Erschienen im
FileMaker Magazin
www.filemaker-magazin.de

Christian Schmitz
(Jg. 1981) entwickelt seit 1998 Software und seit 2006 auch FileMaker Plugins. Außerdem schreibt er Plugins für Xojo und allerlei Software für Mac OS X, Windows, Linux und iOS.
support@monkeybreadsoftware.de

Shop **MBS-Plugin**

Datenaustausch mit SQL

Verbindung zu einem externen Datenbankserver

Manchmal braucht Ihre FileMaker Lösung eine Verbindung zu einer anderen Datenbank, z. B. um Daten zu holen oder abzuliefern. Hierfür können Sie entweder ESS verwenden oder eine entsprechende Funktion des MBS-Plugins nutzen. Letztere möchte ich Ihnen in diesem Beitrag vorstellen.

Dabei verbinden Sie sich per Script mit einer Datenbank, führen SQL-Anweisungen aus und beenden die Verbindung anschließend wieder. So können Sie per Script Datensätze in eine Richtung verschieben oder sogar eine Synchronisationsroutine implementieren.

Alternativ können Sie ein Script einrichten, das die Verbindung beim Start der Lösung öffnet und offen hält. So besteht die Möglichkeit, verschiedene Berechnungen und Scripts auf die geöffnete Verbindung zu verweisen und Abfragen nach Bedarf auszuführen. Beispielsweise könnten Sie ein Formelfeld auf einem Layout haben. Dieses Feld zeigt einen Wert aus der externen Datenbank an. Um den Wert abzuholen, verwenden Sie die geöffnete SQL-Verbindung. Eine Lösung zur Verwaltung von Veranstaltungen könnte z. B. eine SQL-Abfrage zur Datenbank vom Webshop machen und abfragen, wie viele Tickets verkauft wurden. Was brauchen Sie also?

Datenbank-Typ

Die erste Frage ist, mit welcher Datenbank Sie sich verbinden möchten. Das **MBS-Plugin** unterstützt folgende Datenbanken:

- CubeSQL
- Centura SQLBase
- DB2
- DuckDB
- FileMaker via ODBC
- Firebird
- Informix
- InterBase
- MariaDB
- MS Access
- MS SQL Server
- MySQL
- ODBC
- Oracle DB Server
- PostgreSQL
- SQL Anywhere
- SQLite
- SQLCipher
- Sybase

Bei den meisten dieser Datenbanken kann auf **ODBC** verzichtet werden – **PostgreSQL** kann zum Beispiel direkt mit dem nativen Treiber und ohne **ODBC** angebunden werden. Auch wenn wir diesen Weg empfehlen und es eindeutig bevorzugen, **ODBC** als weitere Schicht zwischen Ihnen und dem Datenbankserver zu vermeiden, unterstützen wir auch **ODBC** als Konnektivität und können darüber eine Verbindung zu **PostgreSQL** herstellen.

Wenn Sie wissen, welche Art von Datenbank Sie haben, stellt sich die nächste Frage: Wo können Sie sie finden?

IP und Anschluss

Um eine Verbindung zu einem Server herzustellen, müssen Sie den „Connection String“ (bzw. die Verbindungszeichenfolge) kennen. Diese Zeichenfolge enthält verschiedene

Werte, die für die Verbindung benötigt werden. Der Server wird über eine IP-Adresse oder einen Domänennamen identifiziert – befindet er sich auf demselben Computer, können Sie „localhost" verwenden. Die Netzwerkverbindung zu „localhost" überspringt in der Regel die Firewall und funktioniert mit relativ hoher Wahrscheinlichkeit. Wenn Sie eine Verbindung über TCP/IP herstellen, benötigen Sie neben der IP auch eine Portnummer. Normalerweise werden Standard-Portnummern verwendet, sodass Sie diese nicht angeben müssen (beispielsweise verwendet **PostgreSQL** den Port 5432, **MySQL** 3306 oder **MS SQL Server** 1433).

Sobald Ihnen IP und Port vorliegen, können Sie versuchen, sich über unsere „Socket"-Funktionen zu verbinden. Meldet die Funktion eine Zeitüberschreitung, ist die Kombination aus IP und Port möglicherweise falsch. Wenn die Verbindung verweigert wird, liegt es wahrscheinlich an der Firewall. Im besten Fall funktioniert Ihre IP- und Port-Kombination aber und eine Verbindung wird aufgebaut.

Sie können die Richtigkeit der Angaben auch im Terminal mit dem Befehlszeilenprogramm **CURL** prüfen:

```
curl -v http://localhost:5432
```

Mit diesem Befehl wird versucht, eine Verbindung auf „localhost" zu Port 5432 für den **PostgreSQL Server** herzustellen. Wenn dort kein Server läuft oder die Firewall Sie blockiert, erhalten Sie eine der folgenden Antworten:

```
* Trying ::1:5432...
* connect to ::1 port 5432 failed: Connection refused
```

Hier meldet **CURL** auf Englisch, dass die Verbindung abgelehnt wurde. Bei einer Zeitüberschreitung wird ein Timeout angezeigt:

```
* Trying 192.168.2.222:5432...
* connect to 192.168.2.222 port 5432 failed:
Operation timed out
* Failed to connect to 192.168.2.222 port 5432:
Operation timed out
* Closing connection 0
curl: (28) Failed to connect to 192.168.2.222 port
5432: Operation timed out
```

Wurde die Verbindung erfolgreich hergestellt, erhalten Sie folgendes Ergebnis:

```
* Trying ::1:5432...
* Connected to localhost (::1) port 5432 (#0)
```

Wenn der Versuch fehlschlägt, überprüfen Sie Ihre Firewalls und ändern dort gegebenenfalls die Einstellungen, sodass der Zugriff zugelassen wird. Für **MS SQL Server** müssen Sie eventuell zuerst TCP/IP aktivieren, wenn Sie es für Verbindungen verwenden möchten.

Anmeldedaten

Nun wissen Sie, wo sich Ihr Server befindet. Als Nächstes benötigen Sie die Anmeldeinformationen, um sich zu authentifizieren.

Bitte verwenden Sie nicht Ihren Adminzugang als Login, sondern richten Sie auf dem Datenbankserver eine neue Benutzerrolle für Ihre Automatisierung ein. Dieser Benutzer sollte nur auf die Tabellen (oder Ansichten) zugreifen können, die Sie automatisieren müssen. Die meisten Tabellen sollten nur lesbar sein und bei denjenigen, die geändert werden müssen, sollten Sie nur Einfügen zulassen und Lösch-, Leeren- und Drop-Befehle verbieten. Denken Sie daran, welchen Schaden jemand in Ihrem Unternehmen anrichten könnte, wenn ihm diese Anmeldedaten in die Hände fallen. Leider sieht man viel zu viele Anwendungen, die sich einfach mit einem Administrator-Login verbinden und alles tun können, was sie wollen! Hier ein Beispiel für den neuen Benutzer:

```
Benutzername: FMAutomation
Kennwort: FileMakerIsGreat!!!
Datenbank: Personal
Abzufragende Tabellen: Person
Tabelle zum Einfügen: Stundenzettel
```

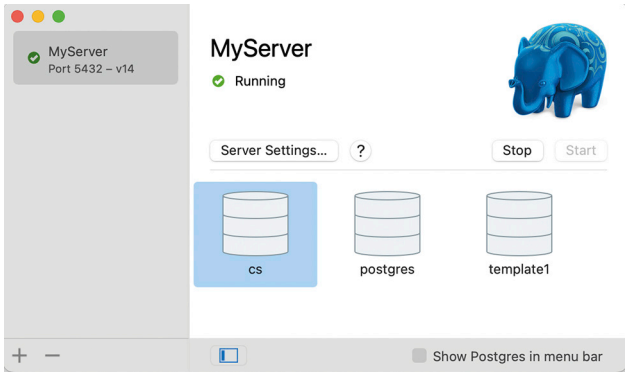
Diese FileMaker Lösung misst die Zeiten der Mitarbeiter und fügt sie dann in eine Tabelle ein. Zudem kann sie eine andere Tabelle namens ***Personal*** abfragen, um zu erfahren, wer zur Mitarbeitergruppe gehört – alle anderen Tabellen sind gesperrt. Die Tabelle ***Person*** ist natürlich für dieses Login auf „nur lesen" eingestellt.

Bibliotheken

Um eine Verbindung zu verschiedenen Datenbanken herzustellen, benötigen Sie entsprechende Treiber. Für **MySQL** ist das die Datei ***libmysqlclient***, die bei Windows **libmySQL.dll** heißt und unter macOS ***libmysqlclient.16.dylib.*** In unserem Bibliotheks-Download-Ordner[1] befindet sich eine aktuelle **MySQL 8**-Bibliothek, die Sie herunterladen können[2]. Da diese Version der Bibliothek verschlüsselte Verbindungen unterstützt, müssen die SSL- und Crypto-Bibliotheken im gleichen Ordner wie die **MySQL**-Bibliothek liegen. Sie können die drei Dateien in einen beliebigen Ordner ablegen, z. B. in den Ordner, in dem sich das Plugin befindet. Bitte beachten Sie, dass es ältere und neuere Bibliotheken geben kann und Sie manchmal eine ältere Bibliothek benötigen, weil der Server nicht auf dem neuesten Stand ist.

Für **PostgreSQL** brauchen Sie ***libssl.1.0.0.dylib, libpq.5.11.dylib*** und ***libcrypto.1.0.0.dylib.*** Die Versionen dürfen unterschiedlich aktuell sein und Windows hat natürlich DLL als Erweiterung. Wie bei **MySQL** können Sie sie an die gewünschte Stelle kopieren und das Plugin anweisen, sie zu laden. Bei **PostgreSQL** gibt es den kompletten Datenbankserver auch als App zum Download.

Befindet sich die ***Postgres.app*** beim Mac im Programme-Ordner, kann das **MBS-Plugin** die darin enthaltene Bibliothek laden, zum Beispiel „/Applications/Postgres.app/Contents/Versions/14/lib/libpq.dylib“. Der Server in der ***Postgres.app*** eignet sich hervorragend zum Ausprobieren.



Mit der App „PostgreSQL“ können Sie einen „PostgreSQL“-Server lokal laufen lassen. Großartig zum Lernen und Spielen.

Für **SQLite** können Sie die eingebaute Bibliothek des **MBS-Plugins** verwenden. Die Internet **SQLite**-Version aktivieren Sie bei Bedarf mit unserer Funktion „SQL.InternalSQLiteLibrary.Activate“. Alternativ können Sie dem Plugin auch sagen, dass es die von Ihnen benötigte Version verwenden soll.

Für den **Microsoft SQL Server** ist unter Windows keine Bibliotheksdatei erforderlich, da er unter Windows vorinstalliert ist.

Wenn Sie mit macOS arbeiten, können Sie die ***FreeTDS-Bibliothek*** laden und dem Plugin in der Verbindungszeichenfolge mitteilen, wo sie sich befindet. Da die heruntergeladenen Bibliotheksdateien eventuell nicht für macOS notarisiert sind, müssen Sie Terminalbefehle wie

```
xattr -cr /Users/cs/Documents/PostgreSQL
```

verwenden, um die Quarantäne für den Ordner zu entfernen. Danach sollten Sie in der Lage sein, die Bibliotheken zu laden.

Die Notarisierung bei Apple würden den Inhalt eines Archives auf Viren oder ähnliches durchsuchen, aber leider kann man dort nur ganze Programme einreichen und keine Bibliotheken. Daher fehlt in der Regel die Notarisierung für die Bibliotheken.

Verbinden

Um die Verbindung zum Server herzustellen, benötigen wir ein Script mit folgendem Befehl:

```
◆ Setze Variable
  [$Connection; Wert: MBS("SQL.NewConnection")]
```

Wenn Sie die Verbindung in einer globalen Variablen speichern, können Sie sie offen halten und in Berechnungen verwenden.

Bitte beachten Sie, dass die Verbindung möglicherweise vom Datenbankserver aufgrund einer Zeitüberschreitung geschlossen wird, wenn sie zu lange inaktiv ist. In diesem Fall führen Sie Ihr Verbindungsscript einfach erneut aus, wenn Sie die Datenbank weiterhin benötigen. Die Funktion „SQL.isAlive“ kann dabei helfen, den Zustand der Verbindung zu erkennen.

Als Nächstes teilen wir dem Plugin mit, wo die Bibliotheksdateien zu finden sind – das ist zum Beispiel bei MySQL unter Windows der Pfad zur DLL:

```
◆ Set Variable
  [ $r; MBS( "SQL.SetConnectionOption"; $Connection; "MYSQL.
  LIBS"; "c:\MySQL\libmySQL.dll" ) ]
```

Hier sehen Sie unterschiedliche Optionsnamen zum Einsetzen für verschiedene Datenbanktypen:

- SQLANY.LIBS
- CUBESQL.LIBS
- DB2CLI.LIBS
- DUCKDB.LIBS
- IBASE.LIBS
- INFCLI.LIBS
- MYSQL.LIBS
- ODBC.LIBS
- OCI8.LIBS
- LIBPQ.LIBS
- SQLBASE.LIBS
- SQLITE.LIBS
- SQLNCLI.LIBS
- SYBCT.LIBS
- SYBINTL.LIBS
- SYBCOMN.LIBS
- SYBTCL.LIBS
- SYBCS.LIBS

Jeder Datenbanktyp hat einen Standardpfad und wenn nichts anderes angegeben wird, verwendet das Plugin diesen Pfad. Ist die Bibliothek im Suchpfad des Betriebssystems auffindbar, benötigen Sie nur den Dateinamen, ansonsten den kompletten Pfad. Wenn Sie z. B. die **Oracle**-Client-Bibliothek installieren, können Sie den Pfad für den DLL-Ordner zur globalen Pfadliste hinzufügen und schon funktioniert es. Anschließend verbinden Sie sich mit dem Server:

```
◆ Set Variable
  [$result; Value: MBS("SQL.Connect"; $Connection;
  $SQLConnectionString; $Username; $Password; $Type) ]
```

Für **$Type** übergeben Sie den Typ der Datenbank, z. B. „PostgreSQL“. Der Verbindungsstring ist normalerweise eine Kombination aus Datenbankname, Servername und verschiedenen Optionen.

Bei **MySQL** und **PostgreSQL** kann das im Format „MyServer@MyDatabase“ erfolgen, wobei der erste Teil für die IP oder Domäne (optional mit durch Komma getrenntem Port) des Servers steht und nach dem „@“ der Name der Datenbank folgt.

Für **Microsoft SQL Server** unter Windows können Sie „MyServer\SQLEXPRESS@pubs“ übergeben – hier steht der erste Teil für den Namen der Serverinstanz, der von Windows in die tatsächliche IP aufgelöst wird. Unter macOS können Sie es mit **freeTDS** und einer Verbindung wie dieser versuchen:

```
DRIVER={" & $path & "};Server=192.168.2.32;UId=SA;
PWD=test;Database=test;TDS_VERSION=7.2;Port=1433"
```

Der Pfad zur ***freetds.dylib*** wird unter macOS im Verbindungsstring übergeben, gefolgt von Angaben zur Server-IP oder -Domäne, zum Port und zu der gewünschten TDS-Version. Wenn der Benutzername und das Kennwort Teil des Verbindungsstrings sind, müssen Sie sie nicht separat übergeben, doch in der Regel werden Passwörter dort nicht aufgenommen, um zu vermeiden, dass sie sich in einer Protokolldatei wiederfinden.

Wenn Sie **ODBC** verwenden, können Sie auch eine Datenquelle auf dem Rechner in der **ODBC**-Konfigurationsanwendung einrichten. In diesem Fall übergeben Sie einfach „ODBC“ als Typ und den Namen der Datenquelle als Verbindungszeichenfolge. Andernfalls kann ein Verbindungsstring für **ODBC** auf den Treiber verweisen und Parameter wie für **Microsoft Access** übergeben:

```
Driver={Microsoft Access Driver (*.mdb, *.accdb)};
Dbq=C:\mydatabase.accdb;"
```

Wie Sie sehen, geben wir hier den Namen des Dateipfads für die lokale Datenbank an.

Wenn **$result** „OK“ enthält, ist die Verbindung hergestellt, andernfalls können Sie einen Fehler anzeigen. Der Fehler kann zum Beispiel ein Timeout sein, wenn die Firewall die Verbindung nicht durchlässt. Oder Sie bekommen einen Fehler zur Verschlüsselung oder zur Authentifizierung oder die Bibliothek konnte aus irgendeinem Grund nicht geladen werden. In der **MBS**-Dokumentation finden Sie zahlreiche Beispiele für „SQL.Connect“ und „SQL:SetConnectionOption“ zu verschiedenen Datenbanken.

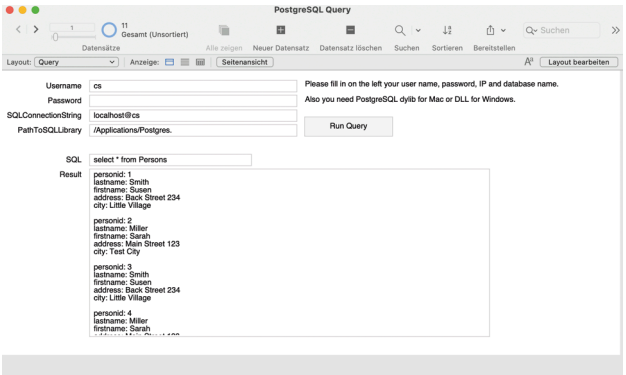
Verbindung verwenden

Nun können Sie Ihre erste Abfrage ausführen, z. B. einen „SELECT“-Befehl, um die Anzahl der aktiven Mitarbeiter des Unternehmens zu ermitteln. Um die Spalte im Ergebnis zu benennen, verwenden wir „AS“, für die „CompanyID“ nutzen wir einen benannten Parameter. Wir setzen die Werte nie direkt in das SQL ein, weil es Probleme verursacht, wenn es Anführungszeichen in den Werten gibt. Daher verwenden wir Platzhalter, die sogenannten Parameter. Diese füllen wir anschließend mit Werten. Parameter können per Nummer oder Name angesprochen werden.

```
◆ Set Variable
  [ $Command ; Value: MBS ( "SQL.NewCommand" ; $$Connection ;
  "SELECT COUNT(*) AS \"Count\" FROM \"Person\" WHERE
  \"Active\" = 1 AND \"CompanyID\" = :CompanyID" ) ]
```

Mit dem Schlüsselwort „SELECT“ startet eine SQL-Abfrage. Anschließend zählt die Funktion „COUNT()“ mit dem Sternchen die ausgewählten Datensätze. Mithilfe von „AS“ kann das Ergebnisfeld umbenannt bzw. dem Ergebnis des „COUNT()“-Aufrufs ein Name gegeben werden. Nach dem „WHERE“ folgen die Bedingungen.

Zuerst schauen wir, ob das Feld namens **Active** eine „1“ enthält und das Feld **CompanyID** den gleichen Wert hat, wie der später übergebene Parameter **CompanyID**.



Das Beispielprojekt beim MBS-Plugin kann direkt eine Abfrage in „PostgreSQL“ ausführen.

Ich bevorzuge benannte Parameter gegenüber Indexen, weil man leichter mal einen neuen Parameter einfügen kann, ohne alle Indexe zu ändern. Als Namen für den Parameter verwende ich gern den gleichen Namen wie für das Feld. Die Feldnamen werden alle in Anführungszeichen gesetzt, damit keine Probleme auftreten, falls sie ein Sonderzeichen enthalten.

Als Nächstes setzen wir den benannten Parameter „CompanyID“ auf den Text für die Unternehmenskennung (wahrscheinlich eine UUID):

```
◆ Variable setzen
  [$SetParamResult; MBS( "SQL.SetParamAsText"; $Command;
  "CompanyID"; $CompanyID ) ]
```

Anschließend können wir den Befehl ausführen:

```
◆ Variable setzen
  [$ExecuteResult; MBS( "SQL.Execute"; $Command ) ]
```

Wenn das erfolgreich war, müssen wir den ersten Datensatz des Ergebnisses abrufen, um das Ergebnis aus dem Count-Feld auszulesen. In unserem Beispiel gibt nur einen Datensatz, sodass das Ergebnis folgendermaßen abgefragt wird:

```
◆ Variable setzen
  [$FetchResult; MBS( "SQL.FetchNext"; $Command ) ]
```

Ist auch hier kein Fehler aufgetreten, kann das Feld entweder nach dem Namen:

```
◆ Set Variable
  [$Count; MBS( "SQL.GetFieldAsNumber"; $Command; "Count" ) ]
```

oder über den Index abgefragt werden:

```
◆ Setze Variable
  [$Count; MBS( "SQL.GetFieldAsNumber"; $Command; 1 ) ]
```

Anschließend geben Sie den Befehl mit „SQL.FreeCommand“ und die Verbindung mit „SQL.FreeConnection“ frei. Vergessen Sie nicht, nach jeder Zeile auf Fehler zu prüfen, z. B. mit der Funktion „MBS ("IsError")“.

Ausblick

Sobald Sie damit vertraut sind, wie man man eine Verbindung aufbaut, können Sie diese Funktionalität vielfältig benutzen. FileMaker ist keine Insel und Ihre Firma hat vermutlich noch andere Datenbanksysteme, an die Sie regelmäßig Daten per CSV oder **Excel** übertragen. Nun können Sie sich diesen Umweg sparen, da Sie sich direkt mit einem externen Server verbinden und die Daten übermitteln können.

Das **MBS-Plugin** bietet zudem einige Funktionen, um viele Datensätze direkt zu übertragen. Schauen Sie sich dafür die Funktion „FM.SQL.InsertRecordsToSQL“ an. Mithilfe von „FM.SQL.Execute“ wählen Sie zunächst einige Datensätze in FileMaker aus, zum Beispiel alle neuen Datensätze seit der letzten Synchronisierung.

Um Datensätze in FileMaker einzufügen, können Sie die SQL-Funktionen nutzen, z. B. „SQL.InsertRecords“. Auch das Aktualisieren bereits vorhandener Datensätze ist möglich, hierfür steht Ihnen die Funktion „SQL.InsertOrUpdateRecords“ zur Verfügung.

Probieren Sie gern die Beispiele beim **MBS-Plugin** und in der Dokumentation aus. In älteren Ausgaben des FileMaker Magazins von 2016 finden Sie in den Ausgaben 1 bis 4 die Artikelreihe „FileMaker und SQL “ von Jörg Wenzel zu SQL-Abfragen in FileMaker mit dem **MBS-Plugin**. Bei Bedarf führe ich die beschriebene Technik auch gern einmal auf einer Konferenz oder einem Stammtisch live vor. Sprechen Sie mich gern an. ■

Fußnoten

1) https://www.monkeybreadsoftware.com/filemaker/files/Libs/
2) z. B. libssl.1.1.dylib, libmysqlclient.21.dylib und libcrypto.1.1.dylib für macOS

FileMaker Magazin

Das FileMaker Magazin

- Einzige deutschsprachige Fachzeitschrift zu FileMaker
- Wissen aus erster Hand von anerkannten FileMaker Fachautoren
- Große Themenvielfalt für Anwender und Entwickler

Exklusiv für Premium-Abonnenten

- Sechs FMM Ausgaben pro Jahr
- Kostenlose Nutzung des Abonnentenbereichs auf ***www.filemaker-magazin.de***
- PDF-Archiv mit allen bisher veröffentlichen Ausgaben
- Jede Ausgabe mit kostenlosen Beispieldateien und Zusatzinfos zum Download

Unser Service

- Aktuelle Neuheiten, Tipps und Infos, Kleinanzeigen und vieles mehr jederzeit auf unseren Webseiten
- Hilfe bei allen Fragen zu FileMaker im FMM Forum
- Kompetente Beratung zum Kauf von FileMaker Lizenzen: Einfach anrufen +49 (0)40 589 65 79 70.

Wenn Sie sich für ein FileMaker Magazin **Abo** interessieren, klicken Sie bitte hier!

Eine kostenlose **Leseprobe** des FileMaker Magazins erhalten Sie, wenn Sie hier klicken.

Hier finden Sie Aktuelles zu **FileMaker Lizenzen**, egal ob Sie kaufen, mieten oder sich einfach informieren möchten.