Tabellenabfragen unter LIMBAS

Jump now!

INTRO

Manchmal lassen sich Tabelleninhalte mit Hilfe von Abfragen komfortabler als in der tatsächlichen Tabellenstruktur darstellen. Dabei können Abfragenfelder aus verschiedenen im Datenbankserver vorhandene Tabellen in einer virtuelle Tabelle zusammenfügen und diese wie eine eigene Tabelle darstellen. Auf diese Weise wird bei komplexen Abhängigkeiten mehrerer Tabellen der Datenbankzugriff vereinfacht.

Realisiert werden Abfragen in LIMBAS ab Version 2.2 als Datenbank-View, die über ein SQL Statement oder mit Hilfe eines grafischen Editors erstellt werden kann. Tabelleninhalte aus Abfragen werden in der LIMBAS Benutzersicht auf die gleiche Art angezeigt wie die normalen Tabellen. Die einzige Abweichung ist, dass die bei Abfragen nicht relevante Menüpunkte ausgeblendet werden. Selbst die Detaildarstellung eines Datensatzes ist möglich, sofern nicht aktuell Änderungen bei den zugrunde liegenden Tabellen vorgenommen werden. Allerdings lassen sich über Abfragen keine Dateninhalte ändern, hinzufügen oder entfernen. Das Erstellen neuer Abfragen erfolgt in LIMBAS mit dem Abfragegenerator. Dieser startet, wenn bei den LIMBAS Tabelleneinstellungen mit der linken Maustaste auf das Editiersymbol einer Tabellen des Typs *Abfrage* geklickt wird. Er bietet zum einen die Möglichkeit, die Abfrage über eine SQL-Anweisung direkt einzugeben. Zum anderen ermöglicht er mit dem Abfrageeditor aber auch die vereinfachte Eingabe über eine grafische Darstellung und daraus dann die Generierung der SQL-Anweisung. Über die Vorschau lässt sich sofort prüfen, ob das Ergebnis den Erwartungen entspricht.

In einem Beispiel soll gezeigt werden, wie man in LIMBAS eine Abfrage erstellt. Wir wollen basierend auf einer Auftragstabelle mit Angeboten und bereits abgerechneten Aufträgen nur die Kunden anzeigen lassen, die noch offene Rechnungen zu begleichen haben. Es soll der Name des Kunden, die Anzahl der nicht beglichenen Rechnungen, die Anzahl der zugehörigen Positionen und die fällige Gesamtsumme dargestellt werden. Dazu legen wir eine neue Tabelle vom Typ *Abfrage* an. Nachdem wir diese dem LIMBAS-System hinzugefügt haben, klicken wir auf das zugehörige Editiersymbol und öffnen damit den Abfragegenerator. Um die SQL-Anweisung nicht von Hand eingeben zu müssen, wählen wir den Reiter für den Abfrageeditor. Der Arbeitsbereich des Abfrageeditors unterteilt sich in einen oberen Bereich für die Darstellung der bei der Abfrage referenzierten Tabellen und deren Verknüpfungen, sowie einen unteren Bereich mit einer Tabelle, in der die Felder der neu zu erstellenden Abfrage einzutragen sind. (Abb.1)

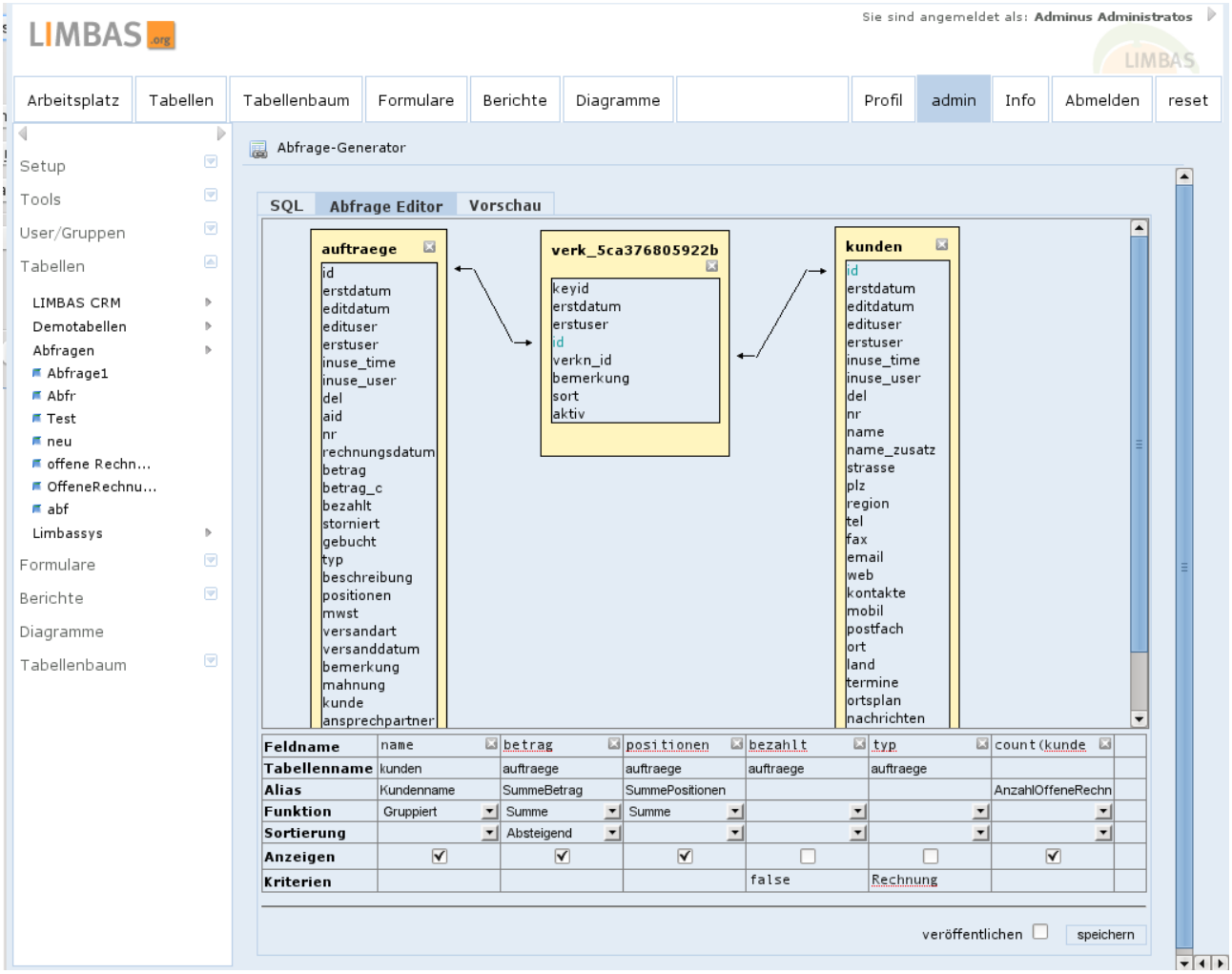

Abb.1: Abfrageeditor in LIMBAS

Wir klicken mit der rechten Maustaste in den oberen Arbeitsbereich und erhalten eine Liste mit allen momentan von LIMBAS verwalteten Tabellen und Abfragen. Für unser Beispiel wählen wir die Tabelle *auftraege*, die daraufhin als Block im oberen Arbeitsbereich eingefügt wird. Über einen Blick in die Tabelleneinstellungen der Tabelle *auftraege* ermitteln wir die Verknüpfungstabelle zu *kunden*. Dann wiederholen wir den Vorgang und fügen die ermittelte Verknüpfungstabelle sowie die Tabelle *kunden* dem oberen Arbeitsbereich des Abfrageeditors hinzu.

Als Nächstes sollen die Verknüpfungen dieser Tabellen eingebaut werden. Dazu ziehen wir das Feld *id* der Tabelle *auftraege* per Drag and Drop auf *id* der Verknüpfungstabelle. Die Verknüpfung wird durch einen in beide Richtungen zeigenden Pfeil zwischen diesen beiden Feldern dargestellt. Durch Klick mit der linken Maustaste auf eine der beiden Pfeilspitzen wird ein Fenster zum Einstellen der Verknüpfungsart (*INNER/LEFT/RIGHT JOIN*) sowie zum Löschen der Verknüpfung eingeblendet. Wir belassen die Default-Einstellung *(INNER JOIN)* und erstellen in gleicher Weise die zweite Verknüpfung zwischen dem Feld *verkn_id* der Verknüpfungstabelle und dem Feld *id* der Tabelle *Kunden*. Die als Block dargestellten Tabellen im oberen Arbeitsbereich des Abfrage-Editors lassen sich per Drag and Drop beliebig verschieben. Die Verknüpfungspfeile werden entsprechend angepasst.

Nun ist es an der Zeit, die Felder für die Abfrage anzulegen. Dabei soll der Name des Kunden an vorderster Stelle, also am linken Rand stehen. Wir ziehen das Feld *name* der Tabelle *kunden* per Drag and Drop auf die rechte freie Spalte der Tabelle des unteren Arbeitsbereiches. Daraufhin füllt LIMBAS die Zeilen *Feldname* und *Tabellenname* für das erste Abfrage-Feld entsprechend aus und fügt rechts von der ursprünglich freien Tabellenspalte eine neue wieder freie Tabellenspalte hinzu. In der Zeile *Alias* kann dem Abfragefeld eine selbst.ausgewählte Spaltenüberschrift gegeben werden. Für unser Beispiel tragen wir *Kundenname* ein. In der Zeile *Funktion* stellt LIMBAS einige Aggregatfunktionen zur Verfügung, die auf das Abfragefeld ausgeführt werden können.

Soll eine Funktion auf ein Abfrage-Feld angewandt werden, so ist zu beachten, dass auch für alle anderen anzuzeigenden Abfragefelder eine passende Funktion zu wählen ist. Ansonsten wäre die Abfrage nicht ausführbar, da es zu Inkonsistenzen bei der Anzahl der Abfrageergebnisse der verschiedenen Abfragefelder kommen könnte. Da wir in unserem Abfrageergebnis jeden Kunden nur einmal sehen möchten, wählen wir die Funktion *Gruppiert*. Wir wollen den Kunden mit den höchsten fälligen Rechnungsbetrag und nicht den im Alphabet vorrangigen als erstes sehen. Deshalb nehmen wir in der Zeile *Sortierung* des Kundennamens keine Einstellung vor. Die Einstellung der Zeile *Sortierung* wird in der LIMBAS-Benutzeransicht nur dann verwendet, wenn beim Aufruf der Abfrage keine eigene Sortierung angegeben ist. Wird bei mehreren Abfrage-Feldern die Sortierung eingestellt, so wird das Feld, dass in dieser Tabelle am weitesten links liegt, priorisiert.

Da wir den Kundennamen sehen wollen, bleibt das Häkchen der Zeile *Anzeigen* gesetzt.
Nur bei gesetzter Option *Anzeigen* wird das Feld im *SELECT* der entsprechenden SQL-Query berücksichtigt, bei nicht gesetzter Option kann dieses Feld nur für Funktionen, die Sortierung oder für Anzeigekriterien herangezogen werden. Die Zeile *Kriterien* für Bedingungen, die in der *WHERE*-Klausel des SQL-Aufrufs eingefügt werden sollen, bleibt beim Kundennamen leer.

Als nächstes wollen wir den Rechnungsbetrag, den dieser Kunde schuldig ist und die Anzahl seiner offenen Positionen, wissen, und zwar die Gesamtsumme seiner Rechnungen und Positionen. Dazu verfahren wir mit dem Feldern *betrag* und *positionen* der Tabelle *auftraege* so wie zuvor mit dem Feld *name*. Wir ziehen sie per Drag and Drop auf die rechte freie Tabellenspalte des unteren Arbeitsbereichs. Als *Alias* geben wir *SummeBetrag* und *SummePositionen* ein. Wir wählen für beide Felder die Funktion „Summe“, da die Beträge und Positionen für jeden Kunden addiert werden sollen. Um den Kunden mit den meisten Schulden ganz oben zu haben, wählen wir bei *betrag* für die Sortierung *Absteigend*. Die Option *Anzeigen* bleibt beides mal gesetzt, weitere Kriterien gibt es nicht. In unserer Abfrage sollen nur unbezahlte Rechnungen, aber keine Angebote berücksichtigt werden. Deshalb müssen wir noch die entsprechenden Bedingungen eingeben. Per Drag and Drop holen wir uns die Felder *bezahlt* und *typ* der Tabelle *auftraege* in die Tabelle des unteren Arbeitsbereichs. Diese wollen wir allerdings nicht anzeigen. Also löschen wir die Haken der Zeile *Anzeigen*. Bei *bezahlt* geben wir *false* und bei *typ* geben wir *Rechnung* in der *Kriterien*-Zeile ein. Dadurch werden nur Datensätze mit diesen Einträgen berücksichtigt.

Als letztes wollen wir noch wissen, wie viele offene Rechnungen bei einem Kunden fällig sind. Wir müssen also, für jeden Kunden, die Anzahl der berücksichtigten Datensätze

zählen. Dies erledigt die Aggregatfunktion *COUNT* für uns. Dazu tragen wir in der Zeile *Feldname* der rechten leeren Spalte von Hand *count(kunden.name)* ein. In der Zeile *Alias* schreiben wir *AnzahlOffeneRechnungen* und das Häkchen zum *Anzeigen* bleibt gesetzt. Weitere Eintragungen sind für dieses Feld nicht nötig.

Um Berechnungen, String-Manipulationen oder SQL-Funktionen auf Abfrage-Felder ausführen zu können, muss ein Eintrag von Hand in die rechte, leere Spalte der Tabelle mit den Abfragefeldern vorgenommen werden. Dabei ist in der Zeile *Feldname* ggf. die auszuführende Funktion einzutragen und es muss die referenzierte Tabelle bzw. Abfrage mit angegeben werden. Die Zeile *Tabellenname* bleibt in diesem Fall leer. Spätestens jetzt sollten wir einmal speichern. Dabei werden zum einen die Positionen der Tabellen und ihrer Verknüpfungen gespeichert.und zum anderen generiert LIMBAS aus unseren Einstellungen eine SQL-Anweisung, die man sich im SQL-Reiter ansehen kann. Um zu testen, ob wir alles richtig gemacht haben, wechseln wir zum Reiter *Vorschau* und überprüfen, ob das Ergebnis unseren Erwartungen entspricht.

Vorherige SQL-Anweisungen des SQL-Reiters werden beim Speichern im Abfrageeditor überschrieben. Beim Speichern im SQL-Reiter wird die SQL-Anweisung übernommen, die Daten für die grafische Darstellung im Abfrage-Editor werden jedoch NICHT abgeglichen. Damit die Abfrage auch in der Benutzer-Ansicht zur Verfügung steht, setzen wir im SQL-Reiter oder Abfrage-Editor das Häkchen zum veröffentlichen. Nach einem Reset können alle berechtigten Benutzer diese Abfrage im Benutzer-Menü am linken Bildschirmrand auswählen. LIMBAS zeigt das Ergebnis auf die gleiche Art wie eine Tabelle an, lediglich mit Einschränkungen bei den Menüpunkten. Will man eine Abfrage nur in einer dritten Abfrage nutzen, ist es nicht unbedingt notwendig die Abfrage zu veröffentlichen. Dazu lässt man das Häckchen *veröffentlichen* einfach leer.

FAZIT

Wie unser Beispiel zeigt, ermöglicht LIMBAS die schnelle Eingabe auch nicht trivialer Abfragen ohne programmiertechnischen Aufwand und detaillierte SQL-Kenntnisse. Diese integrieren sich nahtlos in das bisherige LIMBAS und können so auch für Berichte und Formulare genutzt werden, sowie mit Gruppenrechten versehen werden. In späteren Versionen soll es auch möglich gemacht werden Abfrageinhalte in Formularen soweit möglich editieren zu können.

AUTOR

Petra Strohmeier