Praxistipp

Holger Darjus
(Jg. 1962) Entwickelt seit 1996 FileMaker Lösungen für Prozesssteuerung, Auftragsabwicklung und Informationsmanagement im Bereich Marketing, Handel und Industrie. Besondere Ausrichtung: benutzerorientierte Gestaltung nach EN ISO 9241.
holger@darjus.de

Nachdruck aus dem FileMaker Magazin
siehe auch: www.filemaker-magazin.de

Need for Speed

Wie Sie mit wenig Aufwand die Geschwindigkeit Ihrer FileMaker Datenbanken erheblich beschleunigen können

Mitunter hat FileMaker den Ruf, eine „Datenschnecke“ zu sein. Im Zeitalter von Google und Co. fragt sich so mancher Kunde, warum eine Suchabfrage im Internet mehrere Tausend Treffer in wenigen Millisekunden liefert, eine Abfrage in der Datenbank auf einem lokal betriebenen Server aber durchaus mehrere Minuten dauern kann. In eine solche Diskussion sollte man eigentlich gar nicht erst einsteigen – und wenn doch, dann mit dem Hinweis, dass es bei Google ganze Abteilungen mit Hunderten Mitarbeitern gibt, die den ganzen Tag nichts anderes machen, als die Performance der Abfragen zu optimieren.

Allerdings können wir FileMaker Entwickler selbst eine ganze Menge dafür tun, dass unsere Datenbanken flott und geschmeidig laufen. Teilweise sind dafür nur minimale Veränderungen in Scripts oder Layouts nötig. Dabei spielt es keine Rolle, ob die Datenbank nativ mit FileMaker Pro/Pro Advanced, mit FileMaker Go oder per WebDirect genutzt wird. In diesem Artikel erfahren Sie also nicht etwa, wie Sie in dem Computerspiel „Need for Speed“ am schnellsten die Kurve kratzen, sondern was Sie bei Ihren Entwicklungen beherzigen sollten, wenn Sie die Geschwindigkeit Ihrer Datenbank optimieren wollen.

Wo steht das „Wenn“?

Beim Setzen von Variablen haben wir zwei Möglichkeiten. Der klassische Weg, der sich besser „debuggen“ lässt und den Vorteil der leichteren Lesbarkeit hat, unterliegt in puncto Geschwindigkeit der leicht modifizierten Methode, die im nachfolgenden Beispiel zu sehen ist.

Soll heißen:

```
◆ Variable setzen
    Name:    $VariableName
    Formel:  Wenn (Bedingung; Wahr; Falsch)
```

ist deutlich schneller als

```
◆ Wenn                                          ▼
    Formel:  Bedingung
◆ Variable setzen
    Name:    $VariableName
    Formel:  Wahr
◆ Sonst                                         ◆
◆ Variable setzen
    Name:    $VariableName
    Formel:  Falsch
◆ Ende Wenn                                     ▲
```

Die Geschwindigkeitsdifferenzen variieren, aber bei Tests mit mehreren Tausend Schleifendurchläufen hat sich gezeigt, dass die erste Methode um bis zu 30 % schneller ist.

„Verlasse Schleife wenn …“

Ähnlich verhält es sich beim Beenden von Schleifen. In der Regel wird vor dem Eintritt in eine Schleife ermittelt, wie oft sie durchlaufen werden soll. Mit einer kleinen Variation wird die gesamte Schleifenberechnung deutlich schneller. Nachfolgend sind beide Varianten aufgeführt:

Klassisch

```
◆ Variable setzen
   Name:    $i
   Formel:  0
◆ Variable setzen
   Name:    $n
   Formel:  5
◆ Schleife (Anfang)                    ▼
◆ Variable setzen
   Name:    $i
   Formel:  $i +1
◆ Verlasse Schleife wenn               ◆
   Formel:  $i > $n
◆ Mache irgendwas …
◆ Schleife (Ende)                      ▲
```

Schneller

```
◆ Variable setzen
   Name:    $i
   Formel:  0
◆ Variable setzen
   Name:    $n
   Wert:    5
◆ Schleife (Anfang)                    ▼
◆ Verlasse Schleife wenn               ◆
   Formel:  SetzeVars ( [$i=$i+1];$i>$n)]
◆ Mache irgendwas …
◆ Schleife (Ende)                      ▲
```

Schleife vs. „Ersetzen“

Die Funktion „Feldinhalt ersetzen“, eine überaus mächtige und damit auch sehr gefährliche Routine, kann in Script-Durchläufen ein echter Performance-Killer sein. Deutlich wird der Effekt natürlich erst, wenn wir mit vielen Datensätzen operieren. Außerdem ist die Anzahl der Felder relevant, die geändert werden sollen. Doch bereits bei einer Änderung von fünf Feldern in einer Tabelle mit 1.000 Datensätzen ist zugunsten der Geschwindigkeit eine Schleife vorzuziehen.

Klassisch

```
◆ Ersetze alle Feldwerte
   Option:  Ohne Dialog
   Name:    Feld::Name
   Option:  Berechneter Wert
◆ Ersetze alle Feldwerte
   Option:  Ohne Dialog
   Name:    Feld::Name
   Option:  Berechneter Wert
◆ ...
◆ usw.
```

Wenn wir auf diese Weise fünf Felder in 10.000 Datensätzen verändern, passiert Folgendes: Fünf Mal nacheinander werden 10.000 Datensätze geöffnet, verändert und geschrieben. Wenn die Datei auf einem Server bereitgestellt ist, muss der Server zudem allen verbundenen Clients die jeweiligen Aktionen mitteilen.

Bereits bei zehn verbundenen Clients kann man wunderbar beobachten, wie die Auslastung der CPU des Servers an ihre Grenzen stößt. Die Anwender bekommen das in Form einer „Eieruhr“ und eines überaus „klebrigen“ Verhaltens der gesamten Lösung zu spüren.

Schneller

Die Anzahl der Events lässt sich durch eine Schleife deutlich verringern. Um viele Datensätze in einem Durchgang zu verändern, kann man wie folgt vorgehen:

```
◆ Gehe zu Datens./Abfrage/Seite
   Option:  Erste(r)
◆ Schleife (Anfang)                    ▼
◆ Feldwert setzen
   Name:    Feld::Name
   Option:  Berechneter Wert
◆ Feldwert setzen
   Name:    Feld::Name
   Option:  Berechneter Wert
◆ …
◆ Gehe zu Datens./Abfrage/Seite
   Option:  Nächste(r)
   Option:  Nach letztem beenden
◆ Schleife (Ende)                      ▲
```

Auf diese Weise können zudem Fehler abgefangen werden. Konnte ein Datensatz beispielsweise nicht verändert werden, weil er von einem anderen Anwender bearbeitet wird, erhalten wir bei der „Ersetzen-Methode“ lediglich eine Meldung, dass nur 9.999 Datensätze verändert wurden und ein Fehler aufgetreten ist. Wir erfahren weder, um welchen Datensatz es sich handelt, noch bekommen wir eine Information darüber, welcher Fehler aufgetreten ist. Wurde die Option „Ohne Dialog“ aktiviert, gehen wir sogar vollkommen leer aus und können uns nicht darauf verlassen, dass alles geklappt hat.

Wem das noch nicht schnell genug ist, der verwendet am besten das **MBS-Plugin**[1] und nutzt die SQL-Funktion „UPDATE“.

Mir ist zwar völlig schleierhaft, warum FileMaker mit SQL so viel schneller ist als im nativen Umfeld, aber im Vergleich zur Funktion „Ersetze alle Feldwerte“ lassen sich derartige Operationen um den Faktor 10 beschleunigen.

Neue Fenster (Dialoge etc.)

Extreme Geschwindigkeitsgewinne können beim Öffnen neuer Fenster erzielt werden – insbesondere dann, wenn der Anwender in einem relativ komplexen und großen Layout mit vielen Feldern steht, in dem möglicherweise noch Bilder enthalten sind. Soll mithilfe eines Script-Aufrufs ein neues Fenster geöffnet werden, damit der Anwender beispielsweise eine Auswahl von Datensätzen aus einer anderen Tabelle vornehmen kann, müssen zunächst sämtliche Inhalte des aktuellen Fensters nochmals geladen werden. Zudem ist es nach dem Layoutwechsel möglicherweise nötig, die Fenstergröße und -position zu ändern und die neuen Inhalte zu laden. Ein sehr „teurer“ Prozess, bei dem man sich eventuell auch noch um Layout-Trigger kümmern muss.

Klassisch

◆ Variable setzen
Name: *$$Trigger*
Wert: *"Aus"*

◆ Neues Fenster
Name: *"Mein neues Fenster"*
Höhe: *100*
Breite: *200*
Oben: *30*
Links: *300*

◆ Gehe zu Layout
Name: *Wunschlayout*

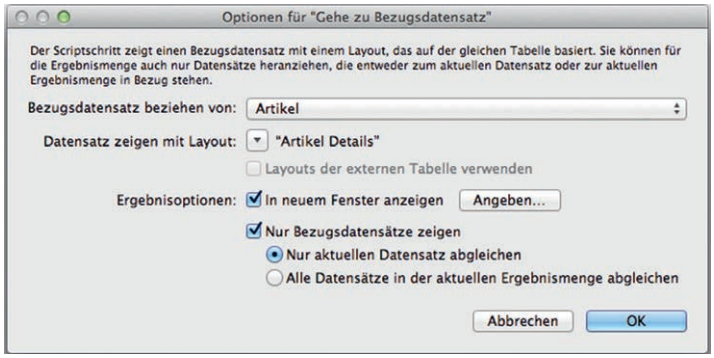
◆ Variable setzen
Name: *$$Trigger*
Wert: *"An"*

Schneller

◆ Gehe zu Bezugsdatensatz
Aus Tabelle: *"SelfJoin"*
Mit Layout: *"Wunschlayout" (Mein neues Fenster)*

◆ Fensterposition/-größe ändern
Option: *Aktuelles Fenster*

Voraussetzung für „Gehe zu Bezugsdatensatz“ ist, dass es eine gültige Beziehung zwischen der Ausgangstabelle und der Tabelle, zu der man wechseln möchte, gibt. Da eine Tabelle immer in Beziehung zu sich selbst steht, kann man einfach zur aktuellen Tabelle wechseln. Dort benötigt man lediglich ein entsprechendes Layout, das angesprungen werden soll.



Wenn man also zu jedem Tabellenauftreten, in dem sich die Anwender bewegen, ein leeres Layout anlegt, zu dem über „Gehe zu Bezugsdatensatz“ gewechselt (und möglicherweise erst dann zu einem weiteren Layout gesprungen) wird, kann extrem viel Datenfluss zwischen Server und Anwender vermieden werden. Zudem lassen sich „flackernde“ Fenster und unnötige Trigger-Aufrufe umgehen.

Platzhalter statt Feld

Warum die Verwendung von Platzhaltern die Performance verbessert, kann ich mir selbst nicht erklären. Aber auch in den „FileMaker Guidelines zur Gestaltung von Layouts für FileMaker Go“[2] wird dies deutlich empfohlen:

> *An allen Stellen, an denen Sie Feldinhalte zeigen wollen, diese von den Anwendern aber nicht bearbeitet werden müssen, verwenden Sie Platzhalter statt Felder.*

Über Platzhalter können wir Feldinhalte in Layouts anzeigen, indem der Name des gewünschten Feldes in doppelte spitze Klammern eintragen wird. Beispiel:

```
<<Adresse::Name>>
```

Im Vergleich zu Layouts mit „normalen“ Feldern werden diese Layouts um 50 oder mehr Prozent schneller geladen.

Tipp

Der Text, der in einem Platzhalter steht, kann mitunter recht lang sein und dann Probleme bei der Positionierung auf den Layouts machen. Mit einem kleinen Trick kann die Größe des Textes verringert werden, ohne dass die Darstellung davon betroffen ist. Dazu muss man lediglich wissen, dass die beiden äußeren Klammern das Textformat für die Darstellung im Layout bestimmen, während die beiden inneren Klammern das Erscheinungsbild im Layoutmodus steuern. Um einen Platzhalter mit langen Feld- oder Beziehungsnamen elegant im Layout zu positionieren, können die inneren Klammern einfach auf eine Schriftgröße von 4 pt gesetzt werden.

Wer macht den Job?

Den vermutlich größten Geschwindigkeitsgewinn können wir erreichen, wenn wir andere die Aufgaben erledigen lassen – in diesem Fall den Server. Wird die Datenbank auf einem adäquat ausgestatteten Server betrieben, kann diesem ein großer Teil der Aufgaben zugewiesen werden. Die oben angesprochene Änderung von 10.000 Datensätzen muss nicht vom Client ausgeführt, sondern kann genauso gut vom Server übernommen werden, der diese Aufgabe in der Regel sehr viel schneller erledigt.

Dazu müssen jedoch eine Reihe von Voraussetzungen erfüllt sein und der Entwickler sollte sich mit dem Verhalten von Server-Scripts auskennen. Besonders zu beachten sind dabei aufgerufene Ergebnismengen, Sortierungen und gesetzte Variablen. Weiterhin ist zu beachten, dass ein Server nicht über das volle Repertoire an Funktionen verfügt. Um Scripts für die Ausführung auf dem Server zu schreiben, ist dringend empfohlen, sich die jeweiligen Kompatibilitäten der einzelnen Funktionen anzeigen lassen.

Einen kleinen Haken gibt es außerdem noch: Wenn die Datei beim Aufruf der Funktion „Script auf Server ausführen“ nicht von einem Server bereitgestellt wird, sondern lokal auf einem Rechner liegt oder von einem anderen Rechner freigegeben ist, wird das Script nicht ausgeführt. Daher sollte vor jedem Aufruf geprüft werden, ob die Datei von einem Server bereitgestellt wird oder nicht. Hierfür eignet sich die Statusfunktion „Hole (HostAnwendungVersion)“ am besten. Ist die Datei lokal von einem Rechner geöffnet, wird ein leerer Wert zurückgegeben.

Anfangs habe ich Tasten immer so belegt, dass ich zunächst geprüft habe, ob die Datei gehostet ist oder nicht.

```
◆ Wenn                                                    ▼
   Formel:   Hole (HostAnwendungVersion) = "Server xxx"
◆ Script auf Server ausführen
   Name:     Name des Scripts
◆ Sonst                                                   ◆
◆ Script ausführen
   Name:     Name des anderen Scripts
◆ Ende (wenn)                                             ▲
```

Das kann aber auf die Dauer recht nervig werden, weshalb ich mir angewöhnt habe, Scripts möglichst so zu schreiben, dass sie auf dem Server ausgeführt werden können und der Ausführungsort im Script selbst untergebracht ist. Ruft der Anwender ein Script auf, prüft dieses zunächst, ob es auf dem Server ausgeführt werden kann. Ist das der Fall, ruft das Script sich selbst nochmals mit der Funktion „Script auf Server ausführen“ auf und verlässt den ursprünglichen Aufruf.

„MusterAnzahl“ vs. „Position“

Abschließend noch ein Druck auf das Gaspedal, wenn es darum geht zu ermitteln, ob in einem Text ein bestimmter Begriff vorkommt.

Wenn nur herausgefunden werden soll, ob ein bestimmter Suchbegriff in einem Text vorkommt und die Häufigkeit seines Auftretens irrelevant ist, empfiehlt es sich, anstelle von „MusterAnzahl“ die Funktion „Position ("Text"; "Suchtext"; Start; Auftreten)“ zu verwenden.

Diese Funktion bricht sofort ab, sobald der übergebene Suchbegriff gefunden wurde und liefert als Ergebnis einen Wert größer als Null. Das ist vor allem beim Arbeiten mit großen Textmengen ein nicht zu unterschätzender Vorteil. „MusterAnzahl“ hingegen durchsucht den gesamten Text, was bei langen Texten entsprechend lange dauern kann.

Die gleiche Zeit benötigen die beiden Funktionen übrigens, wenn der übergebene Suchbegriff gar nicht im Text vorkommt.

Fazit

Wie Sie gesehen haben, lassen sich unsere FileMaker Datenbanken bereits mit kleinen Umstellungen teilweise erheblich beschleunigen und wir können das Image von FileMaker weiter aufpolieren. Wenn zudem auf die Verwendung von ungespeicherten Formelfeldern weitestgehend verzichtet wird und Layouts so smart und übersichtlich wie möglich gestaltet werden, sind auch FileMaker Datenbanken mit mehreren Millionen Datensätzen durchaus für den Rennsport geeignet. ♦

Fußnoten

[1] *www.monkeybreadsoftware.de/filemaker bzw. www.filemaker-magazin.de/shop/produkt/338*

[2] *https://fmhelp.filemaker.com/docs/13/de/fmgo13_development.pdf*

FileMaker Magazin

Das FileMaker Magazin

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

Exklusiv für Premium-Abonnenten

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

Unser Service

- Aktuelle Neuheiten, Tipps und Infos, Kleinanzeigen und vieles mehr jederzeit online 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.

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

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

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

K&K Verlag GmbH · www.filemaker-magazin.de · Telefon +49 40 589657970