeichenfolge** ergibt den Wert 0.

siehe auch: chr

Beispiel:

```
asc("A")
```

```
65
```

```
asc("DataCool")
```

```
68
```

Ausgabeziel (Konstanten)

Typ: Konstante

Werte:

Name	Wert	Bedeutung
printer	1	Ausgabe auf Drucker
screen	2	Ausgabe auf Bildschirm
pdf	3	Ausgabe in PDF-Datei

Ausrichtung (Konstanten)

Typ: Konstante

Werte:

Name	Wert	Bedeutung
left	1	linksbündig
center	2	mittig
right	3	rechtsbündig

barcode

Typ: Druckbefehl

Syntax: barcode *Wert, Barcodetyp, Prüfziffer, Höhe, Modulbreite, [Korrekturfaktor]*

Beschreibung: Dieser Befehl druckt den **Wert** als Strichcode vom **Barcodetyp**.
Die linke obere Ecke des Barcodes wird an der aktuellen Cursorposition platziert.
Der **Wert** darf nur Zeichen enthalten, die im Zeichensatz des Barcodetyps vorkommen.
Der Parameter **Prüfziffer** kann die Werte *Checksum* und *NoChecksum* annehmen.
Die **Höhe** bestimmt die Höhe des gedruckten Strichcodes in mm.
Die **Modulbreite** bestimmt die Breite eines schmalen Elements in mm.

Der **Korrekturfaktor** ist optional und dient zum Ausgleich von Drucktoleranzen bei Nadeldruckern. Er gibt den Faktor an, um den eine schmale Lücke vergrößert werden soll, damit sie gleich groß erscheint wie ein schmaler Strich. Der Faktor kann Werte zwischen 0 und 200% annehmen. Bei Laserdruckern kann dieser Parameter entfallen.

Barcodetyp:

Barcodetyp	Wert	Zeichensatz
Code25Interleaved	1	0123456789
Code25Industrial	2	0123456789
Code25Matrix	3	0123456789
Code128C	4	0123456789
Codabar	5	0123456789-$:/.+ABCD
Code39	6	0123456789 ABCDEFGHIJKLMNOPQRSTUVWXYZ -.$/+%
Code39E	7	!"#$%&'()*+,-./0123456789:;<=>?@ ABCDEFGHIJKLMNOPQRSTUVWXYZ []ÄÖÜ^_´abcdefghijklmnopqrstuvwxyz{|}äöüß
Code93	8	0123456789 ABCDEFGHIJKLMNOPQRSTUVWXYZ -.$/+%
Code93E	9	!"$%&#()*+,-./0123456789:;<=>?@ ABCDEFGHIJKLMNOPQRSTUVWXYZ []ÄÖÜ^_´abcdefghijklmnopqrstuvwxyz{|}äöüß
EAN13	10	0123456789
EAN8	11	0123456789
UPCA	12	0123456789

Beispiel:

```
barcode "012345", Code39, NoChecksum, 10, 0.3;
```

Fehlerbehandlung: Bei Problemen wird anstelle des Barcodes ein Fehlertext ausgegeben:

Fehler	Beschreibung
Error 1	Die Barcodelänge muss zwischen 1 und 30 Zeichen liegen
Error 2	Unzulässiger Wert für den Barcodetyp
Error 3	Unzulässiger Wert für den Parameter Prüfziffer
Error 4	Die Höhe des Barcodes muss zwischen 5 und 50 mm liegen
Error 5	Der Wert für die Elementbreite muss zwischen 0.1 und 2 liegen
Error 6	Der Korrekturfaktor muss zwischen 0 und 200% liegen
Error 7	Der Barcodewert enthält ein ungültiges Zeichen
Error 8	Sonstiger Fehler

Barcodetyp (Konstanten)

Typ: Konstante

Werte:

Barcodetyp	Wert	Zeichensatz
Code25Interleaved	1	0123456789
Code25Industrial	2	0123456789
Code25Matrix	3	0123456789
Code128C	4	0123456789
Codabar	5	0123456789-$:/.+ABCD
Code39	6	0123456789 ABCDEFGHIJKLMNOPQRSTUVWXYZ -.$/+%
Code39E	7	!"#$%&'()*+,-./0123456789:;<=>?@ ABCDEFGHIJKLMNOPQRSTUVWXYZ []ÄÖÜ^_´abcdefghijklmnopqrstuvwxyz{|}äöüß
Code93	8	0123456789 ABCDEFGHIJKLMNOPQRSTUVWXYZ -.$/+%
Code93E	9	!"$%&#()*+,-./0123456789:;<=>?@ ABCDEFGHIJKLMNOPQRSTUVWXYZ []ÄÖÜ^_´abcdefghijklmnopqrstuvwxyz{|}äöüß
EAN13	10	0123456789
EAN8	11	0123456789
UPCA	12	0123456789

Berechtigung (Konstanten)

Typ: Konstante

Werte:

Name	Wert	Bedeutung
low	1	niedrige Berechtigungsstufe
medium	2	mittlere Berechtigungsstufe
high	3	hohe Berechtigungsstufe
admin	4	Administrator

case

Typ: Befehl

Syntax:

```
case Ausdruck;
value Wert1:
  Anweisung1;
value Wert2:
  Anweisung2;
...
else
  Alternativanweisung;
end
```

Beschreibung: Die case-Anweisung entscheidet anhand einer Reihe von Vergleichswerten, welche Anweisungsfolge ausgeführt wird. Gibt es keine Übereinstimmung des Ausdrucks mit einem der Vergleichswerte, so wird die Alternativanweisung nach dem Schlüsselwort *else* ausgeführt.

siehe auch: if-Anweisung, if-Funktion

Beispiel:

```
case current.level;
value low:
   message "Berechtigungsstufe: niedrig";
value medium:
   message "Berechtigungsstufe: mittel";
value high:
   message "Berechtigungsstufe: hoch";
value admin:
   message "Berechtigungsstufe: admin";
end
```

ceil

Typ: Funktion

Syntax: ceil(*Zahl, n*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion rundet die **Zahl** auf **n** Nachkommastellen auf.

siehe auch: floor, formatnumber, labvalue , round

Beispiel:

```
ceil(10.3, 0)
11

ceil(10.462, 2)
10.47

ceil(-10.4, 0)
-10
```

choice

Typ: Funktion

Syntax: choice(*Auswahlname, n*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Rückgabe des Auswahltextes mit der Nummer **n** aus einer Auswahlliste.

siehe auch: choicenum

Beispiel:

```
choice("Belegart", 2)
Gutschrift
```

choicenum

Typ: Funktion

Syntax: choicenum(*Auswahlname, Auswahltext*)

Rückgabewert: Eine Zahl

Beschreibung: Rückgabe der Nummer von **Auswahltext** aus einer Auswahlliste.

siehe auch: choice

Beispiel:

```
choicenum("Belegart", "Gutschrift")
```

```
2
```

chr

Typ: Funktion

Syntax: chr(*Zeichencode*)

Rückgabewert: Ein Zeichen

Beschreibung: Die Funktion gibt das Zeichen zurück, das dem angegebenen ASCII-Code entspricht.

siehe auch: asc

Beispiel:

```
chr(65)
```

```
A
```

clearform

Typ: Befehl

Syntax: clearform

Beschreibung: Mit diesem Befehl wird das aktuelle Eingabeformular geleert und dar Status auf 'neu' gesetzt.

Beispiel:

```
if Status="erfasst" then
   clearform;
end
```

close

Typ: Befehl

Syntax: close

Beschreibung: Der Befehl schließt die mit *open* oder *create* geöffnete Datei.

siehe auch: open, create

Beispiel:

```
if create("c:\\test.txt") = true then
   write "Hello World!\n";
   close;
end
```

closecom

Typ: Befehl

Syntax: closecom

Beschreibung: Dieser Befehl schließt den mit opencom geöffneten COM-Port.

siehe auch: opencom, readcom, writecom

Beispiel:

```
dim Num as text 10;

if opencom(1, "9600,N,8,1", true)=false then
   message "Fehler beim Initialisieren von COM1.";
   exit script;
end

truncate table BarcodeInventur;

repeat
   Num=readcom(chr(13));
   if Num<>noexist then
      insert into BarcodeInventur set Nummer=Num;
   end
until Num=noexist;

closecom;
```

closewin

Typ: Befehl

Syntax: closewin

Beschreibung: Dieser Befehl schließt das Fenster, das zuvor mit *openwin* geöffnet wurde.

siehe auch: openwin

Beispiel:

```
openwin infotext, progressbar;
select * from Kunde where Umsatz>1000;
   info current.recpos;
   progress current.reccount, current.recpos;
next
closewin;
```

colorbar

Typ: Druckbefehl

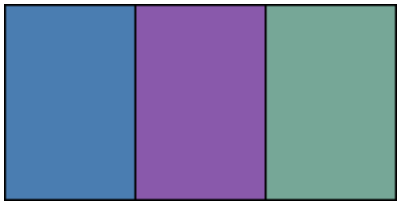
Syntax: colorbar *Breite*, *Höhe*, *Farbwert*

Beschreibung: Zeichnet ein gefülltes Rechteck mit den angegebenen Werten für Breite und Höhe. Das Rechteck beginnt mit der linken oberen Ecke an der aktuellen Cursorposition. Die Cursorposition bleibt unverändert. Breite und Höhe werden in mm angegeben. Der Parameter Farbwert verändert die Füll- und Linienfarbe des Rechtecks. Die Farbangabe erfolgt entweder über eine der festgelegten Farbkonstanten oder einen RGB-Wert.

siehe auch: rectangle, shadow

Beispiel:

```
colorbar 30, 30, "4A7DB1";
rectangle 20, 30;
rmove 20, 0;
colorbar 20, 30, "8959AB";
rectangle 20, 30;
rmove 20, 0;
colorbar 20, 30, "76A797";
rectangle 20, 30;
```



commit

Typ: SQL-Befehl

Syntax: commit

Beschreibung: Dieser Befehl markiert das Ende einer Transaktion. Eine Transaktion ist eine Abfolge von SQL-Befehlen, die als eine Einheit betrachtet werden. Eine Transaktion endet mit *commit*, *rollback* oder durch einen Verbindungsabbruch. Alle Operationen innerhalb der Transaktion werden erst wirksam, wenn sie mit *commit* bestätigt worden sind.

Transaktionsbefehle wirken sich auf alle aktiven Verbindungen aus.

siehe auch: rollback, transaction

Beispiel:

```
transaction;
select BelegNr from Rechnung where KdNr="314";
   insert into Storno set BelegNr=BelegNr;
   delete record;
next
commit;
```

connect

Typ: Funktion

Syntax: connect(*Server [, Benutzer] [, Kennwort]*)

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Mit dieser Funktion wird eine Verbindung zu einem SMTP-Server hergestellt. Nach dem Verbindungsaufbau können mit dem Befehl *sendmail* eine oder mehrere Nachrichten verschickt werden. Die Angabe von **Benutzer** und **Kennwort** ist optional und wird nur bei Providern benötigt, die mit SMTP-Authentifizierung arbeiten.

Parameter:

Server	Name des SMTP-Servers (z.B. mx.freenet.de)
Benutzer	Benutzername für SMTP-Authentifizierung
Kennwort	Kennwort für SMTP-Authentifizierung

siehe auch: disconnect, sendmail

Beispiel:

```
dim Abs as text 20;
dim AAdr as text 40;
dim EAdr as text 40;
dim Betr as text 50;
dim Nachr as text 100;

Abs = "Absender";
AAdr = "absender@freenet.de";
EAdr = "empfaenger@freenet.de";
Betr = "Grüße";
Nachr = "Viele Grüße vom DataCool Team.";

if connect("mx.freenet.de", "usr", "pwd")=true then
   if sendmail(Abs, AAdr, EAdr, Betr, Nachr)=true then
      message "Nachricht wurde versendet";
   else
      message "Fehler beim Senden der Nachricht";
   end
   disconnect;
else
   message "Verbindungsaufbau fehlgeschlagen";
end
```

copyfile

Typ: Funktion

Syntax: copyfile(*Quelldatei, Zieldatei*)

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Die Funktion kopiert die **Quelldatei** in die **Zieldatei**.

siehe auch: deletefile, renamefile

Beispiel:

```
if copyfile("c:\\test.txt", "c:\\test.bak")=true then
   message "Datei erfolgreich kopiert.";
else
   message "Fehler beim Kopieren der Datei.";
end
```

COS

Typ: Funktion

Syntax: cos(*Zahl*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion berechnet den Kosinus eines Winkels.

siehe auch: sin, tan

Beispiel:

```
cos(0)
1
```

```
cos(90)
0
```

count

Typ: Funktion

Syntax: count(*Subformular*)

Rückgabewert: Eine Ganzzahl

Beschreibung: Die Funktion gibt die Anzahl der Datensätze in dem **Subformular** zurück.

siehe auch: occurrence, sum

Beispiel:

```
count(Rechnungsposition)
```

```
Anzahl der Subsätze
```

create

Typ: Funktion

Syntax: create(*Dateiname*)

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Die Funktion öffnet die Datei für den Schreibzugriff.
Ist die Datei bereits vorhanden, so wird sie überschrieben.
Wenn die Datei noch nicht existiert, wird sie neu erstellt.

siehe auch: close, open

Beispiel:

```
if create("c:\\test.txt") = true then
   write "Hello World!\n";
   close;
end
```

createdir

Typ: Funktion

Syntax: createdir(*Verzeichnisname*)

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Die Funktion erstellt ein neues Verzeichnis. Der Rückgabewert ist *true* wenn das Verzeichnis erstellt werden konnte oder wenn das Verzeichnis bereits existiert. Andernfalls ist der Rückgabewert *false*.

siehe auch: copyfile, deletefile, renamefile

Beispiel:

```
if createdir("c:\\exportdaten") = true then
   message "Verzeichnis wurde erstellt.";
else
   message "Fehler beim Anlegen des Verzeichnisses.";
end
```

current.check

Typ: Systemvariable

Syntax: current.check

Beschreibung: DataCool prüft vor jeder Datensatz-Operation in einem Formular den Inhalt der Systemvariable *current.check*. Wird diese Variable per Script auf *false* gesetzt, so findet die Datensatz-Operation nicht statt.

In Verbindung mit den Ereignissen current.event="insert", current.event="update" oder current.event="delete", die vor der Datensatz-Operation auftreten, sowie in Verbindung mit dem Ereignis current.event=“check“ lassen sich auf diese Weise weitergehende Eingabeprüfungen per Script realisieren.

siehe auch: current.event, error

Beispiel:

```
if current.event="insert"
or current.event="update" then
   if Form.UStID=null and Form.UStTyp="EU" then
      message "Bei EU ist die USt-Nr nötig.";
      current.check=false;
      exit script;
   end
end
```

oder gleichbedeutend eine kürzere Fassung
mit dem Befehl error:

```
if current.event="insert"
or current.event="update" then
   if Form.UStID=null and Form.UStTyp="EU" then
      error "Bei EU ist die USt-Nr nötig.";
   end
end
```

current.date

Typ: Systemvariable

Syntax: current.date

Beschreibung: Die Systemvariable enthält das aktuelle Systemdatum.

Beispiel:


`current.date`


aktuelles Systemdatum (z.B. `24.12.2004`*)*

current.desktopdir

Typ: Systemvariable

Syntax: current.desktopdir

Beschreibung: Die Systemvariable gibt das Desktopverzeichnis des aktuell angemeldeten Windows-Benutzers zurück (inklusive nachfolgendem Backslash)

siehe auch: current.progdir, current.userdir

Beispiel:

```
current.desktopdir
Desktopverzeichnis
z.B. C:\Users\gordon.shumway\Desktop\
```

current.dsn

Typ: Systemvariable

Syntax: current.dsn

Beschreibung: Die Systemvariable enthält den Data Source Name (DSN) der Standard-Connection.

Beispiel:

```
current.dsn
```

```
REWE
```

current.event

Typ: Systemvariable

Syntax: current.event

Beschreibung: Die Ausführung von DataCool Scripten wird durch verschiedene Ereignisse ausgelöst. Hierbei wird zwischen benutzerdefinierten Ereignissen und Systemereignissen unterschieden.

Folgende Systemereignisse können auftreten:

current.event	Ereignis
load	Laden eines Dokumentes
check	Tritt vor dem Ausführen eines Scriptes auf. Auf diese Weise können weitergehende Prüfungen der Eingabewerte per Script vorgenommen werden. Bei eventuellen Fehlern kann das Script eine Meldung ausgeben und den Start des Scriptes unterbinden, indem die Systemvariable *current.check* auf *false* gesetzt wird.
run	Ausführen eines Scriptes
insert	Tritt **vor** dem Speichern eines neuen Datensatzes auf. Die Speicherung des Satzes kann unterbunden werden, wenn im Script die Systemvariable *current.check* auf *false* gesetzt wird.
postinsert	Tritt **nach** dem Speichern eines neuen Datensatzes auf. Die Speicherung des Satzes kann rückgängig gemacht werden, wenn im Script die Systemvariable *current.check* auf *false* gesetzt wird.
update	Tritt **vor** der Änderung eines bestehenden Datensatzes auf. Die Speicherung des Satzes kann unterbunden werden, wenn im Script die Systemvariable *current.check* auf *false* gesetzt wird.
postupdate	Tritt **nach** der Änderung eines bestehenden Datensatzes auf. Die Speicherung des Satzes kann rückgängig gemacht werden, wenn im Script die Systemvariable *current.check* auf *false* gesetzt wird.
delete	Tritt **vor** dem Löschen eines Datensatzes auf. Die Löschung des Satzes kann unterbunden werden, wenn im Script die Systemvariable *current.check* auf *false* gesetzt wird.
click_[Feldname]	Anklicken eines Auswahlbuttons
browse_[Feldname]	Auswahl eines Datensatzes mit dem SQL-Browser

Folgende benutzerdefinierte Ereignisse sind möglich:

current.event	Ereignis
[benutzerdefiniert]	Aufruf eines Menüeintrags
[benutzerdefiniert]	Anklicken eines Bildes
[benutzerdefiniert]	Anklicken eines Buttons

siehe auch: current.check, error

Beispiel:

```
if current.event="insert"
or current.event="update" then
   if Form.UStID=null and Form.UStTyp="EU" then
      error "Bei EU ist die USt-Nr nötig.", "UStID";
   end
end
```

current.file

Typ: Systemvariable

Syntax: current.file

Beschreibung: Diese Systemvariable gibt den Namen der zuletzt importierten oder exportierten Datei zurück.
Die Rückgabe des Dateinamens erfolgt inclusive Pfadangabe.

siehe auch: import, export

Beispiel:

```
import "Artikel";
copy current.file, "C:\\Importlog\\Sicherung.txt";
```

current.form

Typ: Systemvariable

Syntax: current.form

Beschreibung: Die Systemvariable enthält den Namen des aktuell geladenen Formulares. Kann in Verbindung mit current.recnum verwendet werden, wenn ein Script aus unterschiedlichen Formularen heraus aufgerufen wird.

siehe auch: current.recnum

Beispiel:

```
current.form
```

```
Formularname (z.B. Mitarbeiter)
```

current.height

Typ: Systemvariable

Syntax: current.height

Beschreibung: Die Systemvariable enthält die aktuell eingestellt Zeilenhöhe des Druckers in mm.

Beispiel:


```
current.height
```

aktuelle Zeilenhöhe in mm

current.level

Typ: Systemvariable

Syntax: current.level

Beschreibung: Die Systemvariable enthält die Berechtigungsstufe des aktuell angemeldeten Benutzers:

current.level	Wert	Bedeutung
low	1	niedrige Berechtigungsstufe
medium	2	mittlere Berechtigungsstufe
high	3	hohe Berechtigungsstufe
admin	4	Administrator

Beispiel:


```
current.level
3
```

current.page

Typ: Systemvariable

Syntax: current.page

Beschreibung: Die Systemvariable enthält die aktuelle Seitenzahl seit Beginn des letzten Startjob-Befehls.

siehe auch: current.xpos, current.ypos

Beispiel:

```
current.page
```

aktuelle Seitenzahl

current.progdir

Typ: Systemvariable

Syntax: current.progdir

Beschreibung: Die Systemvariable gibt das Programmverzeichnis von DataCool zurück (inklusive nachfolgendem Backslash)

siehe auch: current.desktopdir, current.userdir

Beispiel:

```
current.progdir
```

Programmverzeichnis (z.B. F:\DataCool\)

current.reccount

Typ: Systemvariable

Syntax: current.reccount

Beschreibung: Die Systemvariable enthält die Gesamtanzahl der Sätze, die mit dem aktiven select-Befehl im Script ausgewählt wurden.

siehe auch: current.recpos

Beispiel:

```
openwin infotext, progressbar;
select * from Kunde where Umsatz>1000;
   info current.recpos;
   progress current.reccount, current.recpos;
next
closewin;
```

current.reclock

Typ: Systemvariable

Syntax: current.reclock

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Die Systemvariable gibt Auskunft darüber, ob der aktuell angezeigte Datensatz gesperrt ist. Gesperrte Datensätze können nicht geändert werden.

siehe auch: current.recnum

Beispiel:

```
if current.event="buchen" then
   if current.reclock=true then
      message "Buchung nicht möglich";
   end
end
```

verfügbar ab Build 1107

current.recnum

Typ: Systemvariable

Syntax: current.recnum

Beschreibung: Die Systemvariable enthält die eindeutige Satznummer des aktuell im Formular angezeigten Datensatzes. Die Nummer wird im SQL-Feld *recnum* gespeichert. Mit Hilfe von *current recnum* kann ein Script bei der Datenverarbeitung auf den aktuell angezeigten Satz Bezug nehmen.

Beispiel:

```
select * from Kunde where recnum = current.recnum;
   message Name;
next
```

current.recpos

Typ: Systemvariable

Syntax: current.recpos

Beschreibung: Die Systemvariable enthält die aktuelle Position innerhalb der aktiven select-next-Schleife.

siehe auch: current.reccount

Beispiel:

```
openwin infotext, progressbar;
select * from Kunde where Umsatz>1000;
   info current.recpos;
   progress current.reccount, current.recpos;
next
closewin;
```

current.station

Typ: Systemvariable

Syntax: current.station

Beschreibung: Mit dieser Systemvariable kann der Computername der Arbeitsstation bestimmt werden.

Beispiel:

```
current.station
```

aktuelle Arbeitsstation (z.B. `"PC01"`*)*

current.tempdir

Typ: Systemvariable

Syntax: current.tempdir

Beschreibung: Mit dieser Systemvariable kann der Pfad für temporäre Dateien bestimmt werden.

Beispiel:

```
current.tempdir
C:\Windows\Temp\
```

current.time

Typ: Systemvariable

Syntax: current.time

Beschreibung: Die Systemvariable enthält die aktuelle Systemzeit.

Beispiel:

```
current.time
```

aktuelle Systemzeit (z.B. `08:15`*)*

current.user

Typ: Systemvariable

Syntax: current.user

Beschreibung: Die Systemvariable enthält den Namen des aktuell angemeldeten Benutzers.

Beispiel:


`current.user`


aktueller Benutzer (z.B. `"spiderman"`*)*

current.userdir

Typ: Systemvariable

Syntax: current.userdir

Beschreibung: Die Systemvariable gibt das Benutzerverzeichnis des aktuell angemeldeten Windows-Benutzers zurück (inklusive nachfolgendem Backslash).

Beispiel:

```
current.userdir
```

```
Benutzerverzeichnis
z.B. C:\Users\gordon.shumway\Documents\
```

current.xpos

Typ: Systemvariable

Syntax: current.xpos

Beschreibung: Die Systemvariable enthält die aktuelle Cursorposition des Drucker in x-Richtung. Die Angabe erfolgt in mm und beginnt ab der Position des linken Randes.

siehe auch: current.ypos, current.page

Beispiel:

```
current.xpos
```

aktuelle x-Position in mm

current.ypos

Typ: Systemvariable

Syntax: current.ypos

Beschreibung: Die Systemvariable enthält die aktuelle Cursorposition des Druckers in y-Richtung. Die Angabe erfolgt in mm und beginnt ab der Position des oberen Randes.

siehe auch: current.xpos, current.page

Beispiel:

```
current.ypos
```

aktuelle y-Position in mm

date

Typ: Funktion

Syntax: date(*Tag, Monat, Jahr*)

Rückgabewert: Ein Datum

Beschreibung: Die Funktion bildet aus **Tag**, **Monat** und **Jahr** einen Datumswert. Wird das Jahr 2-stellig angegeben, so wird die 4-stellige Jahreszahl unter Verwendung des Startjahres in der Konfiguration berechnet.

Beispiel:

```
date(24, 12, 2004)
```

```
24.12.2004
```

datediff

Typ: Funktion

Syntax: datediff(*Start, Ende*)

Rückgabewert: Eine Zahl

Beschreibung: Gibt eine Zahl zurück, die die Anzahl der Tage zwischen zwei Datumswerten (**Start** und **Ende**) angibt.

Beispiel:

```
datediff("10.01.2006", "15.01.2006")
```

```
5
```

day

Typ: Funktion

Syntax: day(*Datum*)

Rückgabewert: Eine Ganzzahl

Beschreibung: Die Funktion bestimmt den Tag aus einem **Datum**.

siehe auch: month, year

Beispiel:

```
day(06.12.2004)
6
```

decrypt

Typ: Funktion

Syntax: decrypt(*Zeichenfolge*)

Rückgabewert: Eine Zeichenenfolge

Beschreibung: Die Funktion entschlüsselt die übergebene Zeichenfolge.
Hierzu wird die mit Base 64 codierte Zeichenfolge in Binärdaten zurückgewandelt und anschließend wieder entschlüsselt (RC4 128 Bit).

Mit Hilfe der Funktionen 'encrypt' und 'decrypt' kann ein Script Daten in verschlüsselten Formularfeldern lesen und schreiben.

siehe auch: encrypt

Beispiel:

```
decrypt("LJgbOJatQqQ=")
```

```
DataCool
```

delete

Typ: SQL-Befehl

Syntax: **delete from** Tabelle [**where** Bedingung] ;

oder

delete record;

Beschreibung: Mit dem Befehl **delete** werden alle selektierten Datensätze einer Tabelle gelöscht. Der Befehl **delete record** löscht nur den mit *select* ausgewählten Datensatz.

siehe auch: insert, update

Beispiel:

```
delete from Kunde where Umsatz = 0;
```

```
select * from Kunde where Umsatz = 0;
   delete record;
next
```

deletefile

Typ: Funktion

Syntax: deletefile(*Dateiname*)

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Die Funktion löscht die angegebene Datei.
Der **Dateiname** muss eine Pfadangabe enthalten.

siehe auch: copyfile, renamefile

Beispiel:

```
if deletefile("c:\\test.txt") = true then
   message "Datei wurde gelöscht.";
else
   message "Fehler beim Löschen der Datei.";
end
```

dim

Typ: Befehl

Syntax: dim *Variable* as *Typ*

Beschreibung: Mit der Anweisung *dim* wird eine **lokale** Variable definiert. Lokale Variablen sind nur innerhalb des Scriptes verfügbar, in dem sie deklariert wurden.

Nach dem Schlüsselwort *as* werden Datentyp und Format festgelegt. DataCool unterstützt eindimensionale Datenfelder. Zur Definition eines Datenfeldes müssen Klammern nach dem Variablennamen gesetzt werden

Typ	Beschreibung	Wertebereich
text n	Text mit der Länge n	n = 1-2048
numeric m	Nummer mit führenden Nullen (Länge m)	m = 1-9
number m	Gleitkommazahl mit Länge m	m = 1-9
number m.0	Ganzzzahl mit m Stellen	m = 1-9
number m.n	Festkommazahl mit m Vorkomma- und n Nachkommastellen	m = 1-9; n = 1-9
date	Datum (TT.MM.JJJJ)	
time	Uhrzeit (HH:MM)	
boolean	*true* oder *false* bzw. *ja* oder *nein*	

siehe auch: public

Beispiel:

```
dim Name as text 20;
dim Nummer as numeric 4;
dim Gleitkomma as number 5;
dim Ganzzahl as number 5.0;
dim Festkomma as number 5.2;
dim Datum as date;
dim Uhrzeit as time;
dim Richtig as boolean;
dim Zahlfeld() as number;
```

disconnect

Typ: Befehl

Syntax: disconnect

Beschreibung: Mit diesem Befehl wird die Verbindung zum SMTP-Server getrennt.

siehe auch: connect, sendmail

Beispiel:

```
dim Abs as text 20;
dim AAdr as text 40;
dim EAdr as text 40;
dim Betr as text 50;
dim Nachr as text 100;

Abs = "Absender";
AAdr = "absender@freenet.de";
EAdr = "empfaenger@freenet.de";
Betr = "Grüße";
Nachr = "Viele Grüße vom DataCool Team.";

if connect("mx.freenet.de", "usr", "pwd")=true then
   if sendmail(Abs, AAdr, EAdr, Betr, Nachr)=true then
      message "Nachricht wurde versendet";
   else
      message "Fehler beim Senden der Nachricht";
   end
   disconnect;
else
   message "Verbindungsaufbau fehlgeschlagen";
end
```

dos2win

Typ: Funktion

Syntax: dos2win(*Zeichenfolge*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion wandelt den Zeichensatz von DOS nach Windows.

siehe auch: utf2win, win2dos, win2utf

Beispiel:

```
dos2win("šbel, bel sprach der Dbel und
verschwand in der Wand.")
```

```
Übel, übel sprach der Dübel und
verschwand in der Wand.
```

encrypt

Typ: Funktion

Syntax: encrypt(*Zeichenfolge*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion verschlüsselt die übergebene Zeichenfolge (RC4 128 Bit) und codiert das Ergebnis mit dem Base 64 Verfahren.

Mit Hilfe der Funktionen 'encrypt' und 'decrypt' kann ein Script Daten in verschlüsselten Formularfeldern lesen und schreiben.

siehe auch: decrypt

Beispiel:

```
encrypt("DataCool")
```

```
LJgbOJatQqQ=
```

endjob

Typ: Druckbefehl

Syntax: endjob

Beschreibung: Der Befehl *endjob* beendet einen Druckauftrag.

siehe auch: startjob

Beispiel:

```
startjob printer;
print "Hello World!";
endjob;
```

eof

Typ: Funktion

Syntax: eof()

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Die Funktion liefert als Ergebnis den Wert *true*, wenn das Ende der mit *open* geöffneten Datei erreicht ist. Andernfalls ist der Rückgabewert *false*.

siehe auch: open, read

Beispiel:

```
if open("c:\\test.txt") = true then
   while eof() = false;
      message read();
   wend
   close;
end
```

error

Typ: Befehl

Syntax: error *Infotext [, Feldname]*

Beschreibung: Dieser Befehl gibt eine Fehlermeldung mit **Infotext** aus. Anschließend wird *current.check=false* gesetzt um das Speichern oder Löschen des angezeigten Satzes zu unterbinden. Mit dem optionalen Paramter *Feldname* kann der Focus in ein Eingabefeld aus dem Formular gesetzt werden. Zusätzlich wird die Ausführung des Scriptes beendet.

siehe auch: current.check, current.event

Beispiel:

```
if current.event="insert"
or current.event="update" then
   if Form.UStID=null and Form.UStTyp="EU" then
      error "Bei EU ist die USt-Nr nötig.";
   end
end
```

oder gleichbedeutend der Ablauf ohne den Befehl error:

```
if current.event="insert"
or current.event="update" then
   if Form.UStID=null and Form.UStTyp="EU" then
      message "Bei EU ist die USt-Nr nötig.";
      current.check=false;
      exit script;
   end
end
```

exit

Typ: Befehl

Syntax: exit ...

Beschreibung: Die Anweisung *exit* wird verwendet, um die Ausführung der Befehle *select*, *repeat* oder *while* sofort zu beenden. Mit *exit script* wird das Script beendet.

Anweisung	Beschreibung
exit select	Ausführung fortsetzen nach *next*
exit repeat	Ausführung fortsetzen nach *until*
exit while	Ausführung fortsetzen nach *wend*
exit script	Script beenden
exit document	Script beenden und Dokument schliessen

siehe auch: select, repeat, while

Beispiel:

```
if current.level < middle then
   message "Sie sind nicht berechtigt!";
   exit script;
end
```

export

Typ: Befehl

Syntax: export *Name [, Pfad]*

Beschreibung: Der Befehl startet einen Datenexport gemäß der Exportdefinition **Name**.
Die optionale Angabe **Pfad** dient zur Festlegung der Zieldatei.
Fehlt diese Angabe, so wird der Dateiname laut Exportdefinition verwendet.

siehe auch: current.file, import

Beispiel:

```
export "Artikeldaten";
export "Artikeldaten", "c:\Artikel.xml";
```

Farbangaben (Konstanten)

Typ: Konstante

Werte:

Name	Wert	Bedeutung
black	000000	schwarz
blue	0000FF	blau
green	00FF00	grün
red	FF0000	rot

fetch

Typ: Funktion

Syntax: fetch("select *Feld* from *Tabelle* [where *Auswahlbedingung]*")

Rückgabewert: Eine Zeichenfolge

Beschreibung: Selektiert **Feld** von **Tabelle** unter Berücksichtigung von **Bedingung**. Zurückgegeben wird der Feldinhalt des ersten passenden Datensatzes. In der Auswahlbedingung werden Variablen mit vorangestelltem "@"-Zeichen ersetzt:
Normale Variablen, Systemvariablen (z.B. @current.date), Eingabefelder (z.B. @Form.Feld) und Felder aus umgebenden select-Schleifen (z.B. @Tabelle.Feld). Das Schlüsselwort "select" kann entfallen.

Beispiel:

```
fetch("Firmenname from Setup")
```

```
Neuhahn GmbH
```

```
fetch("max(BelegNr) from Beleg " &
      "where Belegart='Rechnung'")
```

```
004711
```

fill

Typ: Funktion

Syntax: fill(*Zeichenfolge, n*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion gibt eine Zeichenfolge mit der Gesamtlänge **n** zurück. Hierzu werden am Ende der **Zeichfolge** ggf. Leerzeichen angehängt.

siehe auch: trim

Beispiel:

```
fill("DataCool", 9) & "ist cool!"
```

```
DataCool ist cool!
```

first

Typ: Funktion

Syntax: first(*Zeichenfolge, n*)

Rückgabewert: Eine Zeichenfolge mit der Länge n

Beschreibung: Die Funktion gibt die ersten **n** Zeichen der **Zeichenfolge** zurück.

siehe auch: firstword, mid, last, lastword

Beispiel:

```
first("DataCool", 4)
```

```
Data
```

firstword

Typ: Funktion

Syntax: firstword(*Zeichenfolge [, Trennzeichen]*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion gibt den Anfang der **Zeichenfolge** bis zum ersten Auftreten eines Leerzeichens zurück. Mit dem optionalen Parameter **Trennzeichen** kann ein alternatives Abgrenzungszeichen angegeben werden.

siehe auch: first, mid, last, lastword

Beispiel:

```
firstword("DataCool ist super!")
DataCool

firstword("Willkommen|in|der|Zukunft", "|")
Willkommen
```

floor

Typ: Funktion

Syntax: floor(*Zahl, n*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion rundet die **Zahl** auf **n** Nachkommastellen ab.

siehe auch: ceil, formatnumber, labvalue, round

Beispiel:

```
floor(10.3, 0)
10
```

```
floor(10.468, 2)
10.46
```

```
floor(-10.4, 0)
-11
```

focus

Typ: Befehl

Syntax: focus *Feldname*

Beschreibung: Mit diesem Befehl wird der Cursor auf das Feld **Feldname** gesetzt.

Beispiel:

```
if form.KdNr=null then
   message "Bitte geben Sie eine Kundennummer ein.";
   current.check=false;
   focus "KdNr";
   exit script;
end
```

font

Typ: Druckbefehl

Syntax: font *Schriftart, Eigenschaften, Schriftgröße [, Schriftfarbe]*

Beschreibung: Die ausgewälte Schriftart muß in Windows installiert sein. Falls die Schriftart nicht vorhanden ist, wird eine ähnliche Schrift verwendet. Als Standardschrift wird *Arial 10 Punkt* verwendet. Folgende Schrifteigenschaften können ausgewählt und miteinander kombiniert werden:

b	fett drucken
i	kursiv drucken
u	unterstreichen

Der optionale Parameter Schriftfarbe verändert die Farbe der eingestellten Schrift. Die Farbangabe erfolgt entweder über eine der nachfolgenden Farbkonstanten oder einen RGB-Wert. Standardmäßig ist die Schriftfarbe schwarz.

Schriftfarbe	Wert	Bedeutung
black	000000	schwarz
red	FF0000	rot
green	00FF00	grün
blue	0000FF	blau

Beispiel:

```
font "ARIAL", "bu", 12;
print "Hello World!";
```

Hello World!

```
font "ARIAL", "", 12, blue;
print "Hello World!";
```

Hello World!

```
font "ARIAL", "", 12, "FF0000";
print "Hello World!";
```

Hello World!

formatdate

Typ: Funktion

Syntax: formatdate(*Datum, Formatangabe [, Sprache]*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion gibt eine Zeichenfolge zurück, die das **Datum** gemäß der **Formatangabe** darstellt. Mit dem optionalen Parameter **Sprache** kann die Schreibweise des Monats beeinflusst werden.

Formatangabe:

Formatangabe	Bedeutung
T	Tag (1-31)
TT	Tag (01-31)
M	Monat (1-12)
MM	Monat (01-12)
MMM	Sprache=1 oder Sprache="D": JAN, FEB, MÄR, APR, MAI, JUN, JUL, AUG, SEP, OKT, NOV, DEZ Sprache=2 oder Sprache="E" (default): JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC
JJ	Jahreszahl, 2-stellig
JJJJ	Jahreszahl, 4-stellig

Beispiel:

```
formatdate(02.05.2004, "TT-MMM-JJJJ")
02-MAY-2004

formatdate(02.05.2004, "MM/TT/JJJJ")
05/02/2004
```

formatnumber

Typ: Funktion

Syntax: formatnumber(*Zahl, n*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion rundet die **Zahl** auf **n** Nachkommastellen.
Der Rückgabewert ist eine formatierte Zeichenfolge mit Tausendertrennern.

siehe auch: ceil, floor, labvalue, round

Beispiel:

```
formatnumber(1000.507, 2)
```

```
1.000,51
```

getdir

Typ: Funktion

Syntax: getdir(*Verzeichnisname*)

Rückgabewert: Ein Array

Beschreibung: Die Funktion getdir wird benutzt, um ein Verzeichnis auszulesen und die somit ermittelten Dateinamen in einem Datenfeld (array) zu speichern.

Beispiel:

```
dim i as number 2.0;
dim Files() as text 50;

Files=getdir("C:\\Temp");

i=1;
while Files(i)>null;
   message Files(i);
   i=i+1;
wend
```

gethtml

Typ: Funktion

Syntax: gethtml(*url*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Mit der Funktion *gethtml* kann der Quelltext einer Internetseite mit der Adresse **url** abgerufen werden. Hierdurch können aktuelle Informationen wie beispielsweise Devisenkurse aus dem Internet übernommen werden. Ist die Internetseite nicht verfügbar, so hat die Zeichenfolge die Länge 0.

Beispiel:

```
gethtml("http://spk.gedif.de/159/kurse_
          crossrates.htm?u=0&p=0&k=0")
```

Auszug aus dem HTML-Quelltext:

```
<td id="bg04"> AUD </td>
<td id="bg02" align="right">-</td>
<td id="bg02" align="right">0,9372</td>
<td id="bg02" align="right">5,5617</td>
<td id="bg02" align="right">78,8735</td>
<td id="bg02" align="right">0,8927</td>
...
```

getsoap

Typ: Funktion

Syntax: getsoap(*url, Request, ContentType [, Action] [, Stylesheet]*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: SOAP (Simple Object Access Protocol) ist ein Netzwerkprotokoll, mit dessen Hilfe Daten via Internet zwischen Systemen ausgetauscht werden können. Hierzu wird eine Anfrage in einen sog. SOAP-Envelope verpackt und mit dem Parameter **request** an den Server mit der **url** übermittelt. Der Parameter **ContentType** enthält Angaben über die Codierung der Anfrage.

Mit dem optionalen Parameter **Action** kann zwischen unterschiedlichen Diensten des Servers unterschieden werden.

Die Antwort des Servers ist eine Zeichenkette im XML-Format.

Auf Wunsch können Sie ein XSLT-Stylesheet im Funktionsaufruf übergeben, mit dessen Hilfe die XML-Antwort in HTML-Quelltext gewandelt wird.

Beispiel:

```
getsoap(url, request, "text/xml; charset='utf-8'",
"Biblio", "c:\epoline.xslt")
```

getrest

Typ: Funktion

Syntax: getrest(*Methode, URL, ContentType [, Parameter] [, Stylesheet] [, Authorization]*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: REST (Representational State Transfer) ist eine Übertragungsmethode, mit dessen Hilfe Daten via Internet zwischen Systemen ausgetauscht werden können. Hierzu wird eine **URL** mit Hilfe der gewählten **Methode** ("POST" oder "GET") aufgerufen. Der **ContentType** enthält Angaben über die Codierung der Anfrage. Der optionale **Parameter** dient zur Übergabe weiterer Informationen an den Server. Mit der optionalen **XSLT-Stylesheet**-Datei kann die XML-Antwort vor der Rückgabe aufbereitet werden. Der optionale Parameter **Authorization** dient zur Übergabe von Login-Informationen oder Kennwörtern. Die Antwort des Servers ist eine Zeichenkette im XML-Format.

Beispiel:

```
getrest("GET", url,
"text/xml; charset='utf-8'", "", "test.xslt");
```

getval

Typ: Funktion

Syntax: getval()

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion gibt die in der Zwischenablage gespeicherte **Zeichenfolge** zurück.

siehe auch: putval

Beispiel:

```
getval()
```

Inhalt der Zwischenablage

h2time

Typ: Funktion

Syntax: h2time(*Stunden, Format*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion wandelt die übergebenen **Stunden** (Dezimalwert) in ein Zeitliteral um. Bei der Ausgabe kommt das **Format** zur Anwendung.

siehe auch: m2time, time2h, time2m

Beispiel:

```
h2time(3.5, "HH:MM")
03:30

h2time(1.25, "H:MM")
1:15

h2time(-4.75, "HHH:MM")
-004:45
```

height

Typ: Druckbefehl

Syntax: height *Zeilenhöhe*

Beschreibung: Dieser Befehl legt die aktuelle **Zeilenhöhe** in mm fest. Der Befehl *newline(n)* bewirkt eine Bewegung des Cursors um den Betrag n * Zeilenhöhe.

siehe auch: newline

Beispiel:

```
height 5;
```

hline

Typ: Druckbefehl

Syntax: hline *Länge [, Breite]*

Beschreibung: Zeichnet eine horizontale Linie mit den angegebenen Werten für Länge und Breite. Ausgangspunkt der Linie ist die aktuelle Cursorposition. Die Cursorposition bleibt unverändert. Die Angabe der Parameter erfolgt in der Einheit mm.

siehe auch: line, vline

Beispiel:

```
hline 100, 0.3;
```

hours

Typ: Funktion

Syntax: hours(*Uhrzeit*)

Rückgabewert: Eine Ganzzahl

Beschreibung: Die Funktion bestimmt den Stundenwert (0-23) aus einer **Uhrzeit**.

siehe auch: minutes

Beispiel:


```
hours(08:15)
8
```

if-Anweisung

Typ: Befehl

Syntax:

```
if Bedingung then
   Anweisung1;
else
   Anweisung2;
end
```

Beschreibung: Die if-Anweisung führt eine von zwei verschiedenen Anweisungen (oder Befehlsfolgen) in Abhängigkeit davon aus, ob die **Bedingung** zutrifft oder nicht. Das Schlüsselwort *else* kann entfallen.
In diesem Fall werden die Anweisungen zwischen *if* und *end* nur dann ausgeführt, wenn die Bedingung erfüllt ist.

siehe auch: case, if-Funktion

Beispiel:

```
if current.level < high then
   message "Sie sind nicht berechtigt!";
   exit script;
else
   delete from Buchung;
   message "Buchungen wurden gelöscht.";
end
```

if-Funktion

Typ: Funktion

Syntax: if(*Bedingung, Wert1, Wert2*)

Beschreibung: Trifft die Bedingung zu, so ist das Ergebnis der Funktion **Wert1**, andernfalls **Wert2**.

siehe auch: case, if-Anweisung

Beispiel:

```
if(current.date>="01.01.2000", "21", "20") &
". Jahrhundert"
```

```
21. Jahrhundert
```

import

Typ: Befehl

Syntax: import *Name [, Pfad]*

Beschreibung: Der Befehl startet einen Import gemäß der Importdefinition **Name**.
Die optionale Angabe **Pfad** dient zur Festlegung der Quelldatei.
Fehlt diese Angabe, so wird der Dateiname laut Importdefinition verwendet.

siehe auch: current.file, export

Beispiel:

```
import "Artikeldaten";
import "Artikeldaten", "c:\Artikel.xml";
```

info

Typ: Befehl

Syntax: info *Zeichenfolge*

Beschreibung: Dieser Befehl setzt den Infotext in einem mit *openwin* geöffneten Fenster.

siehe auch: openwin

Beispiel:

```
openwin infotext, progressbar;
select * from Kunde where Umsatz>1000;
   info current.recpos;
   progress current.reccount, current.recpos;
next
closewin;
```

Infofenster (Konstanten)

Typ: Konstante

Werte:

Name	Wert	Bedeutung
infotext	1	Textzeile, die mit *info* verändert werden kann
progressbar	2	Forschrittsbalken, der mit *progress* verändert werden kann
cancel	3	Button für Scriptabbruch

input

Typ: Funktion

Syntax: input(*Fragetext, Länge, Eingabezwang)*

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion öffnet einen Dialog mit dem **Fragetext** und einer Textbox.
Die max. Eingabelänge für den Text kann mit **Länge** begrenzt werden.
Eingabezwang legt fest ob der Dialog ohne Eingabe verlassen werden kann.
Zurückgegeben wird die Eingabe der Textbox.

Beispiel:

```
input("Stornogrund:", 25, true)
```

inputbulk

Typ: Befehl

Syntax: inputbulk

Beschreibung: Der Befehl öffnet einen Dialog zur automatischen Datenerfassung.

Übernommene Einträge werden in die Tabelle SYS_Input gespeichert, welche vor jedem Ausführen des Befehls geleert wird.

Beispiel:

```
inputbulk;

   select * from SYS_Input
   where Station=@current.station;
      message value;
   next
```

insert

Typ: SQL-Befehl

Syntax: **insert into** Tabelle **set** Feld1=Wert1, Feld2=Wert2, ..., Feldn=Wertn;

Beschreibung: Mit dem Befehl **insert** wird ein neuer Datensatz in eine Tabelle eingefügt. Nach dem Schlüsselwort **set** werden alle Felder mit den zugehörigen Werten belegt.

siehe auch: delete, update

Beispiel:

```
insert into Kunde set
    Nummer = 0001,
    Name = "Alf",
    Ort = "Melmac";
```

labvalue

Typ: Funktion

Syntax: labvalue(*Zahl, n*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion rundet die **Zahl** auf **n** signifikante Stellen.

siehe auch: ceil, floor, formatnumber, round

Beispiel:

```
labvalue(0.0567, 3)
0.057

labvalue(1.764, 3)
1.76

labvalue(102.5, 3)
103

labvalue(1025, 3)
1030
```

last

Typ: Funktion

Syntax: last(*Zeichenfolge, n*)

Rückgabewert: Eine Zeichenfolge mit der Länge n

Beschreibung: Die Funktion gibt die letzten **n** Zeichen der **Zeichenfolge** zurück.

siehe auch: first, firstword, mid, lastword

Beispiel:

```
last("DataCool", 4)
```

```
Cool
```

lastday

Typ: Funktion

Syntax: lastday(*Datum*)

Rückgabewert: Ein Datum

Beschreibung: Die Funktion verschiebt ein **Datum** auf den letzten Tag des Monats.

Beispiel:


```
lastday(24.12.2004)
31.12.2004
```

lastword

Typ: Funktion

Syntax: lastword(*Zeichenfolge [, Trennzeichen]*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion gibt das Ende der **Zeichenfolge** ab dem letzten Auftreten eines Leerzeichens zurück. Mit dem optionalen Parameter **Trennzeichen** kann ein alternatives Abgrenzungszeichen angegeben werden.

siehe auch: first, firstword, mid, last

Beispiel:

```
lastword("DataCool ist super!")
```

```
super!
```

```
lastword("Willkommen|in|der|Zukunft", "|")
```

```
Zukunft
```

len

Typ: Funktion

Syntax: len(*Zeichenfolge*)

Rückgabewert: Eine Ganzzahl

Beschreibung: Die Funktion gibt die Anzahl der Zeichen einer **Zeichenfolge** zurück.

Beispiel:

```
len("DataCool")
```

```
8
```

line

Typ: Druckbefehl

Syntax: line *x, y [, Breite]*

Beschreibung: Zeichnet ausgehend von der aktuellen Cursorposition eine Linie bis zur angegebenen Koordinate (**x**, **y**). Der Cursor wird anschließend auf die neue Koordinate (**x**, **y**) gesetzt. Dadurch wird die Erstellung von Liniendiagrammen möglich. Die Angabe der Koordinaten und der Breite erfolgt in mm.

siehe auch: hline, vline

Beispiel:

```
line 10, 20, 0.3;
```

log

Typ: Funktion

Syntax: log(*Zahl*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion berechnet den 10er-Logarithmus der **Zahl**.

siehe auch: power, sqrt

Beispiel:

```
log(100)
2
```

lower

Typ: Funktion

Syntax: lower(*Zeichenfolge*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion wandelt alle Buchstaben der **Zeichfolge** in Kleinbuchstaben.

siehe auch: normtext, replace, upper

Beispiel:

```
lower("Structured Query Language")
```

```
structured query language
```

m2time

Typ: Funktion

Syntax: m2time(*Minuten, Format*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion wandelt die übergebenen **Minuten** in ein Zeitliteral um. Bei der Ausgabe kommt das **Format** zur Anwendung.

siehe auch: h2time, time2h, time2m

Beispiel:

```
m2time(30, "HH:MM")
00:30

m2time(210, "H:MM")
3:30

m2time(-15,"HHH:MM")
-000:15
```

md5

Typ: Funktion

Syntax: md5(*Zeichenfolge*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion bildet für die übergebene einen md5-Hashwert.
Ein Hashwert ist eine Zusammenfassung oder Prüfsumme einer Zeichenfolge bzw. eines digitalen Dokumentes und wird mit Hilfe von mathematischen Algorithmen errechnet.

Beispiel:

```
md5("DataCool ist cool!")
```

```
d9d84c03a8e3d41c32436fa5310d7eea
```

memosize

Typ: Funktion

Syntax: memosize(*Wert*)

Rückgabewert: Eine Ganzzahl

Beschreibung: Die Funktion bestimmt die Anzahl der Zeilen, die zur Ausgabe der Zeichenfolge **Wert** benötigt werden. Hierbei wird die Spaltenbreite zugrundegelegt, die sich aus der Anzahl der Zeichen in der Felddefinition ergibt.

Beispiel:

```
memosize("Zeile1 \n Zeile2 \n Zeile3")
```

```
3
```

memolines

Typ: Funktion

Syntax: memolines(*Wert*)

Rückgabewert: Ein Array

Beschreibung: Die Funktion gibt alle Zeilen eines Memofeldes als Array zurück. Bei der Formatierung wird die Spaltenbreite zugrundegelegt, die sich aus der Anzahl der Zeichen in der Felddefinition ergibt.

Beispiel:

```
dim i as number 5.0;
dim Anzahl as number 5.0;
dim Zeile() as text 100;

select * from Kunde where KdNr="4711";

   Zeile=memolines(Bemerkung);
   Anzahl=memosize(Bemerkung);

   startjob screen;
   i=0;
    while i<Anzahl;
      i=i+1;
      print Zeile(i), left, 0;
      newline;
   wend
   endjob;

next
```

verfügbar ab Build 1162 (10/2018)

message

Typ: Befehl

Syntax: message *Infotext*

Beschreibung: Mit diesem Befehl wird der **Infotext** in einer Messagebox angezeigt. Der Programmablauf wird erst nach Bestätigung der angezeigten Meldung fortgesetzt.

siehe auch: query

Beispiel:

```
message "Rechnung wurde storniert.";
```

mid

Typ: Funktion

Syntax: mid(*Zeichenfolge, p[, n]*)

Rückgabewert: Eine Zeichenfolge mit der Länge n

Beschreibung: Die Funktion entnimmt **n** Zeichen der **Zeichenfolge** beginnend ab der Position **p**. Die Angabe der Länge **n** kann entfallen. In diesem Falle werden alle Zeichen ab der Position **p** zurückgegeben.

siehe auch: first, firstword, last, lastword

Beispiel:

```
mid("Structured Query Language", 12, 5)
Query

mid("Structured Query Language", 12)
Query Language
```

minutes

Typ: Funktion

Syntax: minutes(*Uhrzeit*)

Rückgabewert: Eine Ganzzahl

Beschreibung: Die Funktion bestimmt den Minutenwert (0-59) aus einer **Uhrzeit**.

siehe auch: hours

Beispiel:


```
minutes(08:15)
15
```

mod

Typ: Funktion

Syntax: mod(*Zahl1, Zahl2*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion bestimmt den Rest der Division von **Zahl1** durch **Zahl2**.

siehe auch: ceil, floor, round

Beispiel:

```
mod(8, 3)
```

```
2
```

month

Typ: Funktion

Syntax: month(*Datum*)

Rückgabewert: Eine Ganzzahl

Beschreibung: Die Funktion bestimmt den Monat aus einem **Datum**.

siehe auch: day, year

Beispiel:

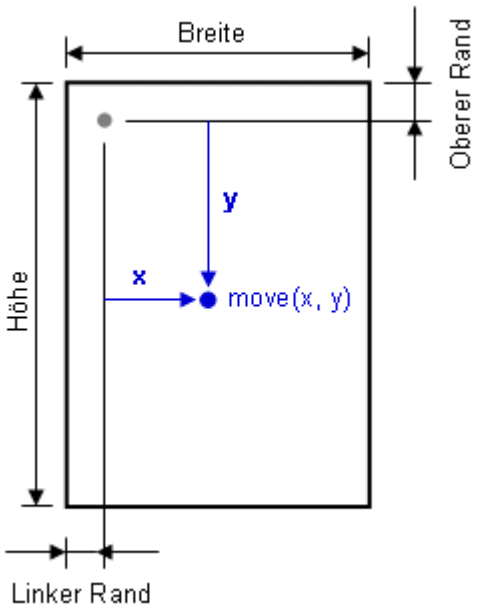

```
month(06.01.2004)
1
```

move

Typ: Druckbefehl

Syntax: move *x, y*

Beschreibung: Bewegt den Cursor auf die angegebene Koordinate (**x**, **y**).
Der Koordinatenursprung ist die linke obere Ecke unter Berücksichtigung der Randeinstellungen. Die Angaben erfolgen in der Maßeinheit mm.



siehe auch: rmove, tab

Beispiel:

```
move 20, 50;
```

newline

Typ: Druckbefehl

Syntax: newline *[n]*

Beschreibung: Bewegt den Cursor **n** Zeilen nach unten an den linken Rand.
Fehlt die Angabe **n**, so wird um **eine** Zeile weitergeschaltet.
Der Zeilenabstand kann mit dem Befehl *height* eingestellt werden.

siehe auch: height, move

Beispiel:

```
newline;
newline 2;
```

newpage

Typ: Druckbefehl

Syntax: newpage

Beschreibung: Durch diesen Befehl wird ein Seitenvorschub auf dem Drucker ausgeführt. Anschließend wird der Cursor auf den linken oberen Rand der neuen Seite gesetzt.

Beispiel:

```
newpage;
```

noexist

Typ: Konstante

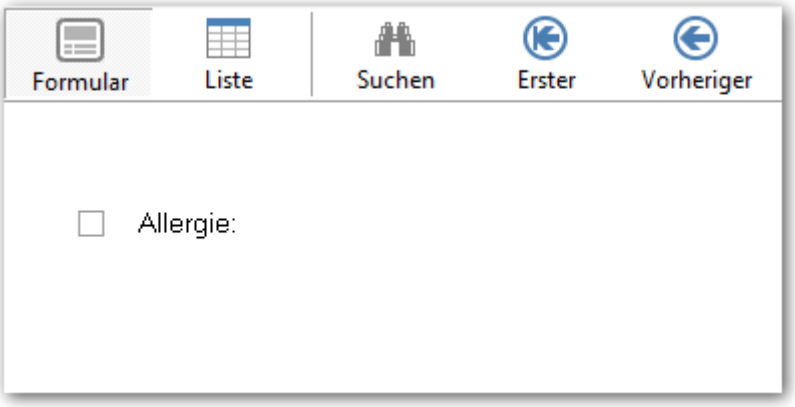
Syntax: noexist

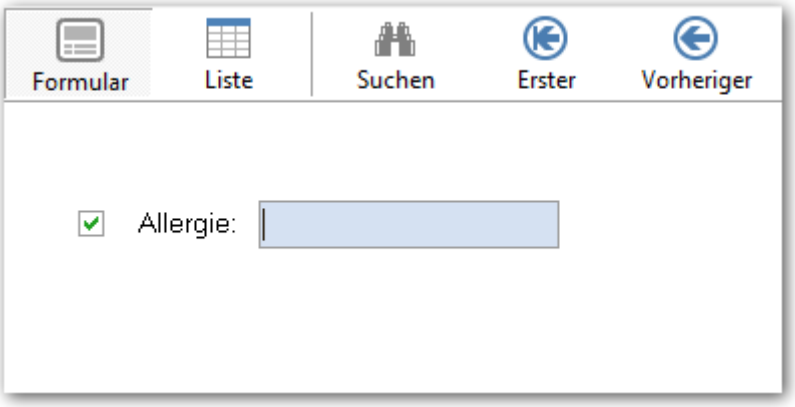
Beschreibung: Wenn die Ableitungsformel eines Eingabefeldes den Wert *noexist* ergibt, dann wird das betroffene Feld ausgeblendet. Auf diese Weise können Formulare erstellt werden, die sich in Abhängigkeit der Benutzereingaben dynamisch verändern.

Beispiel:

```
if(Allergie="ja", null, noexist)
```

Die obige Ableitungsformel wird für das Feld Allergiename verwendet und bewirkt, dass die Eingabe eines Allergienamens nur dann abgefragt wird, wenn das Feld "Allergie" angekreuzt ist:





normtext

Typ: Funktion

Syntax: normtext(*Zeichenfolge [, Filter]*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion erzeugt eine normierte Zeichfolge, die als standardisiertes Suchkriterium verwendet werden kann. Der normierte Text wird auf die Zeichen 0-9 und A-Z reduziert.
Hierbei werden alle Buchstaben in Großbuchstaben gewandelt.

Mit Hilfe des optionalen Parameters **Filter** kann alternativ festgelegt werden, welche Zeichen von der Funktion zurückgegeben werden. In diesem Fall werden alle Zeichen entfernt, die nicht in der Filterzeichenfolge enthalten sind. Hierbei werden ebenfalls alle Buchstaben in Großbuchstaben gewandelt.

siehe auch: lcase, replace, ucase

Beispiel:


```
normtext("4711/2004 de")
47112004DE
```


```
normtext("4711/2004 de", "0123456789")
47112004
```

occurence

Typ: Funktion

Syntax: occurrence(*Subformular*, *Feldname*, *Wert*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion zählt, wie oft ein bestimmter Wert in einer Subformularspalte auftritt.

siehe auch: count, sum

Beispiel:

```
occurrence(Dokument, Ausgabeziel, "E-Mail")
```

```
2
```

open

Typ: Funktion

Syntax: open(*Dateiname*)

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Die Funktion öffnet die Datei für den Lesezugriff.

siehe auch: close, create

Beispiel:

```
if open("c:\\test.txt") = true then
   message read();
   close;
end
```

opencom

Typ: Funktion

Syntax: opencom (*Port, Parameter, Fenster*)

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Diese Funktion öffnet den COM-Port des PC und initialisiert die serielle Schnittstelle mit den benötigten Übertragungsparametern. Der mit Komma-getrennte Parameterstring gibt folgendes an:

„Übertragungsrate,Parität,Datenbits,Stoppbits“

Übertragungsrate	Übertragungsgeschwindigkeit (Bits pro Sekunde).
Parität	Bestimmt, ob bei der Übertragung ein Paritätsbit zur Fehlerkorrektur angehängt werden soll (J/N).
Datenbits	Anzahl der Datenbits eines Datenwortes.
Stoppbits	Anzahl der Stoppbits am Ende eines Datenwortes.

Fenster gibt an ob ein Statusfenster geöffnet werden soll.

siehe auch: closecom, readcom, writecom

Beispiel:

```
dim Num as text 10;

if opencom(1, "9600,N,8,1", true)=false then
   message "Fehler beim Initialisieren von COM1.";
   exit script;
end

truncate table BarcodeInventur;

repeat
   Num=readcom(chr(13));
   if Num<>noexist then
      insert into BarcodeInventur set Nummer=Num;
   end
until Num=noexist;

closecom;
```

opendoc

Typ: Befehl

Syntax: opendoc *Dateiname*

Beschreibung: Der Befehl *opendoc* öffnet die angegebene Datei mit der vom Betriebssystem dafür vorgesehenen Standardanwendung.

Beispiel:

```
opendoc "c:\\temp\\Dokument.pdf";
opendoc "c:\\temp\\Sound.wav";
opendoc "c:\\temp\\Brief.doc";
```

openwin

Typ: Befehl

Syntax: openwin *Element1 [, Element2] [, Element3]*

Beschreibung: Dieser Befehl öffnet ein Fenster, das zur Anzeige von Informationen während der Scriptausführung benutzt werden kann. Das Fenster kann folgende Elemente enthalten, die zur Laufzeit verändert werden können:

progressbar	Forschrittsbalken, der mit *progress* verändert werden kann
infotext	Textzeile, die mit *info* verändert werden kann
cancel	Button für Script-Abbruch

siehe auch: closewin, info, progress

Beispiel:

```
openwin infotext, progressbar;
select * from Kunde where Umsatz>1000;
   info current.recpos;
   progress current.reccount, current.recpos;
next
closewin;
```

Papierformate (Konstanten)

Typ: Konstante

Werte:

Name	Wert	Bedeutung
A4	"210x297"	DIN A4, 210 x 297 mm, Hochformat
A4quer	"297x210"	DIN A4, 297 x 210 mm, Querformat
Endlos	"216x305"	Endlospapier, 8.5 x 12 Zoll

picture

Typ: Druckbefehl

Syntax: picture *Bild, Breite, Höhe, [Logo], [Qualität]*

Beschreibung: Dieser Befehl druckt eine Grafik mit den Abmessungen für **Breite** und **Höhe.** Die linke obere Ecke der Grafik beginnt bei der aktuellen Cursorposition. Die Grafik wird automatisch auf die angegebenen Maße in mm skaliert. Folgende Bildformate sind zulässig: JPG, PNG, TIF, GIF, BMP, WMF. Je nach Ausgabeziel konvertiert DataCool einige Grafikformate intern in das JPG-Format. Aus Gründen der Geschwindigkeitsoptimierung empfehlen wir daher die Verwendung des Formates JPG. Die Cursorposition bleibt unverändert.

Wird die Breite auf 0 gesetzt, so erfolgt eine Skalierung entsprechend der angegebenen Höhe.

Wird die Höhe auf 0 gesetzt, so erfolgt eine Skalierung entsprechend der angegebenen Breite.

Die Parameter „**Logo**“ und „**Qualität**“ sind ab Build 1056 (DataCool 2014) verfügbar.

Parameter:

Bild	Dateiname inclusive Pfadangabe oder Bildfeld einer SQL-Tabelle
Breite	Breite der Grafik in mm
Höhe	Höhe der Grafik in mm
Logo	Die Angabe "true" definiert das Bild als wiederkehrendes Logo. Auf diese Weise kann die Größe einer PDF-Datei erheblich reduziert werden, da das Bild nur einmal in die PDF-Datei eingebettet wird.
Qualität	Prozentwert für die Ausgabequalität des Bildes in PDF-Dateien. Mögliche Werte liegen zwischen 10% (minimale Dateigröße, Qualitätsverluste durch hohe Kompression) und 100% (maximale Dateigröße, verlustfreie Ausgabe).

siehe auch: print

Beispiele:

```
picture "c:\\bilder\\briefkopf.jpg", 180, 30;
```

```
select Abbildung from Artikel where BestNr="4711";
   picture Abbildung, 50, 0;
next
```

```
select Logo from Setup;
   picture Logo, 70, 20, true, 100;
next
```

pos

Typ: Funktion

Syntax: pos(*Textausdruck, Teilzeichenfolge*)

Rückgabewert: Eine Ganzzahl

Beschreibung: Die Funktion sucht nach dem ersten Auftreten der **Teilzeichenfolge** im **Textausdruck**. Das Ergebnis ist die Position der **Teilzeichenfolge**. Ist die **Teilzeichenfolge** nicht enthalten so ist der Funktionswert 0.

Beispiel:

```
pos("DataCool", "Cool")
5

pos("DataCool", "y")
0
```

posrev

Typ: Funktion

Syntax: posrev(*Textausdruck, Teilzeichenfolge*)

Rückgabewert: Eine Ganzzahl

Beschreibung: Die Funktion sucht nach dem letzdem Auftreten der **Teilzeichenfolge** im **Textausdruck**. Das Ergebnis ist die Position der **Teilzeichenfolge** von vorne gerechnet. Ist die **Teilzeichenfolge** nicht enthalten so ist der Funktionswert 0.

Beispiel:

```
posrev("DataCool is cool", "cool")
14

posrev("DataCool", "y")
0
```

power

Typ: Funktion

Syntax: power(*Basis, Exponent*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion berechnet den Wert von **Basis** hoch **Exponent**.

siehe auch: log, sqrt

Beispiel:

```
power(2, 8)
```

```
256
```

price

Typ: Funktion

Syntax: price(*Betrag, Grenze, Dezimalwert, Schwelle, Stelle1, Stelle2*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion rundet Beträge auf im Einzelhandel gebräuchliche Verkaufspreise.

Parameter:

Betrag	Wert, der gerundet werden soll.
Grenze	Liegt der **Betrag** unterhalb der **Grenze**, so bleiben die Vorkommastellen unverändert. Andernfalls wird der **Betrag** nach einem speziellen Verfahren gerundet. zulässige Werte: 10 - 99999
Dezimalwert	Wert, der fix hinter dem Komma stehen soll. zulässige Werte: 0 - 99
Schwelle	Der **Betrag** wird abgerundet, wenn die letzte Ziffer der Vorkommastellen kleiner als die **Schwelle** ist. Andernfalls wird der **Betrag** aufgerundet. zulässige Werte: 0 - 9
Stelle1	Nach der Rundung ist die letzte Ziffer der Vorkommastellen entweder **Stelle1** oder **Stelle2**. zulässige Werte: 0 - 9
Stelle2	Nach der Rundung ist die letzte Ziffer der Vorkommastellen entweder **Stelle1** oder **Stelle2**. zulässige Werte: 0 - 9

Beispiel:

```
price(17.34, 20, 90, 4, 5, 9)
17.90

price(33.60, 20, 90, 4, 5, 9)
29.90

price(34.60, 20, 90, 4, 5, 9)
35.90

price(38.50, 20, 90, 4, 5, 9)
39.90
```

print

Typ: Druckbefehl

Syntax: print *Wert [, Ausrichtung] [, x-Position] [, y-Position] [, Winkel]*

Beschreibung: Mit dem Befehl *print* wird ein **Wert** ausgedruckt. Der **Wert** kann ein Feld, eine Variable oder ein Ausdruck sein. Der Cursor wird nach der Ausgabe an das Ende des Textes bewegt.

Die Ausrichtung bzgl. der aktuellen Cursorposition ergibt sich aus der Einstellung von *align* bzw. aus der Angabe für den optionalen Parameter **Ausrichtung** (left, center, right).

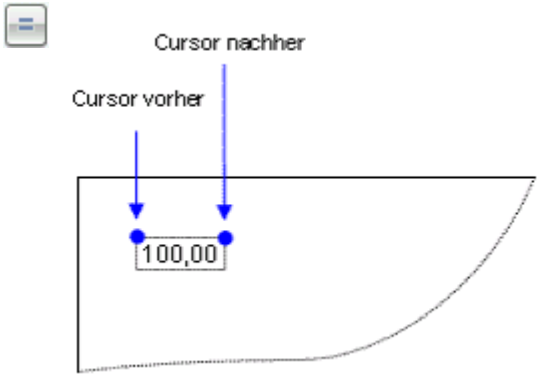
Mit den optionalen Parametern **x-Position** und **y-Position** kann die aktuelle Cursorposition vor der Ausgabe beeinflußt werden.

Der optionale Parameter **Winkel** bestimmt die Drehrichtung des auszugebenden Textes in Grad. Bei Angabe eines Winkels bleibt die Cursorposition unverändert und die Ausgabe erfolgt linksbündig.

siehe auch: align, tab

Beispiel:

```
dim Summe as number 6.2;
Summe = 100;
startjob printer;
print Summe;
endjob;
```



progress

Typ: Befehl

Syntax: progress *[Gesamt] [, Aktuell]*

Beschreibung: Dieser Befehl bewegt den Balken in einer Fortschrittsanzeige. Der Prozentwert der Anzeige wird aus den Werten von **Gesamt** und **Aktuell** berechnet.

Werden die beiden optionalen Paremeter nicht mit angegeben, so werden standardmäßig die Systemvariablen *current.reccount* für **Gesamt** und *current.recpos* für **Aktuell** verwendet.

siehe auch: openwin

Beispiel:

```
openwin infotext, progressbar;
select * from Kunde where Umsatz>1000;
   info current.recpos;
   progress current.reccount, current.recpos;
next
closewin;
```

public

Typ: Befehl

Syntax: public *Variable* as *Typ*

Beschreibung: Mit der Anweisung *public* wird eine **globale** Variable definiert. Globale Variablen dienen zum Informationsaustausch zwischen DataCool Scripten. Bitte beachten Sie, dass die Variable in allen betroffenen Scripten mit dem Befehl *public* deklariert werden muss.

Nach dem Schlüsselwort *as* werden Datentyp und Format festgelegt. DataCool unterstützt eindimensionale Datenfelder. Zur Definition eines Datenfeldes müssen Klammern nach dem Variablennamen gesetzt werden.

Typ	Beschreibung	Wertebereich
text n	Text mit der Länge n	n = 1-2048
numeric m	Nummer mit führenden Nullen (Länge m)	m = 1-9
number m	Gleitkommazahl mit Länge m	m = 1-9
number m.0	Ganzzzahl mit m Stellen	m = 1-9
number m.n	Festkommazahl mit m Vorkomma- und n Nachkommastellen	m = 1-9; n = 1-9
date	Datum (TT.MM.JJJJ)	
time	Uhrzeit (HH:MM)	
boolean	*true* oder *false* bzw. *ja* oder *nein*	

siehe auch: dim

Beispiel:

```
public Name as text 20;
public Nummer as numeric 4;
public Gleitkomma as number 5;
public Ganzzahl as number 5.0;
public Festkomma as number 5.2;
public Datum as date;
public Uhrzeit as time;
public Richtig as boolean;
public Zahlfeld() as number;
```

putval

Typ: Funktion

Syntax: putval(*Zeichenfolge*)

Beschreibung: Die Funktion überträgt eine **Zeichenfolge** in die Zwischenablage. Bei Erfolg wird die übertragene **Zeichenfolge** zurückgegeben, andernfalls ein Null-String.

siehe auch: getval

Beispiel:

```
putval("DataCool")
```

```
DataCool
```

query

Typ: Funktion

Syntax: query(*Fragetext [, Vorgabe]*)

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Die Funktion öffnet einen ja/nein Auswahldialog mit einem **Fragetext**.
Das Ergebnis ist *true*, wenn die Frage mit *ja* beantwortet wird.
Das Erbebnis ist *false* wenn die Frage mit *nein* beantwortet wird.
Per Default ist der "nein"-Button vorausgewählt und kann einfach per Enter-Taste bestätigt werden.
Wenn der optionale Parameter **Vorgabe** den Wert *true* erhält, dann wird der "ja"-Button als Default-Button vorbelegt.

siehe auch: message

Beispiel:

```
query("Alles löschen?")
```

random

Typ: Funktion

Syntax: random()

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion liefert eine Zufallszahl zwischen 0 und 1.

Beispiel:

```
random()
```

?

read

Typ: Funktion

Syntax: read()

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion liest eine Zeile aus der mit *open* geöffneten Datei.

siehe auch: write, writebase64, writehtml, writexml

Beispiel:

```
if open("c:\\test.txt") = true then
   message read();
   close;
end
```

readcom

Typ: Funktion

Syntax: readcom (*Trennzeichen*)

Rückgabewert: Boolean (*true* oder *false*) oder noexist

Beschreibung: Diese Funktion liest von der seriellen Schnittstelle des Computers bis zum **Trennzeichen**. Ist die Übertragung beendet wird noexist zurückgegeben.

siehe auch: closecom, opencom, readcom

Beispiel:

```
dim Num as text 10;

if opencom(1, "9600,N,8,1", true)=false then
   message "Fehler beim Initialisieren von COM1.";
   exit script;
end

truncate table BarcodeInventur;

repeat
   Num=readcom(chr(13));
   if Num<>noexist then
      insert into BarcodeInventur set Nummer=Num;
   end
until Num=noexist;

closecom;
```

rectangle

Typ: Druckbefehl

Syntax: rectangle *Breite, Höhe [, Linenbreite]*

Beschreibung: Zeichnet ein Rechteck mit den angegebenen Werten für Breite und Höhe. Das Rechteck beginnt mit der linken oberen Ecke an der aktuellen Cursorposition. Die Cursorposition bleibt unverändert. Alle Angaben erfolgen in der Maßeinheit mm.

siehe auch: shadow

Beispiel:

```
rectangle 100, 30, 0.3;
```

renamefile

Typ: Funktion

Syntax: renamefile(*DateinameAlt, DateinameNeu*)

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Die Funktion benennt die Datei **DateinameAlt** in **DateinameNeu** um.

siehe auch: copyfile, deletefile

Beispiel:

```
if renamefile("c:\\test.txt", "c:\\test.bak")=true
then
   message "Datei wurde umbenannt.";
else
   message "Fehler beim Umbenennen der Datei.";
end
```

savepicture

Typ: Funktion

Syntax: Savepicture(*Bild, Dateiname*)

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Dieser Befehl speichert die Information aus einem Bildfeld wieder als Datei ab. Der Funktionswert ergibt im Erfolgsfall *true* und *false* beim Auftreten eines Fehlers.

siehe auch: picture

Beispiel:

```
select Abbildung from Artikel where BestNr="4711";
   if savepicture(Abbildung, "c:\\4711.png")=false
      message "Fehler beim Speichern der Datei";
   end
next
```

repeat

Typ: Schleife

Syntax:
repeat
until Bedingung;

Beschreibung: Die repeat ... until Schleife führt eine Reihe von Anweisungen solange aus, bis eine gegebene **Bedingung** *true* wird.

siehe auch: exit, while

Beispiel:

```
repeat
    i = i + 1;
    message i;
until i = 10;
```

replace

Typ: Funktion

Syntax: replace(*Zeichenfolge, alt, neu*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion ersetzt alle Vorkommnisse der Zeichenfolge **alt** durch die Zeichenfolge **neu**. Bitte beachten Sie, dass die Funktion im Gegensatz zu anderen Funktion case-sensitiv ist. Dies bedeutet, dass Groß- und Kleinschreibung unterschieden wird.

siehe auch: lcase, normtext, ucase

Beispiel:

```
replace("DataBase", "Base", "Cool")
```

```
DataCool
```

rmove

Typ: Druckbefehl

Syntax: rmove *x, y*

Beschreibung: Bewegt den Cursor ausgehend von der aktuellen Position **x** mm nach rechts und **y** mm nach unten. Durch Angabe negativer Werte kann der Cursor nach links bzw. nach oben bewegt werden.

siehe auch: move, tab

Beispiel:

```
rmove 20, 50;
```

rollback

Typ: SQL-Befehl

Syntax: rollback

Beschreibung: Dieser Befehl bricht eine Transaktion ab. Eine Transaktion ist eine Abfolge von SQL-Befehlen, die als eine Einheit betrachtet werden. Eine Transaktion endet mit *commit*, *rollback* oder durch einen Verbindungsabbruch. Durch den Befehl *rollback* werden alle Änderungen zurückgenommen, die seit dem letzten dem Befehl *transaction* ausgeführt wurden.

Transaktionsbefehle wirken sich auf alle aktiven Verbindungen aus.

siehe auch: commit, transaction

Beispiel:

```
transaction;
select BelegNr from Rechnung where KdNr="314";
   insert into Storno set BelegNr=BelegNr;
   delete record;
next
if query("Storno wirklich ausführen?")=true then
   commit;
else
   rollback;
end
```

round

Typ: Funktion

Syntax: round(*Zahl, n*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion rundet die **Zahl** auf **n** Nachkommastellen.

siehe auch: ceil, floor, formatnumber, labvalue

Beispiel:

```
round(10.49, 0)
10
```

```
round(10.5, 0)
11
```

```
round(10.567, 2)
10.57
```

run

Typ: Befehl

Syntax: run *Pfad*

Beschreibung: Der Befehl *run* startet ein externes Programm.
Die Ausführung des Scripts wird nicht unterbrochen.

Beispiel:

```
run "c:\Programme\DataCool\DataCool.exe";
```

secpos

Typ: Funktion

Syntax: secpos(*ServerIP, POPID, WorkstationID, Befehl [, TAN] [, Betrag] [, Zahlungstyp]*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion ermöglicht Kartenzahlungen mit einem SECpos Server. Der SECpos Server ist ein Programm zur Steuerung von ELME Kassenterminals. Die Kommunikation erfolgt via Winsock über die XML-basierte OPI-Schnittstelle. Die Ports 20002 und 20007 dürfen nicht durch eine Firewall geblockt werden.

Der Parameter **ServerIP** legt die IP-Adresse des SECpos Servers fest. Die **POPID** ist eine 3-stellige Identifikationsnummer für das Zahlungsterminal. Die **WorkstationID** dient zur Identifikation des Kassen-PC's.

Der **Befehl** "closing" führt einen Tagesabschluss durch.
Das Ergebnis ist eine formatierte Zeichenfolge mit fester Länge:

TerminalID (8)	Zahlungstyp (20)	Anzahl (3)	Betrag (10)

Der gelb markierte Bereich wiederholt sich hierbei für jeden Zahlungstyp. Die Betragsangaben sind als Ganzahl mit Faktor 100 ausgeführt.

Mit dem **Befehl** "payment" wird der **Betrag** in EUR eingezogen.
Der optionale Parameter **Zahlungstyp** kann zur Einschränkung der zulässigen Zahlungsmedien verwendet werden:

Zahlungstyp	zulässige Zahlungsmedien
null	alle
"PIN"	EC-Cash (Girocard), Kreditkarte
"ELV"	ELV, Kreditkarte
"OLV"	OLV, Kreditkarte

Mit dem **Befehl** "reversal" wird die Transaktion mit der **TAN** storniert.

Bei der Ausführung der Befehle "payment" und "reversal" sind je nach Kartentyp folgende Rückgabewerte möglich:

Kartentyp	Rückgabewert
Lastschrift (ELV)	"L" + BLZ(8) + Konto(10) + TAN(6) + Bon
Lastschrift (OLV)	"O" + BLZ(8) + Konto(10) + TAN(6) + Bon
EC-Cash (girocard)	"P" + BLZ(8) + Konto(10) + TAN(6) + Bon
Eurocard	"E" + Karten-Nr(16) + Gültigkeit(5) + Aut-Nr(8) + TAN(6) + Bon
Mastercard	"M" + Karten-Nr(16) + Gültigkeit(5) + Aut-Nr(8) + TAN(6) + Bon
VISA	"V" + Karten-Nr(16) + Gültigkeit(5) + Aut-Nr(8) + TAN(6) + Bon
American Express	"A" + Karten-Nr(16) + Gültigkeit(5) + Aut-Nr(8) + TAN(6) + Bon
Sonstige	"K" + Karten-Nr(16) + Gültigkeit(5) + Aut-Nr(8) + TAN(6) + Bon

Beim Auftreten eines Fehlers ist der Rückgabewert der Funktion "F".

Beispiel:

```
secpos("payment", "192.168.1.1", "101", "Kasse1", "", 100.00, "ELV")
L721500000000004711000123

secpos("reversal", "192.168.1.1", "101", "Kasse1", "000123")
L721500000000004711000124
```

select

Typ: SQL-Befehl

Syntax: **select** Feldliste **from** Tabelle [**where** Auswahlbedingung]
[**order by** Sortierfelder] ;next

Beschreibung: Dieser Befehl selektiert Datensätze aus einer SQL-Tabelle. Wird als Feldliste das Zeichen ***** angegeben, so werden alle Felder (SQL-Spalten) in die Abfrage einbezogen. Alternativ kann eine komma-separierte Auflistung einzelner Feldnamen erfolgen. Das Schlüsselwort **where** dient zur Angabe einer Bedingung für die Satzauswahl. Mit der Option **order by** kann die Sortierreihenfolge festgelegt werden.

Jeder **select**-Befehl wird mit **next** abgeschlossen. Alle Anweisungen zwischen diesen beiden Schlüsselwörtern werden für jeden selektierten Datensatz einmal ausgeführt.
Die select/next-Anweisung funktioniert also wie eine Schleifenanweisung.

Tip: Durch die Auflistung benötigter Felder (anstelle von *) kann die Verarbeitungsgeschwindigkeit u.U. erheblich gesteigert werden, da die Ladezeit von der Datenmenge (Satzanzahl * Satzgröße) abhängig ist.

siehe auch: current.reccount, current.recpos

Beispiel:

```
select Nummer, Name from Kunde;
   message Name;
next
```

```
select * from Kunde where Umsatz>1000 order by Umsatz;
   message Name & ": " & Umsatz & " EUR";
next
```

selectfile

Typ: Funktion

Syntax: selectfile(*Pfad [, Erweiterung])*

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion öffnet einen Dateiauswahl-Dialog. Mit dem Parameter **Pfad** wird das voreingestellte Verzeichnis für die Dateiauswahl festgelegt. Der optionale Parameter **Erweiterung** dient zur Angabe eines Filters für den Dateityp. Die Dateierweiterung kann auch als semikolon-getrennte Auflistung übergeben werden. Der Rückgabewert ist der Datei mit Pfadangabe.

Wir der Auswahldialog ohne Auswahl abgebrochen, so ist der Rückgabewert null.

Beispiel:

```
selectfile("C:\\Windows\\System32", "nt")
```

sendmail

Typ: Funktion

Syntax: sendmail(*Absendername, Absender, Ziel, Betreff, Nachricht [, Anlagen()] [, Zip] [, CC]*)

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Mit dieser Funktion wird eine E-Mail Nachricht erstellt und vollautomatisch verschickt. Der Rückgabewert ist *true*, wenn die Nachricht erfolgreich an den SMTP-Server übergeben werden konnte.

Parameter:

Absendername	Name des Absenders
Absender	Absendeadresse
Ziel	Zieladresse (oder semikolon-getrennte Empfängerliste)
Betreff	Betreff (Subject)
Nachricht	Nachricht (Mailbody)
Anlage	Optionale Anlage (einzelner Dateiname oder Datenfeld mit Dateinamen)
Zip	Anlage zippen
CC	Zusätzliche Ziel-Adressen (oder semikolon-getrennte Empfängerliste)

siehe auch: connect, disconnect

Beispiel:

```
dim Abs as text 20;
dim AAdr as text 40;
dim EAdr as text 40;
dim Betr as text 50;
dim Nachr as text 100;

Abs = "Absender";
AAdr = "absender@freenet.de";
EAdr = "empfaenger@freenet.de";
Betr = "Grüße";
Nachr = "Herzliche Grüße vom DataCool Team.";

if connect("mx.freenet.de", "usr", "pwd")=true then
   if sendmail(Abs, AAdr, EAdr, Betr, Nachr)=true then
      message "Nachricht wurde versendet";
   else
      message "Fehler beim Senden der Nachricht";
   end
   disconnect;
else
   message "Verbindungsaufbau fehlgeschlagen";
end
```

sequence

Typ: Funktion

Syntax: sequence(Formularname, Feldname)

Rückgabewert: Eine Nummer

Beschreibung: Die Funktion bestimmt die nächste fortlaufende Nummer in einem Formular. Die Verwendung ist nur für Felder vom Typ Nummer zulässig.

Beispiel:

```
sequence("Rechnung", "Satznummer")
000001
```

setyear

Typ: Funktion

Syntax: setyear(*Datum, Jahr*)

Rückgabewert: Ein Datum

Beschreibung: Die Funktion verändert die Jahreszahl eines Datums. Ergibt sich durch die Veränderung ein ungültiges Datum (z.B. 29.02.2005), so erfolgt eine Vorverlegung auf den letzten Tag des Monats (z.B. 28.02.2005).

Beispiel:


```
setyear(24.12.2004, 2006)
24.12.2006
```


```
setyear(29.02.2004, 2005)
28.02.2005
```

shadow

Typ: Druckbefehl

Syntax: shadow *Breite, Höhe, Grauwert*

Beschreibung: Zeichnet ein grau gefülltes Rechteck mit den angegebenen Werten für Breite und Höhe. Das Rechteck beginnt mit der linken oberen Ecke an der aktuellen Cursorposition. Die Cursorposition bleibt unverändert. Breite und Höhe werden in mm angegeben. Der Grauwert ist der Prozentwert der Schattierung (0=weiß, 100=schwarz).

siehe auch: rectangle

Beispiel:

```
shadow 100, 30, 20;
```

shiftday

Typ: Funktion

Syntax: shiftday(*Datum, n*)

Rückgabewert: Ein Datum

Beschreibung: Die Funktion verschiebt ein **Datum** um **n** Tage.
Negative Werte für **n** bewirken eine Verschiebung in die Vergangenheit.

siehe auch: shiftmonth, shiftyear

Beispiel:


```
shiftday(24.12.2004, 13)
06.01.2005
```

shiftmonth

Typ: Funktion

Syntax: shiftmonth(*Datum, n*)

Rückgabewert: Ein Datum

Beschreibung: Die Funktion verschiebt ein **Datum** um **n** Monate.
Negative Werte für **n** bewirken eine Verschiebung in die Vergangenheit.

Ergibt sich durch die Verschiebung ein ungültiges Datum (z.B. 31.02.2004), so erfolgt eine Vorverlegung auf den letzten Tag des Monats (z.B. 29.02.2004).

siehe auch: shiftday, shiftyear

Beispiel:

```
shiftmonth(24.12.2004, 6)
24.06.2005
```

shiftyear

Typ: Funktion

Syntax: shiftyear(*Datum, n)*

Rückgabewert: Ein Datum

Beschreibung: Die Funktion verschiebt ein **Datum** um **n** Jahre.
Negative Werte für **n** bewirken eine Verschiebung in die Vergangenheit.

Ergibt sich durch die Verschiebung ein ungültiges Datum (z.B. 29.02.2003), so erfolgt eine Vorverlegung auf den letzten Tag des Monats (z.B. 28.02.2003).

siehe auch: shiftday, shiftmonth

Beispiel:

```
shiftyear(24.12.2003, 1)
24.12.2004
```

sin

Typ: Funktion

Syntax: sin(*Zahl*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion berechnet den Sinus eines Winkels.

siehe auch: cos, tan

Beispiel:

```
sin(0)
0

sin(90)
1
```

source

Typ: Druckbefehl

Syntax: source *Schacht*

Beschreibung: Wählt eine Papierquelle für den Ausdruck der nächsten Seite.
Die möglichen Werte für die Schachtnummer sind druckerabhängig.

Beispiel:

```
source 2;
```

spellmonth

Typ: Funktion

Syntax: spellmonth(*Datum*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion gibt den Monatsnamen eines **Datums** zurück.

siehe auch: spellweekday

Beispiel:

```
spellmonth(24.12.2004)
```

```
Dezember
```

spellnumber

Typ: Funktion

Syntax: spellnumber(*Zahl*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion rundet die übergebene *Zahl* und wandelt diese in Worte um. Gewandelt werden Ganzzahlen im Bereich 1 bis 999 Milliarden.

Beispiel:

```
spellnumber(1237)
```

```
eintausendzweihundertsiebenunddreißig
```

spellweekday

Typ: Funktion

Syntax: spellweekday(*Datum*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion gibt den Wochentag eines **Datums** als Text zurück.

siehe auch: spellmonth

Beispiel:

```
spellweekday(24.12.2004)
```

```
Freitag
```

split

Typ: Funktion

Syntax: split(*Zeichenfolge*, *Trennzeichen*)

Rückgabewert: Ein Array

Beschreibung: Die Funktion *split* wird benutzt, um ein Datenfeld (array) mit Werten zu belegen. Die einzelnen Segmente werden mit Hilfe des übergebenen Trennzeichens aus der Zeichenfolge extrahiert.

siehe auch: array

Beispiel:

```
dim i as number 1.0;
dim Farbe() as text 10;
Farbe = split("rot;grün;blau", ";");
i=0;
while i<3;
   i=i+1;
   message Farbe(i);
wend
```

sqldirect

Typ: SQL-Befehl

Syntax: sqldirect *Befehl [, FehlerAnzeige] [, Verbindungsname]*

Beschreibung: Mit dem Befehl **sqldirect** wird der angegebene **Befehl** ohne weitere Prüfungen an den SQL-Server weitergeleitet.

Der optionale Parameter **FehlerAnzeige** bestimmt, ob das Script beim Auftreten eines Fehlers abgebrochen wird. Die Default-Einstellung *true* bewirkt im Fehlerfalle einen Abbruch der Transaktion sowie eine Anzeige der SQL-Server-Meldung am Bildschirm.

Mit dem Parameterwert *false* kann das Script auch im Fehlerfalle ohne die Anzeige einer Meldung fortgesetzt werden. In diesem Fall wird eine laufende Transaktion nicht unterbrochen.

Der optionale **Verbindungsname** ermöglicht die Ausführung von Befehlen auf fremden Datenbanken.

Beispiel:

```
sqldirect "drop table Artikel";
```

sqrt

Typ: Funktion

Syntax: sqrt(*Zahl)*

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion berechnet die Quadratwurzel der **Zahl**.
Die Quadratwurzel kann nur aus positiven Zahlen gezogen werden.

siehe auch: log, power

Beispiel:

```
sqrt(256)
16
```

start

Typ: Befehl

Syntax: start *Script*

Beschreibung: Der Befehl *start* führt ein Script aus.
Es werden nur Scripte ohne eigenes Eingabeformular gestartet.

Beispiel:

```
start "Datenbankabgleich";
```

startbrowser

Typ: Befehl

Syntax: startbrowser *url*

Beschreibung: Dieser Befehl startet den Standard-Browser von Windows und öffnet die Webseite mit der Adresse **url**.

Beispiel:

```
startbrowser "www.datacool.net";
```

startjob

Typ: Druckbefehl

Syntax: startjob *Ziel [, Jobname] [, Format] [, Links] [, Oben] [, Kopien]*

Beschreibung: Der Befehl *startjob* startet einen Druckauftrag. Jeder Druckauftrag muss mit *endjob* beendet werden. Der Parameter **Ziel** ist immer erforderlich. Alle anderen Parameter sind optional.

Parameter:

Ziel	Bestimmt das Ausgabeziel: *printer*, *screen* oder *pdf*. Die Angabe *printer* bewirkt eine Ausgabe auf dem Standarddrucker von Windows. Alternativ kann auch ein Drucker-Aliasname aus der DataCool-Konfiguration angegeben werden. Aliasnamen werden zur Laufzeit unter Verwendung der Arbeitsplatz-Konfiguration in Treibernamen übersetzt.
Jobname	Bei Ausgabeziel *pdf* oder *screen* wird hier der Dateiname der PDF-Datei festgelegt. Die Endung ".pdf" wird ggf. automatisch ergänzt falls sie fehlt. Wenn der Jobname keine Pfadangabe enthält, dann wird die Datei im Benutzerverzeichnis für temporäre Dateien abgelegt.
Format	Angabe des Papierformates: *A4*, *A4quer* oder *A4endlos*. Andere Papierformate können mit der Schreibweise "BxH" festgelegt werden. Die Angabe "148x210" entspricht beispielsweise dem Format DIN A5. Der Default-Wert ist DIN A4 (Hochformat).
Links	Linker Rand in mm (Default=10).
Oben	Oberer Rand in mm (Default=10).
Kopien	Anzahl der zu druckenden Exemplare (Default=1).

Ziel	Wert	Bedeutung
printer	1	Ausgabe auf Drucker
screen	2	Ausgabe auf Bildschirm
pdf	3	Ausgabe in PDF-Datei

Format	Wert	Bedeutung
A4	"210x297"	DIN A4, 210 x 297 mm, Hochformat
A4quer	"297x210"	DIN A4, 297 x 210 mm, Querformat
Endlos	"216x305"	Endlospapier, 8.5 x 12 Zoll

siehe auch: endjob

Beispiel:

```
startjob printer;
print "Hello World!";
endjob;
```

startmailer

Typ: Befehl

Syntax: startmailer *Adresse [, Betreff] [, Nachricht] [, Anlagen()]*

Beschreibung: Dieser Befehl startet das Standard-Mailprogramm von Windows und erstellt eine neue Nachricht mit den Angaben für **Adresse**, **Betreff** und **Nachricht**. Der Parameter **Anlagen** dient zur Festlegung optionaler Anlagen (einzelner Dateiname oder Datenfeld mit Dateinamen), die an die Nachricht angehängt werden sollen.

Beispiel:

```
startmailer "mail@web.de", "Gruß", "Hello World!",
"C:\\boot.ini";
```

sum

Typ: Funktion

Syntax: sum(*Subformular*, *Feldname*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion addiert alle Werte einer Spalte des Subformulares.

siehe auch: count, occurrence

Beispiel:

```
sum(Rechnungsposition, Preis)
```

```
Gesamtbetrag
```

tab

Typ: Druckbefehl

Syntax: tab *x*

Beschreibung: Bewegt den Cursor innerhalb der aktuellen Zeile an die absolute Position **x**. Die Angabe erfolgt in der Maßeinheit mm.

siehe auch: move, rmove

Beispiel:

```
tab 20;
```

tan

Typ: Funktion

Syntax: tan(*Zahl*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion berechnet den Tangens eines Winkels.

siehe auch: cos, sin

Beispiel:

```
tan(0)
0
```

```
tan(45)
1
```

time2h

Typ: Funktion

Syntax: time2h(*Zeichenfolge*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion wandelt das übergebene Zeitliteral (h:m) in Stunden (dezimal) um.

siehe auch: m2time, h2time, time2m

Beispiel:

```
time2h("02:30")
2.5

time2h("100:15")
100.25

time2h("-03:45")
-3.75
```

time2m

Typ: Funktion

Syntax: time2m(*Zeichenfolge*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion wandelt das übergebene Zeitliteral (h:m) in Minuten um.

siehe auch: m2time, h2time, time2h

Beispiel:

```
time2m("02:30")
150

time2m("100:15")
6015

time2m("-03:45")
-225
```

timediff

Typ: Funktion

Syntax: timediff(*Start, Ende*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion berechnet den Zeitraum zwischen zwei Uhrzeiten im Format '00:00' und gibt das Ergebnis in Minuten zurück.

Beispiel:

```
timediff("12:30", "16:45")
255

timediff("23:00", "03:00")
240
```

timespan

Typ: Funktion

Syntax: timespan(*Zeichenfolge*)

Rückgabewert: Eine Zahl

Beschreibung: Die Funktion extrahiert aus **Zeichenfolge** einen Zeitraum im Format '00:00-00:00' und berechnet hieraus die Zeit zwischen diesen beiden Uhrzeiten in Stunden.

Beispiel:

```
timespan("Arbeitsaufwand 01.01.2007, 12:30-16:45")
```

```
4.25
```

transaction

Typ: SQL-Befehl

Syntax: transaction

Beschreibung: Dieser Befehl markiert den Anfang einer Transaktion. Eine Transaktion ist eine Abfolge von SQL-Befehlen, die als eine Einheit betrachtet werden. Eine Transaktion endet mit *commit*, *rollback* oder durch einen Verbindungsabbruch.
Alle Operationen innerhalb der Transaktion werden erst wirksam, wenn sie mit *commit* bestätigt worden sind.

Transaktionsbefehle wirken sich auf alle aktiven Verbindungen aus.

siehe auch: commit, rollback

Beispiel:

```
transaction;
select BelegNr from Rechnung where KdNr="314";
   insert into Storno set BelegNr=BelegNr;
   delete record;
next
commit;
```

trim

Typ: Funktion

Syntax: trim(*Zeichenfolge*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion gibt eine **Zeichenfolge** ohne führende und folgende Leerzeichen zurück.

siehe auch: fill

Beispiel:

```
trim("   DataCool   ")
```

```
DataCool
```

truncate

Typ: SQL-Befehl

Syntax: **truncate table** Tabelle;

Beschreibung: Mit dem Befehl **truncate** werden **alle** Datensätze einer Tabelle gelöscht. Hierbei wird das Rollback-Segment nicht benutzt. Die Löschung ist daher erheblich schneller als bei Verwendung des Befehls 'delete from'.

Beispiel:

```
truncate table Kunde;
```

unzip

Typ: Funktion

Syntax: unzip(Archivname, Zielordner [, Löschen] [, Passwort])

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Mit dieser Funktion kann eine komprimierte ZIP-Datei wieder entpackt werden. Der Rückgabewert ist *true*, wenn das Archiv erfolgreich entpackt werden konnte.

Parameter:

Archivname	Dateiname des zu entpackenden Archivs (incl. Pfadangabe)
Zielordner	Ordner in den entpackt wird
Löschen	Archivdatei hinterher löschen (default: 'false')
Passwort	Passwort zum Entpacken des ZIP-Archivs (default: ohne)

siehe auch: sendmail, zip

Beispiel:

```
if zip("C:\\Temp\\DataCool.zip", "C:\\Temp")=true
then
   message "Die ZIP-Datei wurde entpackt.";
end
```

```
if zip("C:\\Temp\\Neuhahn.zip", "C:\\Temp",
 true, "0815")=true then
   message "Die ZIP-Datei wurde entpackt.";
end
```

upper

Typ: Funktion

Syntax: upper(*Zeichenfolge*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion wandelt alle Buchstaben der **Zeichfolge** in Großbuchstaben.

siehe auch: lower, normtext, replace

Beispiel:

```
upper("Structured Query Language")
```

```
STRUCTURED QUERY LANGUAGE
```

update

Typ: SQL-Befehl

Syntax: **update** Tabelle **set** Feld1=Wert1, Feld2=Wert2, ..., Feldn=Wertn [**where** Bedingung] ;

oder

update record set Feld1=Wert1, Feld2=Wert2, ..., Feldn=Wertn;

Beschreibung: Mit dem Befehl **update** werden alle selektierten Datensätze einer Tabelle geändert. Der Befehl **update record** ändert nur den mit *select* ausgewählten Datensatz. Nach dem Schlüsselwort **set** werden alle Felder mit den zugehörigen Werten belegt.

siehe auch: insert, delete

Beispiel:

```
update Kunde set Umsatz = 0 where Jahr = 2004;
```

```
select * from Kunde where Jahr = 2004;
   update record set Umsatz = 0;
next
```

utf2win

Typ: Funktion

Syntax: utf2win(*Zeichenfolge*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion wandelt den Zeichensatz von UTF-8 nach Windows.
Der Zeichensatz UTF-8 findet beispielsweise in XML-Dateien Verwendung.

siehe auch: dos2win, win2dos, win2utf

Beispiel:


`utf2win(Zeichenfolge)`


Darstellung der Zeichenfolge mit dem Windows-Zeichensatz

vatcheck

Typ: Funktion

Syntax: vatcheck(*Ländercode, USt-Id-Nummer*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion prüft die Gültigkeit einer USt-Id-Nummer und gibt diese formatiert zurück. Wird eine ungültige Nummer übergeben oder sind für ein Land keine Prüfregeln hinterlegt, so wird der Wert *Null* zurückgegeben.

siehe auch: vatsample

Beispiel:

```
vatcheck("DE", "128585731")
DE128585731

vatcheck("DE", "DE128585731")
DE128585731

vatcheck("DE", "ABC")
Null
```

vatsample

Typ: Funktion

Syntax: vatsample(*Ländercode*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion gibt den Formatstring der USt-ID-Nummer des übergebenen Landes zurück.

siehe auch: vatcheck

Beispiel:


```
vatcheck("DE")
DE000000000

vatcheck("AT")
ATU00000000
```

vline

Typ: Druckbefehl

Syntax: vline *Länge [, Breite]*

Beschreibung: Zeichnet eine vertikale Linie mit den angegebenen Werten für Länge und Breite. Ausgangspunkt der Linie ist die aktuelle Cursorposition. Die Cursorposition bleibt unverändert. Die Angabe der Parameter erfolgt in mm.

siehe auch: line, hline

Beispiel:

```
vline 100, 0.3;
```

weekday

Typ: Funktion

Syntax: weekday(*Datum*)

Rückgabewert: Eine Ganzzahl

Beschreibung: Die Funktion gibt den Wochentag als Zahl zwischen 1 (Montag) und 7 (Sonntag) zurück.

Beispiel:

```
weekday(24.12.2004)
```

```
5
```

while

Typ: Schleife

Syntax: while Bedingung;
wend

Beschreibung: Die while ... wend Schleife führt eine Reihe von Anweisungen aus, solange eine gegebene **Bedingung** *true* ist.

siehe auch: exit, repeat

Beispiel:

```
while i < 10;
    i = i + 1;
    message i;
wend
```

win2dos

Typ: Funktion

Syntax: win2dos(*Zeichenfolge*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion wandelt den Zeichensatz von Windows nach DOS.

siehe auch: dos2win, utf2win, win2utf

Beispiel:

```
win2dos("Übel, übel sprach der Dübel und
verschwand in der Wand.")
```

Darstellung der Zeichenfolge mit DOS-Zeichensatz

win2utf

Typ: Funktion

Syntax: win2utf(*Zeichenfolge*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion wandelt den Zeichensatz von Windows nach UTF-8.
Der Zeichensatz UTF-8 findet beispielsweise in XML-Dateien Verwendung.

siehe auch: dos2win, utf2win, win2dos

Beispiel:

```
win2utf(Zeichenfolge)
```

Darstellung der Zeichenfolge mit Zeichensatz UTF-8

win2xml

Typ: Funktion

Syntax: win2xml(*Zeichenfolge*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion wandelt die Zeichenfolge in von XML darstellbare Zeichen.

siehe auch: win2dos, win2utf, xml2win

Beispiel:

```
win2xml("&<>äöüÄÖÜ")
```

```
&#38;&#60;&#62;&#228;&#246;&#252;&#196;&#214;&#220;
```

wordclose

Typ: Befehl

Syntax: wordclose

Beschreibung: Der Befehl schliesst das zuvor mit *wordopen* geöffnete Word-Dokument und beendet die Anwendung Word..

siehe auch: wordopen, wordprint, wordreplace, wordsave

Beispiel:

```
if open("f:\\vorlagen\\faxvorlage.dot")=true then
   wordreplace("#Name#", Form.Name);
   wordreplace("#Fax#", Form.Fax);
   if wordsave("f:\\rundschreiben\\fax.doc")=false
   then
      message "Fehler beim Speichern.";
      exit script;
   end
   wordprint;
   message "Das Dokument wurde gedruckt.";
   wordclose;
end
```

wordopen

Typ: Funktion

Syntax: wordopen(*Vorlage*, *Sichtbar*)

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Die Funktion öffnet ein Word-Dokument oder eine Word-Vorlage zur weiteren Bearbeitung. Der Parameter **Sichtbar** bestimmt, ob das Programm Word hierbei angezeigt werden soll (true) oder im Hintergrund ausgeführt werden soll (false).

siehe auch: wordclose, wordprint, wordreplace, wordsave

Beispiel:

```
if open("f:\\vorlagen\\faxvorlage.dot")=true then
   wordreplace("#Name#", Form.Name);
   wordreplace("#Fax#", Form.Fax);
   if wordsave("f:\\rundschreiben\\fax.doc")=false
   then
      message "Fehler beim Speichern.";
      exit script;
   end
   wordprint;
   message "Das Dokument wurde gedruckt.";
   wordclose;
end
```

wordprint

Typ: Befehl

Syntax: wordprint

Beschreibung: Der Befehl druckt das zuvor mit *wordopen* erstellte Word-Dokument. Die Ausgabe erfolgt auf dem Standarddrucker von Word.

siehe auch: wordclose, wordopen, wordreplace, wordsave

Beispiel:

```
if open("f:\\vorlagen\\faxvorlage.dot")=true then
   wordreplace("#Name#", Form.Name);
   wordreplace("#Fax#", Form.Fax);
   if wordsave("f:\\rundschreiben\\fax.doc")=false
   then
      message "Fehler beim Speichern.";
      exit script;
   end
   wordprint;
   message "Das Dokument wurde gedruckt.";
   wordclose;
end
```

wordreplace

Typ: Befehl

Syntax: wordreplace(*alt*, *neu*)

Beschreibung: Der Befehl ersetzt die Suchzeichenfolge **alt** durch den Wert **neu**.
Mit Hilfe dieses Befehls können Sie Platzhalter in einer Word-Vorlage durch Werte aus dem Script ersetzen.

siehe auch: wordclose, wordopen, wordprint, wordsave

Beispiel:

```
if open("f:\\vorlagen\\faxvorlage.dot")=true then
   wordreplace("#Name#", Form.Name);
   wordreplace("#Fax#", Form.Fax);
   if wordsave("f:\\rundschreiben\\fax.doc")=false
   then
      message "Fehler beim Speichern.";
      exit script;
   end
   wordprint;
   message "Das Dokument wurde gedruckt.";
   wordclose;
end
```

wordsave

Typ: Funktion

Syntax: wordsave(*Pfad*)

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Die Funktion speichert das zuvor mit *wordopen* erstellte neue Word-Dokument. Der **Pfad** ist der Name der neuen Word-Datei. Der Funktionswert ergibt *true* im Erfolgsfall und *false* beim Auftreten eines Fehlers.

siehe auch: wordclose, wordopen, wordprint, wordreplace

Beispiel:

```
if open("f:\\vorlagen\\faxvorlage.dot")=true then
   wordreplace("#Name#", Form.Name);
   wordreplace("#Fax#", Form.Fax);
   if wordsave("f:\\rundschreiben\\fax.doc")=false
   then
      message "Fehler beim Speichern.";
      exit script;
   end
   wordprint;
   message "Das Dokument wurde gedruckt.";
   wordclose;
end
```

write

Typ: Befehl

Syntax: write *Zeichenfolge*

Beschreibung: Der Befehl schreibt die **Zeichenfolge** in die zuvor mit *create* geöffnete Datei.

siehe auch: read, writebase64, writehtml, writexml

Beispiel:

```
if create("c:\\test.txt") = true then
   write "Hello World!\n";
   close;
end
```

writebase64

Typ: Befehl

Syntax: writebase64 *Tag, Bild*

Beschreibung: Der Befehl wandelt die Binärdaten eines Bildfeldes in das Base64-Format und schreibt dieses in die zuvor mit *create* geöffnete Datei. Die Base64-Zeichenfolge wird dabei zwischen zwei XML-Tags eingeschlossen.

Parameter:

Tag	Name des XML-Tags (ohne spitze Klammern)
Bild	Dateiname inclusive Pfadangabe oder Bildfeld einer SQL-Tabelle

siehe auch: read, write, writehtml, writexml

Beispiel:

```
if create("c:\\test.xml") = true then
   select Abbildung from Artikel where BestNr="4711";
      writebase64 "Bilddaten", Abbildung;
      close;
   next
end
```

```
<Bilddaten>
AAABAAEAGBgAAAEAGABIBwAAFgAAACgAAAAYAAAAMAAAAAEAGAAAAAAAIA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAD17+jdxrDFnne2hFSxfUqxfUq2
hFTFnnfdxrD17+gAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAD69/TTtpmxfUqxfUqxfUqxfUqxfUqxfUqxfUqxfUqxfUqxfUrTtpn6
9/QAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADw5927jGGxfUqxfUrFnn
fizrvw593///////w593izrvFnnexfUqxfUq7jGHw590AAAAAAAAAAAAA
AAAAAAAAAAAAAADw5922hFSxfUq7jWHs39L/////////////////////
/////////s39K7jWGxfUq2hFTw590AAAAAAAAAAAAAAAAAAD69/S7jGGx
yxfUq7jGH69/QAAAAAAAAAAAAAADTtpmxfUq7jWH69/T/////////////
AAAAD17+ixfUqxfUrs39L/////////////////////////////////////
//////8=
<Bilddaten>
```

verfügbar ab Build 1092

writecom

Typ: Befehl

Syntax: writecom *Zeichenfolge*

Beschreibung: Diese Funktion sendet eine Zeichenfolge an die serielle Schnittstelle des Computers.

siehe auch: closecom, opencom, readcom

Beispiel:

```
if opencom(1, "9600,N,8,1", true)=false then
   message "Fehler beim Initialisieren von COM1.";
   exit script;
end

select Nummer from BarcodeInventur;
   writecom Nummer & chr(13);
next

closecom;
```

writehtml

Typ: Befehl

Syntax: writehtml *Zeichenfolge*

Beschreibung: Der Befehl schreibt die **Zeichenfolge** in die zuvor mit *create* geöffnete Datei. Hierbei findet eine Umwandlung des Zeichensatzes für HTML statt.

siehe auch: read, write, writexml

Beispiel:

```
if create("c:\\test.htm") = true then
   writehtml "Hello World!";
   close;
end
```

writexml

Typ: Befehl

Syntax: writexml *Tag, Zeichenfolge*

Beschreibung: Der Befehl schreibt die **Zeichenfolge** in die zuvor mit *create* geöffnete Datei. Die angegebene Zeichenfolge wird dabei zwischen zwei XML-Tags eingeschlossen.

siehe auch: read, write, writehtml

Beispiel:

```
if create("c:\\test.xml") = true then
   writexml "info", "Hello World!";
   close;
end
```

```
<info> Hello World! </info>
```

xml2win

Typ: Funktion

Syntax: xml2win(*Zeichenfolge*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion wandelt die XML-Zeichenfolge in den Windows-Zeichensatz.

siehe auch: win2xml

Beispiel:

```
xml2win ("&#38;&#60;&#62;&#228;&#246;&#252;&#196;&#214;&#220;")
```

&<>äöüÄÖÜ

verfügbar ab Build 1107

xmlclose

Typ: Befehl

Syntax: xmlclose

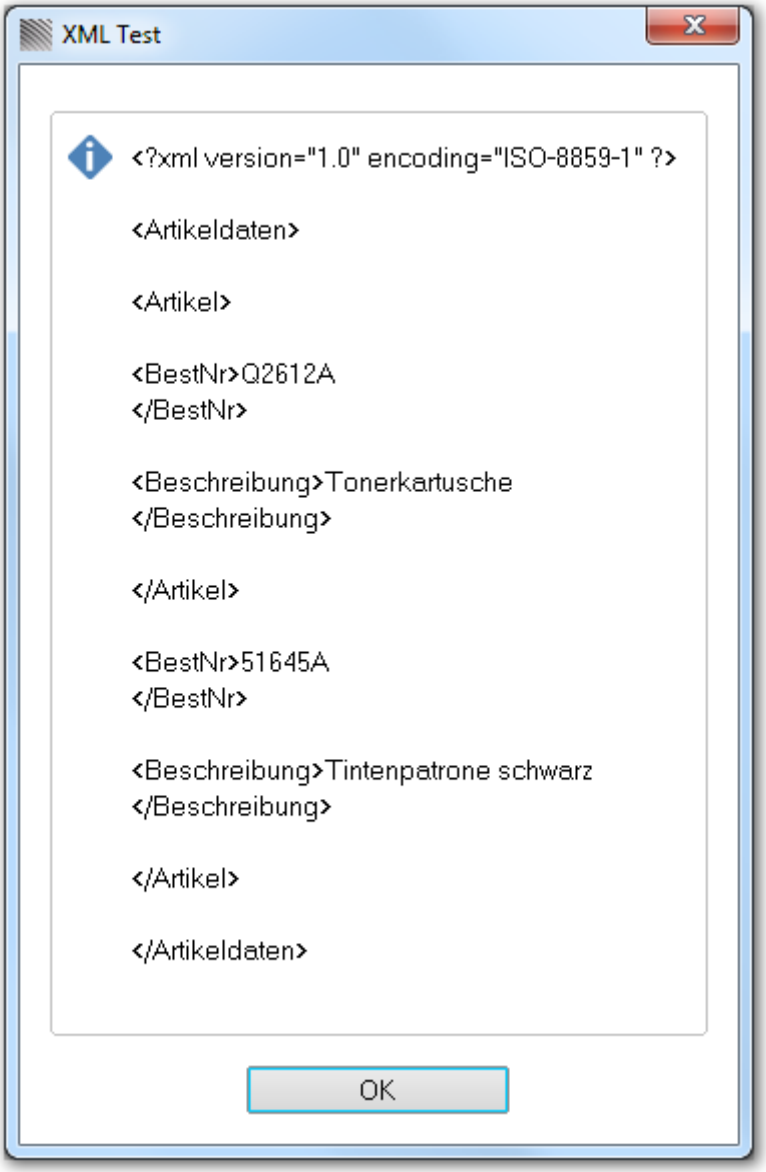
Beschreibung: Der Befehl schließt eine XML-Datei, die zuvor mit *xmlopen* geöffnet wurde.

siehe auch: xmleof, xmlopen, xmlread, xmlvalue

Beispiel 1:

```
dim buffer as text 2048;

if xmlopen("c:\\temp\\artikel.xml")=false then
   message "Fehler beim Öffnen der XML-Datei.";
   exit script;
end

while xmleof()=false;
   buffer=buffer & xmlread() & "\n";
wend

xmlclose;

message buffer;
```

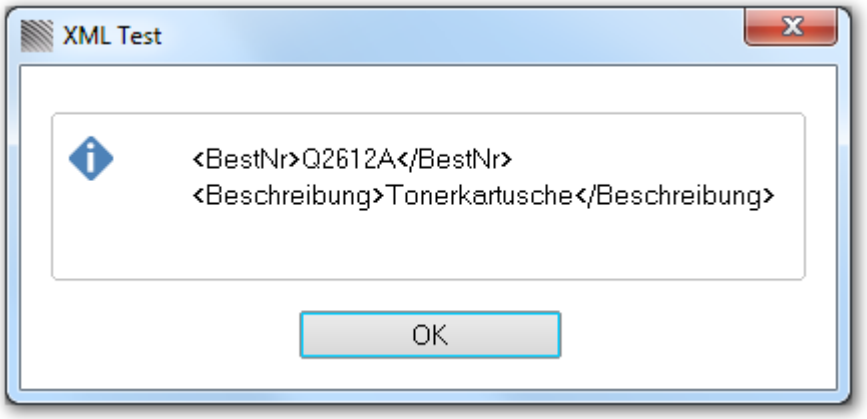
```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <Artikeldaten>
  - <Artikel>
      <BestNr>Q2612A</BestNr>
      <Beschreibung>Tonerkartusche</Beschreibung>
    </Artikel>
  - <Artikel>
      <BestNr>51645A</BestNr>
      <Beschreibung>Tintenpatrone schwarz</Beschreibung>
    </Artikel>
  </Artikeldaten>
```


XML Test
<?xml version="1.0" encoding="ISO-8859-1" ?>
<Artikeldaten>
<Artikel>
<BestNr>Q2612A
</BestNr>
<Beschreibung>Tonerkartusche
</Beschreibung>
</Artikel>
<BestNr>51645A
</BestNr>
<Beschreibung>Tintenpatrone schwarz
</Beschreibung>
</Artikel>
</Artikeldaten>
OK

Beispiel 2:

```
if xmlopen("c:\\temp\\artikel.xml", "Artikel")=false
then
   message "Fehler beim Öffnen der XML-Datei.";
   exit script;
end

while xmleof()=false;
   message xmlread();
wend

xmlclose;
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <Artikeldaten>
  - <Artikel>
      <BestNr>Q2612A</BestNr>
      <Beschreibung>Tonerkartusche</Beschreibung>
    </Artikel>
  - <Artikel>
      <BestNr>51645A</BestNr>
      <Beschreibung>Tintenpatrone schwarz</Beschreibung>
    </Artikel>
  </Artikeldaten>
```



xmleof

Typ: Funktion

Syntax: xmleof()

Rückgabewert: Boolean (*true* oder *false*)

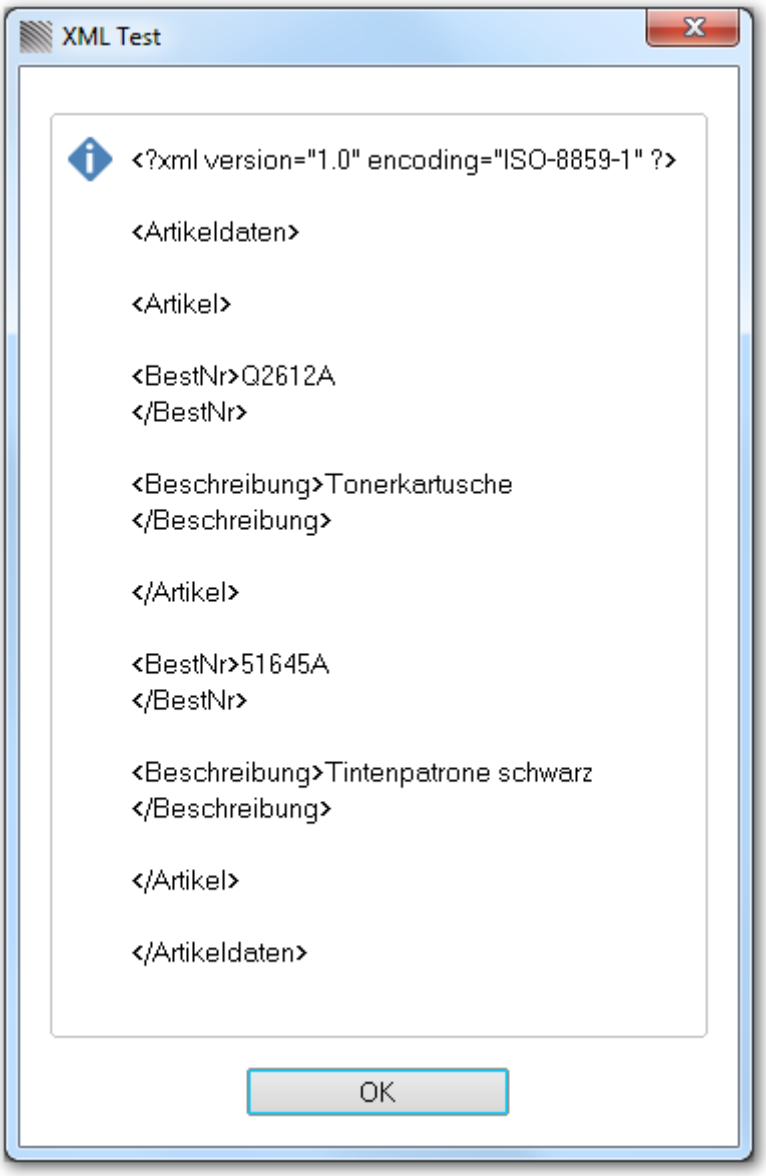
Beschreibung: Die Funktion liefert als Ergebnis den Wert *true*, wenn das Ende der mit *xmlopen* geöffneten Datei erreicht ist. Andernfalls ist der Rückgabewert *false*.

siehe auch: xmlclose, xmlopen, xmlread, xmlvalue

Beispiel 1:

```
dim buffer as text 2048;

if xmlopen("c:\\temp\\artikel.xml")=false then
   message "Fehler beim Öffnen der XML-Datei.";
   exit script;
end

while xmleof()=false;
   buffer=buffer & xmlread() & "\n";
wend

xmlclose;

message buffer;
```

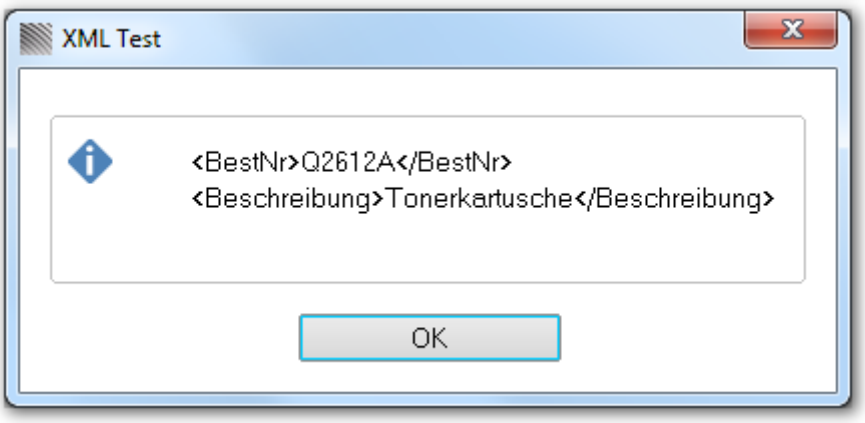
```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <Artikeldaten>
  - <Artikel>
      <BestNr>Q2612A</BestNr>
      <Beschreibung>Tonerkartusche</Beschreibung>
    </Artikel>
  - <Artikel>
      <BestNr>51645A</BestNr>
      <Beschreibung>Tintenpatrone schwarz</Beschreibung>
    </Artikel>
  </Artikeldaten>
```



Beispiel 2:

```
if xmlopen("c:\\temp\\artikel.xml", "Artikel")=false
then
   message "Fehler beim Öffnen der XML-Datei.";
   exit script;
end

while xmleof()=false;
   message xmlread();
wend

xmlclose;
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <Artikeldaten>
  - <Artikel>
      <BestNr>Q2612A</BestNr>
      <Beschreibung>Tonerkartusche</Beschreibung>
    </Artikel>
  - <Artikel>
      <BestNr>51645A</BestNr>
      <Beschreibung>Tintenpatrone schwarz</Beschreibung>
    </Artikel>
  </Artikeldaten>
```



xmlopen

Typ: Funktion

Syntax: xmlopen(*Dateiname [, Datensatzkennung]*)

Rückgabewert: Boolean (*true* oder *false*)

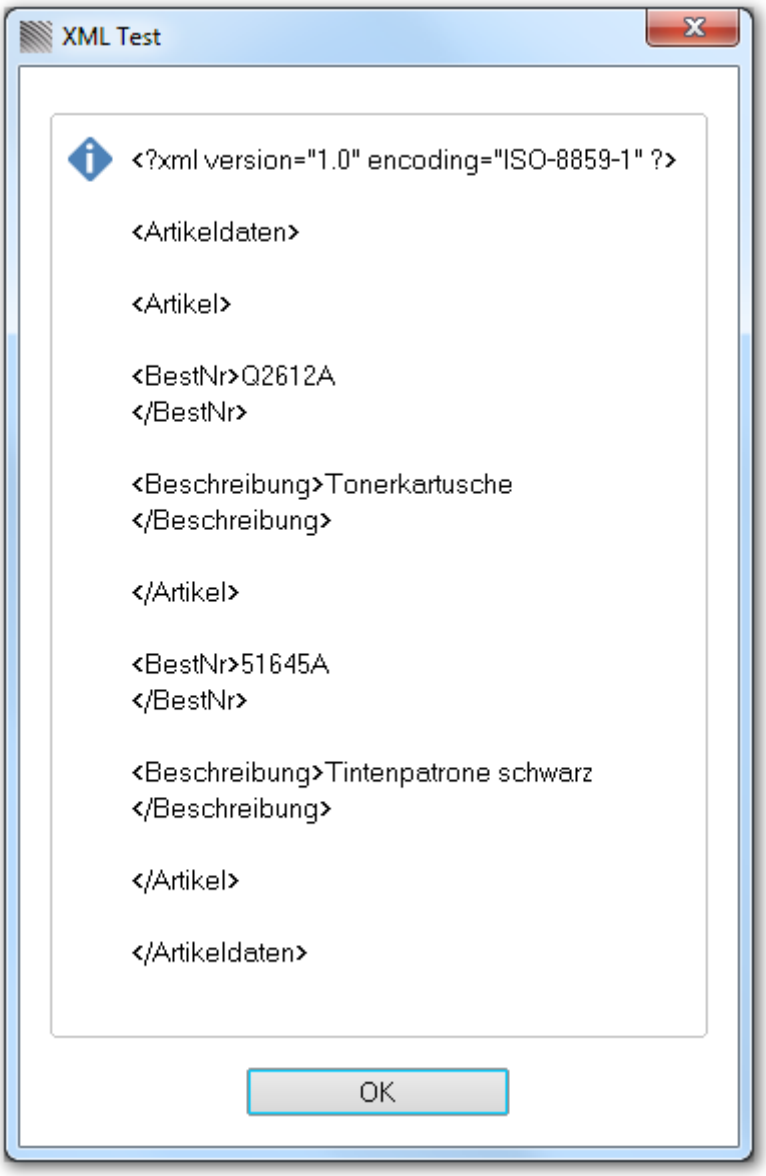
Beschreibung: Die Funktion öffnet eine XML-Datei für einen sequentiellen Lesezugriff. Sofern keine Datensatzkennung angegeben wird ließt die Funktion **xmlread** jeweils das nächste Tag samt zugehörigem Wert aus der Datei. Wird beim Öffnen eine Datensatzkennung angegeben, so wird jeweils ein ganzer Datensatz aus der XML-Datei ausgelesen.

siehe auch: xmlclose, xmleof, xmlread, xmlvalue

Beispiel 1:

```
dim buffer as text 2048;

if xmlopen("c:\\temp\\artikel.xml")=false then
   message "Fehler beim Öffnen der XML-Datei.";
   exit script;
end

while xmleof()=false;
   buffer=buffer & xmlread() & "\n";
wend

xmlclose;

message buffer;
```

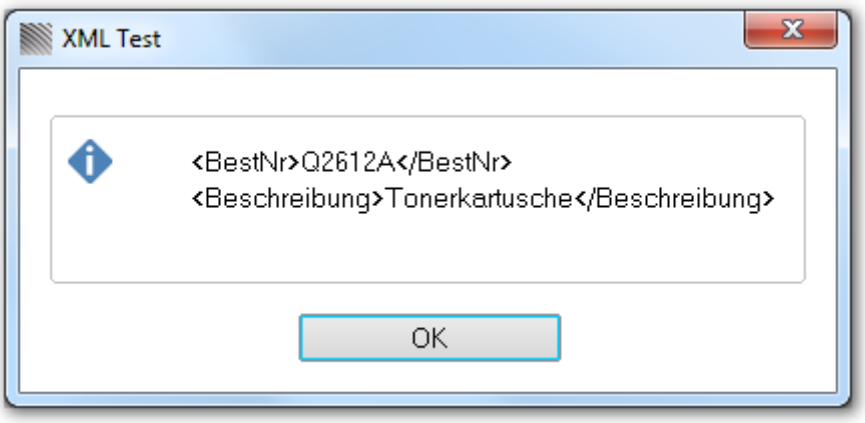
```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <Artikeldaten>
  - <Artikel>
      <BestNr>Q2612A</BestNr>
      <Beschreibung>Tonerkartusche</Beschreibung>
    </Artikel>
  - <Artikel>
      <BestNr>51645A</BestNr>
      <Beschreibung>Tintenpatrone schwarz</Beschreibung>
    </Artikel>
  </Artikeldaten>
```



Beispiel 2:

```
if xmlopen("c:\\temp\\artikel.xml", "Artikel")=false
then
   message "Fehler beim Öffnen der XML-Datei.";
   exit script;
end

while xmleof()=false;
   message xmlread();
wend

xmlclose;
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <Artikeldaten>
  - <Artikel>
      <BestNr>Q2612A</BestNr>
      <Beschreibung>Tonerkartusche</Beschreibung>
    </Artikel>
  - <Artikel>
      <BestNr>51645A</BestNr>
      <Beschreibung>Tintenpatrone schwarz</Beschreibung>
    </Artikel>
  </Artikeldaten>
```



xmlread

Typ: Funktion

Syntax: xmlread()

Rückgabewert: Eine Zeichenfolge

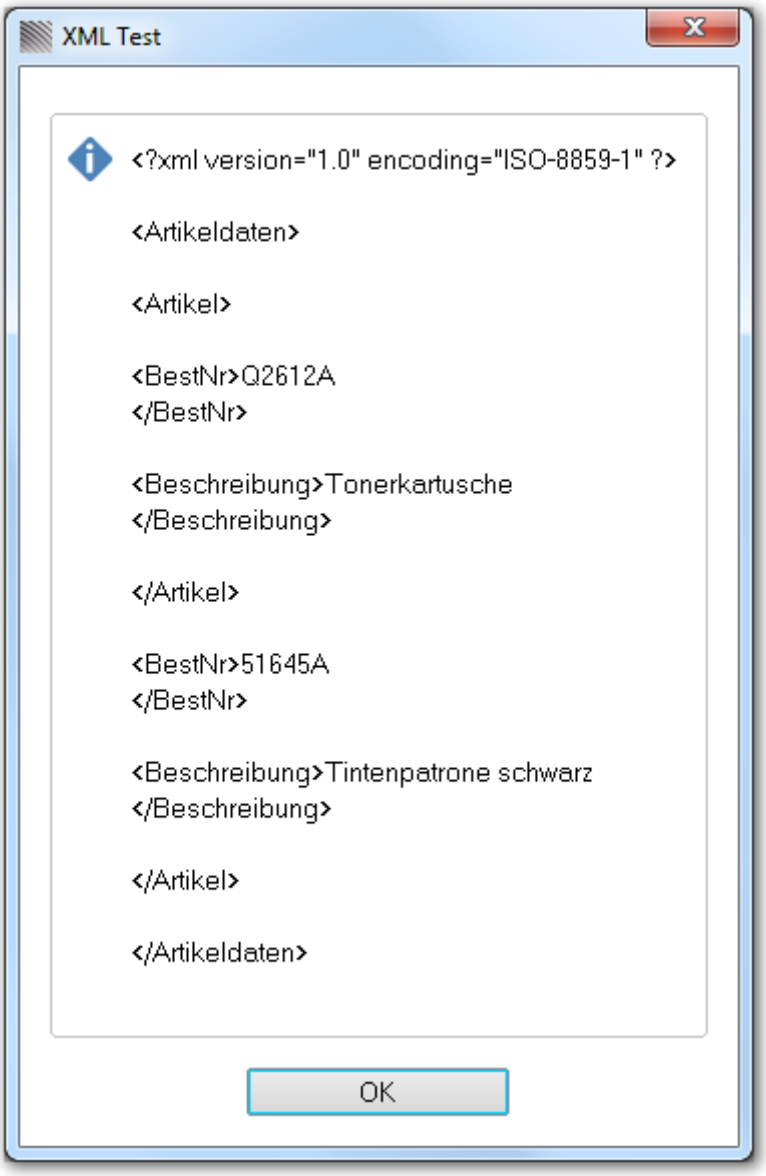
Beschreibung: Die Funktion liest das nächste Tag samt zugehörigem Wert aus einer XML-Datei. Wenn beim Öffnen der XML-Datei eine Datensatzkennung angegeben wurde, dann wird mit **xmlread** der jeweils nächste Datensatz aus der XML-Datei ausgelesen.

siehe auch: xmlclose, xmleof, xmlopen, xmlvalue

Beispiel 1:

```
dim buffer as text 2048;

if xmlopen("c:\\temp\\artikel.xml")=false then
   message "Fehler beim Öffnen der XML-Datei.";
   exit script;
end

while xmleof()=false;
   buffer=buffer & xmlread() & "\n";
wend

xmlclose;

message buffer;
```

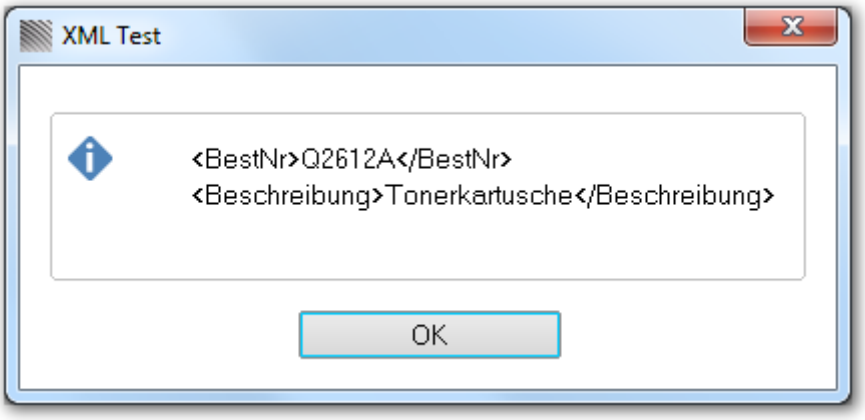
```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <Artikeldaten>
  - <Artikel>
      <BestNr>Q2612A</BestNr>
      <Beschreibung>Tonerkartusche</Beschreibung>
    </Artikel>
  - <Artikel>
      <BestNr>51645A</BestNr>
      <Beschreibung>Tintenpatrone schwarz</Beschreibung>
    </Artikel>
  </Artikeldaten>
```



Beispiel 2:

```
if xmlopen("c:\\temp\\artikel.xml", "Artikel")=false
 then
   message "Fehler beim Öffnen der XML-Datei.";
   exit script;
end

while xmleof()=false;
   message xmlread();
wend

xmlclose;
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <Artikeldaten>
  - <Artikel>
      <BestNr>Q2612A</BestNr>
      <Beschreibung>Tonerkartusche</Beschreibung>
    </Artikel>
  - <Artikel>
      <BestNr>51645A</BestNr>
      <Beschreibung>Tintenpatrone schwarz</Beschreibung>
    </Artikel>
  </Artikeldaten>
```



xmlvalue

Typ: Funktion

Syntax: xmlvalue(*Zeichenfolge, Tag [, Attribut]*)

Rückgabewert: Eine Zeichenfolge

Beschreibung: Die Funktion extrahiert den zwischen <tag> und </tag> eingeschlossenen Feldwert aus einer XML-Zeichenfolge Wenn der optionale Parameter Attribut angegeben ist, so wird stattdessen das gleichnamige Attribut aus dem XML Tag ausgelesen (verfügbar ab Build 1112).

siehe auch: xmlclose, xmleof, xmlopen, xmlread

Beispiel:

```
if xmlopen("c:\\temp\\artikel.xml", "Artikel")=false
then
   message "Fehler beim Öffnen der XML-Datei.";
   exit script;
end

while xmleof()=false;
   message xmlvalue(xmlread(), "BestNr");
wend

xmlclose;
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <Artikeldaten>
  - <Artikel>
      <BestNr>Q2612A</BestNr>
      <Beschreibung>Tonerkartusche</Beschreibung>
    </Artikel>
  - <Artikel>
      <BestNr>51645A</BestNr>
      <Beschreibung>Tintenpatrone schwarz</Beschreibung>
    </Artikel>
  </Artikeldaten>
```

obige Datei führt zu folgendem Ergebnis:

Q2612A
51645A

year

Typ: Funktion

Syntax: year(*Datum*)

Rückgabewert: Eine Ganzzahl

Beschreibung: Die Funktion bestimmt das Jahr aus einem **Datum**.

siehe auch: day, month

Beispiel:

```
year(06.01.2004)
```

```
2004
```

yearweek

Typ: Funktion

Syntax: yearweek(*Datum*)

Rückgabewert: Eine Ganzzahl

Beschreibung: Die Funktion gibt die Kalenderwoche des Datums zurück.

Beispiel:


```
yearweek(05.08.2004)
32
```

zip

Typ: Funktion

Syntax: zip(*Quelldateien(), Archivname [, Löschen] [, Passwort]*)

Rückgabewert: Boolean (*true* oder *false*)

Beschreibung: Mit dieser Funktion können eine oder mehrere Dateien zu einer komprimierten ZIP-Datei zusammengefasst werden. Der Rückgabewert ist *true*, wenn das Archiv erfolgreich erstellt werden konnte.

Parameter:

Quelldateien	einzelner Dateiname oder Datenfeld mit Dateinamen
Archivname	Name des zu erstellenden Archivs (incl. Pfadangabe)
Löschen	Quelldateien nach ZIP-Vorgang löschen (default: 'false')
Passwort	Passwort des neuen ZIP-Archivs (default: ohne)

siehe auch: sendmail, unzip

Beispiel:

```
if zip("C:\\Temp\\DataCool.exe",
"C:\\Temp\\DataCool.zip") = true then
   message "Die ZIP-Datei wurde erstellt.";
end
```

```
dim Dateien() as text 100;

Dateien(0) = "C:\\Temp\\DataCool.pdf";
Dateien(1) = "C:\\Temp\\DataCool.exe";
Dateien(2) = "C:\\Temp\\DataCool.dll";

if zip(Dateien(), "C:\\Temp\\DataCool.zip", true,
"0815") = true then
   message "Die ZIP-Datei wurde erstellt.";
end
```