Universität Rostock
Fakultät für Informatik und Elektrotechnik (IEF)
Lehrstuhl Datenbank- und Informationssysteme

Masterarbeit

Thema: Gesten-gesteuerte Anfragen auf Datenbanken

Eingereicht von: Tino Reichel
Matrikel-Nr.: 211209575
Geboren am: 08. Mai 1984
Erstgutachter: Herr Prof. Dr. rer. nat. Andreas Heuer
Zweitgutachter: Herr Prof. Dr.-Ing. Thomas Kirste
Eingereicht am: 27. Juni 2013

Zusammenfassung

In der vorliegenden Masterarbeit wird untersucht, wie moderne Interaktionstechniken genutzt werden können, um Datenbankanfragen zu erstellen. Es wird eine Interaktionsumgebung definiert, die Elemente beinhaltet, die mittels Gesten manipuliert werden können. Diese Gesten stellen die Operationen einer Datenbankanfrage dar und werden mit Hilfe moderner Interaktionstechniken formuliert. In Bezug auf Datenbankanfragen, werden hier Gesten zur Auswahl bestimmter Attribute in die Ergebnismenge (Projektion), zur Filterung der Daten an Hand von Filterkriterien (Selektion) und zum Verbinden verschiedenen Datenquellen an Hand von Verbundbedingungen (Verbund) spezifiziert. Weitere bekannte Sprachkonstrukte, wie die Nutzung von Mengenoperationen (Vereinigung, Durchschnitt, Differenz) oder die Anwendung von Aggregatfunktionen, sind ebenso Bestandteil der Untersuchung innerhalb dieser Arbeit.

Zur Einführung werden die Aufgabenstellung der Arbeit, die Zielsetzung und die Abgrenzung beschrieben. Hierbei werden die persönlichen Ziele des Autors definiert und bestimmte Grundvoraussetzungen für die Leser dieser Arbeit festgelegt.

Der Hauptteil des Dokumentes beschäftigt sich mit der Erarbeitung einer Spezifikation, die dazu dient, die grundlegenden Datenbankoperationen der mittels Gesten auf modernen Interaktionsmedien zu definieren. Im Anschluss erfolgt die Umsetzung der Spezifikation in einen Prototyp. Hierbei wird kurz auf die eingesetzte Technologie bei der Umsetzung eingegangen. Abschließend wird die Funktionsweise des Prototyps an einem Beispiel demonstriert.

Im Fazit der Arbeit wird abschließend ein kurzer Vergleich zur Anfragesprache QBE gezogen. Des Weiteren erfolgt ein Ausblick auf die Zukunft der Spezifikation und deren Einsatzmöglichkeiten.

Abstract

This thesis is concerned with the analysis of modern interaction techniques used to create database queries. An interaction surface will be defined, containing the elements and components which could be manipulated by gestures. These gestures represent the operations of a database query and are formulated using advanced interaction techniques. In terms of database queries, gestures are specified for selecting specific attributes (projection), to filter the data on the basis of some filter criteria (selection) and to connect different data sources on the basis of bond-conditions (join). Additional well known operations, such as set-operations (union, intersection, difference) or the use of aggregate-functions, are also part of the research in this thesis.

The introduction contains the task-description, the goal of this work and the delimitation. It includes the definition of the personal goals of the author and certain basics for the readers.

The main part of this document is about the development of a specification that is used to define the basic database operations using gestures on modern interaction-media. An implementation of the defined specification is followed and carried out in a prototype. It also contains a brief look into the used technology. Finally, the operation of the prototype is demonstrated by an example.

In conclusion, a brief comparison to the query-language QBE is documented. Furthermore, we will take a look at the future of this specification and their possible operational scenarios in applications.

Inhaltsverzeichnis

1 Einführung

1.1 Motivation

Bei der Formulierung einer Datenbankanfrage hat sich die Anfragesprache SQL bis heute behauptet. Man nutzt einen Texteditor und erzeugt mit Hilfe der Tastatur, und unter Einhaltung der vorgegeben SQL-Syntax, eine Anweisung, die nach der Ausführung ein Ergebnis in Form einer Tabelle zurückliefert.

In den letzten Jahren haben sich die Eingabemöglichkeiten dahingehend erweitert, dass man statt einer Tastatur oder einer Computermaus, einen Finger nutzen kann. Mit diesem Finger zeigen wir auf bestimmte Elemente, schieben sie weg oder zoomen bestimmte Bereiche heran. Beispielsweise können wir heutzutage, statt auf mechanische Tasten auf unserem Handy zu tippen, eine Geste formulieren, die für das gewünschte Wort steht [SWY13]. Dies ist durch die modernen Interaktionstechniken wie beispielsweise der Touchscreen-Technologie möglich.

Diese jungen Interaktionsmöglichkeiten sollen auch bei der Erstellung von Datenbankanfragen Anwendung finden, so dass man nicht mehr zwingend die Tastatur und die Computermaus benötigt, um ein Ergebnis einer solchen Anfrage zu erhalten. Demzufolge muss untersucht werden, wie man diese neue Technik nutzen und in Bezug auf die Erstellung von Datenbankanfragen in einer Spezifikation definieren kann.

1.2 Aufgabenstellung

Ziel der Masterarbeit ist die Untersuchung moderner Interaktionsmöglichkeiten beim Stellen und Verarbeiten von DB-Anfragen. Dabei soll insbesondere die Interaktion mit den Geräten per Gestensteuerung untersucht werden. Diese findet sich in jedem Smartphone oder iPad per Touch-Display oder in vom Spielemarkt eingeführten günstigen 3D-Kameras. Mit Gestik werden dabei verschiedene Interaktionsmöglichkeiten bei der Bedienung der Geräte selbst und in Applikationen angeboten, z.B. bei der Bildverarbeitung, der Kartendarstellung oder im Spielebereich. Die Frage in dieser Arbeit ist, ob man mit Gestensteuerung auch eine komplexe Anfragesprache, wie SQL „simulieren“ kann. Im Bereich der Datenbanken sind Operationen wie Projektion, Selektion, Verbund etc. wesentlich für die Anfragestellung bzw. das Formulieren des Informationsbedürfnisses. Eine gestenbasierte Anfragesprache könnte die Akzeptanz und Nutzerfreundlichkeit verbessern. Solche Anfragemechanismen sind bereits aus früheren Ansätzen, wie bspw. QBE – Query by Example – bekannt.

In der Arbeit sind insbesondere die grundsätzlichen Datenbankoperationen, also der Relationenalgebra für die Gestensteuerung zu prüfen. Dabei könnten Operationen eins zu eins, zusammengefasst oder neu definiert werden – z.B. das Ziehen und Überlappen von Tabellen bei der Deklaration eines Verbundes mit Verbundattribut mittels Drag & Drop-artiger Geste oder die Projektion durch Tippen auf die benötigten Attribute. Besonders interessant erscheint auch die mögliche Verwendung der 3. Dimension oder von mehreren

Fingern sowie 2 Händen bzw. Armen. Bestimmte Anfragekonstrukte können nicht ausschließlich durch Gestik beschrieben werden. Hier ist es nötig, ein sinnvolles Zusammenspiel zwischen neuartiger Gestik und herkömmlicher Eingabemöglichkeit zu entwickeln.

1.3 Zielsetzung

Mit dieser Arbeit sollen natürliche Interaktionsmöglichkeiten auf Grundlage einer Gestensteuerung zur Datenbankanfrage untersucht werden. Hierbei liegt der Fokus auf der Untersuchung möglicher Gesten im zwei dimensionalen Raum.

Es sollen die grundlegenden Datenbankoperationen, wie beispielsweise der Verbund, die Projektion und die Selektion mit einer Geste, oder einem Ablauf von Gesten, spezifiziert werden. Diese Gesten sollen in einer Interaktionsumgebung formulierbar sein und wenn möglich die Kriterien der Anfragesprachen für Datenbanksysteme [SSH13:94 - 95] weitestgehend erfüllen. Das Dokument soll zudem eventuelle Probleme und Problemlösungen, die bei der Erstellung der Spezifikation auftraten, aufzeigen.

Nach der Spezifikation der Operationen (Gesten), soll außerdem ein Prototyp entwickelt werden, der einige der entwickelten Konzepte in einer Demonstrationsanwendung darstellt.

1.4 Abgrenzung

Die Aufgabenstellung beinhaltet zusätzlich zur Erforschung der Interaktionstechniken im zwei dimensionalen Raum, ebenso die Berücksichtigung des drei dimensionalen Raums mit Hilfe einer 3D-Kamera. Auf Grund des Umfangs der Arbeit und der zeitlichen Begrenzung zur Erforschung des Themas, beziehen sich die im Dokument entwickelten Definitionen und Algorithmen auf die Interaktion im zwei dimensionalen Raum (Touch-Display), wobei die Entwicklung aller Konzepte, Definitionen und Algorithmen, sowohl die Bedienung mittels Touch-Display, als auch mittels Gestenerkennung über eine 3D-Kamera, berücksichtig.

In dieser Arbeit werden die grundlegenden Konzepte und Operatoren der relationalen Algebra, sowie die entsprechende SQL-Syntax in eine andere, für Gesten optimierte, Form überführt. Es werden weder vollständige Definitionen, noch sämtliche Operationen der relationalen Algebra sowie der Anfragesprache SQL in den einzelnen Kapiteln beschrieben. Diese Arbeit beinhaltet die Konzeption bekannter Datenbankoperationen und die Berücksichtigung der damit verbundenen Definitionen und Regeln, so dass eine gestenbasierte Anfragesprache auf dieser Grundlage die Akzeptanz und Nutzerfreundlichkeit der Anwender steigert.

2 Stand der Technik

2.1 Anfragesprachen

Anfragesprachen werden für das Suchen und Ändern des Datenbestandes einer Datenbank benötigt und sollten den Kriterien zur Ad-hoc-Formulierung, Deskriptivität, Mengenorientiertheit, Abgeschlossenheit, Adäquatheit, Orthogonalität, Optimierbarkeit, Effizienz, Sicherheit, Eingeschränktheit und Vollständigkeit genügen. [SSH13:94 - 111]

Unter Berücksichtigung dieser Kriterien und unter Betrachtung der nachfolgenden Anfragesprachen wird im vorliegenden Dokument eine neue Anfragesprache spezifiziert. Zu beachten ist, dass im Zuge dieser Arbeit keine neue Sprach-Syntax und demzufolge auch kein Parser entwickelt wird. Somit werden die letztendlichen Anweisungen in SQL übersetzt, so dass entsprechende Datenbanksysteme diese generierten Anweisungen ausführen können.

2.1.1 SQL

SQL ist eine standardisierte Datenbanksprache für relationale Systeme. SQL dient nicht nur als Anfragesprache, sondern ebenso zur Datendefinition. Zudem beinhaltet die Spezifikation auch Operationen zur Datenänderung. SQL basiert auf Anweisungen in Textform (textuelle Sprache) und eignet sich daher nicht als direkte mit Gesten gesteuerte Anfragesprache. [SSH13:211 - 247]

2.1.2 QUEL

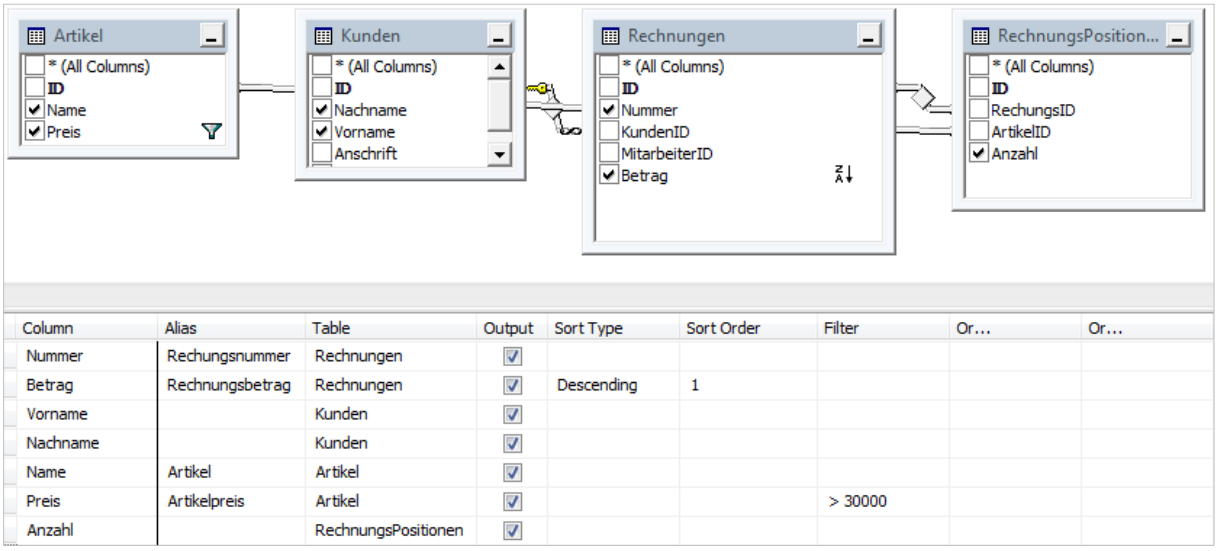
QUEL ist, wie SQL, eine Datenbanksprache und ein quasi Konkurrent von SQL, der sich jedoch nicht gegen SQL behaupten konnte. QUEL basiert ebenso auf Anweisungen in Textform und unterscheidet sich hauptsächlich in der Syntax. Demzufolge eignet sich QUEL auch nicht als direkte mit Gesten gesteuerte Anfragesprache. [SSH13:356 - 359]

2.1.3 Query by Example - QBE

„Die Sprache QBE (Query by Example) ist im Gegensatz zu SQL oder QUEL keine textuelle Sprache. Anfragen werden in QBE durch Einträge in Tabellengerüsten formuliert, wobei die Reihenfolge der Einträge oder die Anordnung auf dem Bildschirm keine Rolle spielen. Die zugrundeliegende Intuition wird durch die Wahl der englischen Langform von QBE angedeutet: Der Benutzer soll Beispieleinträge in Tabellen angeben, die seinen Anfragewünschen entsprechen.“ [SSH13:359]

Um Datenbankanfragen, ohne konkretes Wissen einer Anfragesprache zu besitzen, erstellen zu können, existiert QBE. Der Benutzer hat die Möglichkeit, eine Datenbankanfrage mit Hilfe eines Tabellengerüstes zu definieren. Hierbei werden alle Spalten der Tabelle in das Gerüst geladen. Der Nutzer kann anschließend pro Spalte verschiedene Filterbedingungen setzen,

die je nach Spaltentyp eine Auswahl oder eine Freitexteingabe bereitstellen. Ein Beispiel eines QBE-Editors ist in Abbildung 1 dargestellt.



Column	Alias	Table	Output	Sort Type	Sort Order	Filter	Or...	Or...
Nummer	Rechungsnummer	Rechnungen	☑					
Betrag	Rechnungsbetrag	Rechnungen	☑	Descending	1			
Vorname		Kunden	☑					
Nachname		Kunden	☑					
Name	Artikel	Artikel	☑					
Preis	Artikelpreis	Artikel	☑			> 30000		
Anzahl		RechnungsPositionen	☑					

Abbildung 1 - Stand der Technik: QBE-Editor

Der Anwender kann hier beispielsweise im Tabellengerüst in Spalte *Column*, eine Spalte einer zuvor platzierten Relation (*Artikel, Kunden, Rechnungen* oder *RechnungsPositionen*) auswählen. In den weiteren Spalten kann nun ein Alias, eine Sortierung, eine Gruppierung und/oder eine oder mehrere Selektionsbedingungen (Filter) definiert werden.

Die vorhandenen Interaktionselemente, wie Menüs, Checkboxen, Eingabefelder, Kontext-Menüs und Auswahllisten sind für die Bedienung mit der Tastatur und der Maus optimiert. Die Bedienung mit einem Touch-Display und der beispielhaften Auswahl einer neuen Spalte im Tabellengerüst führt, auf Grund der kleinen Darstellung und der physisch bedingten ungenauen Bedienung mit einem Finger, zu Fehleingaben und somit zu nicht Akzeptanz des Anwenders.

Um dieses Problem lösen und zudem eine Möglichkeit schaffen zu können, eine Datenbankanfrage mit 3D-Gesten zu erstellen, wird eine neues Interaktionskonzept mit, für diese Anwendungsfälle, optimierten Interaktionselementen benötigt.

Die Grundlage der zu entwickelnden Gesten bildet eine von bekannten QBE-Anwendungen[1] adaptierte Interaktionsumgebung. Abbildung 2 illustriert die QBE-Umgebung Microsoft SQL-Server Management Studio.

[1] Microsoft Access, Microsoft SQL Server Management Studio

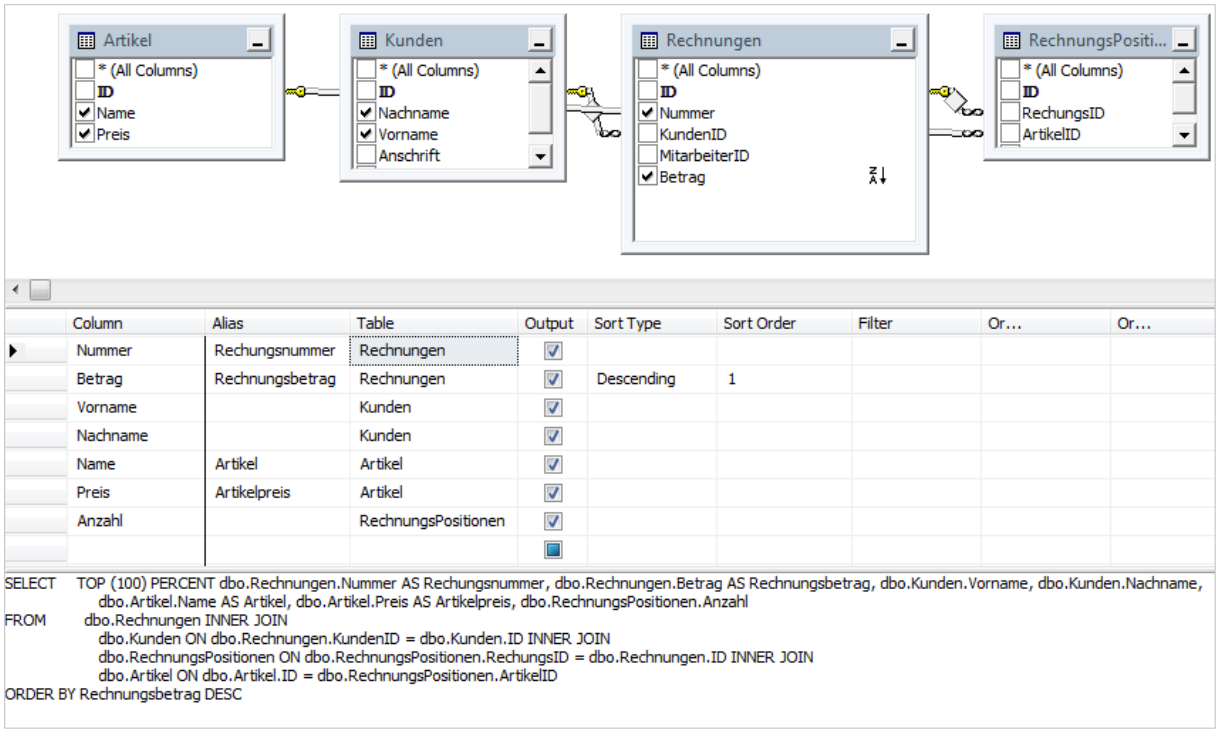


Abbildung 2 - QBE: Microsoft SQL Server 2008 Managment Studio

Wie in den meisten QBE-Umgebungen ist die Oberfläche dreigeteilt.

Im Oberen Teilbereich (Abbildung 3) werden dem Anwender die gewählten Relationen und ihre Beziehungen zueinander präsentiert. Der Benutzer kann die Attribute an- oder abwählen und die Beziehungen mit Hilfe von Maus-Gesten löschen oder hinzufügen.

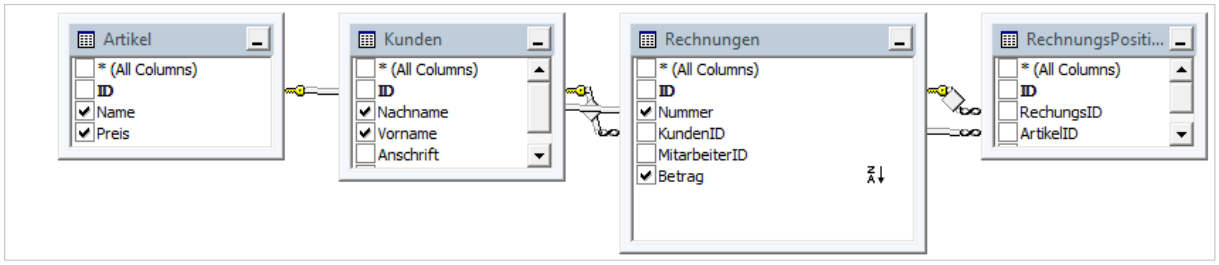


Abbildung 3 - QBE: Teilbereich 1

Der mittlere Teilbereich (Abbildung 4) stellt ein Tabellengerüst zur Verfügung, in dem der Anwender verschiedene Bedingungen zur Selektion, Projektion und Sortierung definieren kann.

	Column	Alias	Table	Output	Sort Type	Sort Order
▶	Nummer	Rechungsnummer	Rechnungen	☑		
	Betrag	Rechnungsbetrag	Rechnungen	☑	Descending	1
	Vorname		Kunden	☑		
	Nachname		Kunden	☑		
	Name	Artikel	Artikel	☑		
	Preis	Artikelpreis	Artikel	☑		
	Anzahl		RechnungsPositionen	☑		
				☐		

Abbildung 4 - QBE: Teilbereich 2

Der untere Teilbereich (Abbildung 5) ermöglicht sowohl die Formulierung der Anfrage in SQL, als auch die Anzeige der SQL-Anweisung, die aus der Definition der Bedingungen aus den Teilbereichen 1 und 2 entstanden ist.

```
SELECT    TOP (100) PERCENT dbo.Rechnungen.Nummer AS Rechungsnummer, dbo.Rechnungen.Betrag AS Rechnungsbetrag, dbo.Kunden.Vorname, dbo.Kunden.Nachname,
                dbo.Artikel.Name AS Artikel, dbo.Artikel.Preis AS Artikelpreis, dbo.RechnungsPositionen.Anzahl
FROM      dbo.Rechnungen INNER JOIN
                dbo.Kunden ON dbo.Rechnungen.KundenID = dbo.Kunden.ID INNER JOIN
                dbo.RechnungsPositionen ON dbo.RechnungsPositionen.RechungsID = dbo.Rechnungen.ID INNER JOIN
                dbo.Artikel ON dbo.Artikel.ID = dbo.RechnungsPositionen.ArtikelID
ORDER BY Rechnungsbetrag DESC
```

Abbildung 5 - QBE: Teilbereich 3

Diese Interaktionsumgebung wird in den Kapitel 3 und 4 in eine für Touch- und 3D-Gesten optimierte Form überführt. Bekannte Funktionsweisen aus QBE, wie die Auswahl der Relationen, die Auswahl der Spalten und die Formulierung von Selektionsbedingungen, werden adaptiert.

Auf Grundlage der Touch- und 3D-Gesten-Optimierung wird ein anderes Interaktionskonzept als bei QBE verwendet. Dies beinhaltet die Konzeption einer neuen Interaktionsumgebung. Hier wird nach Anwendungsstart ein vorerst leerer Bereich dargestellt - Der in Kapitel 3.1 vorgestellte Interaktionscontainer.

2.1.4 Weitere Anfragesprachen

Es existieren weitere Anfragesprachen, wie beispielsweise Datalog und Tutorial D. Hier werden Konzepte von rekursiven Anfragen verfolgt, oder auch fehlende Funktionen in SQL durch eigene Spezifikationen vorgestellt. [SSH13:373 - 385]

2.2 Hardware

In diesem Abschnitt werden verschiedene Hardware-Ressourcen zur Gestenerkennung genannt. Es werden sowohl Geräte, die mit dem Finger bedient werden können, als auch 3D-Sensoren vorgestellt, die für den Einsatz zur Gestenerkennung nutzbar sind.

2.2.1 Displays

Nachfolgend werden zwei Touchscreens vorgestellt, die bei der Anfertigung der Arbeit genutzt wurden. Der Markt bietet unterschiedlichste Touch-Displays an, die sich in verschiedenen Multi-Touch-Technologien unterscheiden. Hier erfolgt nur eine kurze Darstellung zweier Technologien an Hand von zwei ausgewählten Produkten.

2.2.1.1 IIYAMA PROLITE T2452MTS

Der 24 Zoll Touchscreen IIYAMA PROLITE T2452MTS [IIY13] basiert auf einer optischen Berührungserkennung [IAO12:16 - 18]. Das heißt, es werden eine Reihe von Lichtquellen und Sensoren im Monitorgehäuse genutzt, um Berührungen auf dem Bildschirm erkennen zu können. Der Vorteil dieses Gerätes, ist der im Vergleich günstige Preis von ca. 250€. Der Nachteil ist hingegen, dass nur maximal zwei Berührungspunkte erkannt werden können.

Dieses Gerät diente in der ersten Entwicklungsphase des Prototyps als Eingabegerät der Gesten.

2.2.1.2 LG 23ET83V

Der von der Universität Rostock zum Test zur Verfügung gestellte LG 23ET83V [LG13], ist ein 23 Zoll Touchscreen und nutzt als Berührungserkennungstechnologie ein kapazitives Verfahren [IAO12:11 - 12]. Die Berührungserkennung erfolgt hier an Hand von Spannungsänderungen unterhalb der Druckpunkte. Je nachdem, wie viele der Sensoren verbaut wurden, lassen sich mehr als zwei (zehn im Fall des LG 23ET83V) Druckpunkte erkennen. Somit ist der Vorteil dieses Gerätes die mögliche Nutzung von zehn gleichzeitigen Berührungspunkten. Der Nachteil ist hier der mit ca. 400€ vergleichsweise hohe Preis des Gerätes.

Dieses Gerät wurde in der späten Testphase des Prototypen eingesetzt.

2.2.2 3D-Kameras

Bis zum Jahr 2010 wurden 3D-Kameras zur Erforschung der Möglichkeiten der Gestenerkennung hauptsächlich an Universitäten und Forschungseinrichtungen eingesetzt, da die verfügbaren Produkte durch ihren hohen Preis den Käuferkreis einschränkten. Dies änderte sich mit der Einführung der *Microsoft Kinect* im Jahr 2010.

2.2.2.1 Microsoft Kinect

Dieses Produkt, das ursprünglich nur für die Spielekonsole Xbox 360 entwickelt wurde, ermöglicht an Hand von verschiedenen Sensoren die Aufnahme von akustischen sowie räumlichen Signalen. [WF12:5 - 6]

Durch ein zur Verfügung gestelltes SDK [KIN13] ermöglicht Microsoft seit Januar 2012 eine komfortable Auswertung der Sensordaten auf einem PC.

Im Zuge der Veröffentlichung der *Microsoft Kinect* entstanden verschiedene Open Source Projekte, unter anderem auch das Projekt *OpenNI* [ONI13]. Die Entwicklungsgemeinde beschäftigt sich, losgelöst von dem von Microsoft zur Verfügung gestellten SDK, in diesem Projekt mit der Entwicklung von Frameworks und Tools zur Verarbeitung der Sensordaten. Diese Entwicklungen basieren auf den, auch von der *Microsoft Kinect* genutzten Sensoren des Unternehmens *Prime Sense*, wodurch die Nutzung dieser SDK's und Tools auch mit der *Microsoft Kinect* möglich ist.

2.2.2.2 Microsoft Kinect for Windows

Ein weiteres Produkt ist die *Microsoft Kinect for Windows*. Hierbei handelt es sich um einen, speziell für die Gestenerkennung am PC entwickelten Sensor. Die Vorteile gegenüber der *Microsoft Kinect* liegen hauptsächlich in der Unterstützung der Nutzung am Computer und nicht vor der Spielekonsole. [KIN13]

Die Auswertung der Sensordaten ist auch hier mit den in Kapitel 2.2.2.1 vorgestellten SDK's möglich.

2.3 Verwandte Arbeiten

Die nachfolgenden Unterkapitel beinhalten Referenzen zu verwandten Arbeiten, die sich im Teilbereich der Gestenerkennung mit ähnlichen Fragen der vorliegenden Arbeit beschäftigen.

2.3.1 „Query-by-Gesture: An Alternative Content-Based Image Retrieval Query Scheme“

In dem Artikel [QBG02] stellen die Autoren ihre Idee zum content-based image retrieval (CBIR[2]) vor. Es geht um die Frage, wie man eine bestimmte Anfrage auf eine Multimedia-Datenbank mit Hilfe von 3D-Gesten formulieren kann. Hier wird die die Methode Query-by-Sketch in den 3 dimensionalen Raum abstrahiert. Mittels Geste entsteht eine Zeichnung/Skizze (Sketch) im Raum, die anschließend vom Datenbanksystem mittels Ähnlichkeitsvergleich entsprechende Ergebnisse zurückliefern kann.

Im vorliegenden Dokument werden die Formulierungen der Gesten nach dem Prinzip Query-by-Sketch auf Touch-Displays und nach dem hier erwähnten Prinzip Query-by-Gesture mittels 3D-Sensor zu dem Oberbegriff Query by Gesture (QBG) zusammengefasst.

Im Gegensatz zu der Idee, die formulierten Gesten in eine Skizze für einen Bildvergleich zu wandeln, beschäftigt sich diese Arbeit, unter dem Thema QBG, mit der Erkennung der Gesten und der anschließenden Zuordnung zu einer Datenbankoperation.

2.3.2 „Gesture-Based Navigation in Graph Databases – The Kevin Bacon Game –“

In dieser Arbeit [BBBH13] stellen die Autoren ein System zur Navigation innerhalb eines Datenwürfels vor. Es werden verschiedene 3D-Gesten für die Operationen innerhalb des Würfels spezifiziert. Einige der hier genutzten Gesten (zum Beispiel: Zoom und Swipe/Wischen) werden in der vorliegenden Arbeit ebenso genutzt, um mit ihnen Datenbankoperationen abzubilden.

Da sich die vorliegende Arbeit hauptsächlich mit der Formulierung von Datenbankanfragen beschäftigt, und nicht mit der Navigation innerhalb eines Graphen, grenzt sich das vorliegende Dokument von der in diesem Kapitel erwähnten Publikation ab.

[2] content based image retrieval – [MH11]

2.3.3 „Konzeption und Umsetzung eines Gestenerkennungssystems für Intelligente Umgebungen auf Basis von 3D-Kamerasensorik“

In der Projektarbeit mit dem Titel „Konzeption und Umsetzung eines Gestenerkennungssystems für Intelligente Umgebungen auf Basis von 3D-Kamerasensorik“ von Fiete Winter [WF12], wird das Ziel verfolgt, intelligente Umgebungen mit Hilfe einer Gestenerkennung zu steuern.

Der Autor beschreibt hier unter anderem ein Model zur Merkmalsextraktion – das Hidden Markov Model [WF12:28 - 30]. Eine Implementierung [ACC13] dieses Models wurde bei der Erstellung des Prototyps ausgewählt, um die Gesten für die Vergleichsoperatoren erkennen zu können.

2.3.4 „GestureQuery: A Multitouch Database Query Interface”

Einen, im Vergleich zum vorliegenden Dokument, ähnlichen Ansatz verfolgen die Autoren Lilong Jiang, Arnab Nandi und Michael Mandel in Ihrem YouTube-Video [JNM13]. Herr Arnab Nandi beschreibt die in dem Video zu sehenden Gesten in seinen Publikationen [NA13_1] und [NA13_2]. Hier werden ebenso Datenbankanfragen an Hand von Gesten formuliert. Der Autor nutzt zur Navigation und Darstellung der Ergebnisse eine tabellarische Ansicht, wobei sich die vorliegende Arbeit auf ein anderes Konzept stützt, welches die Nachteile der tabellarischen Ansicht abschafft.

3 Spezifikation der Interaktionselemente

3.1 Der Interaktionscontainer

Der Interaktionscontainer ist die Manipulationsoberfläche der Anwendersoftware. Analog dem Zeichenbereich eines Grafikprogrammes, stellt sie einen Container zur Aufnahme und Manipulation von Inhalten dar. Diese Inhalte sind hier unter anderem die Tabellen einer Datenbank. Somit definiert der Interaktionscontainer eine leere Datenbank, die mit dem Positionieren von Tabellen auf diesen Bereich, ihre Inhalte erweitert.

Zu Beginn steht der Entwurf des Containers, der die Funktionalität des QBE-Editors aus Abbildung 1 bietet. Den Rahmen bildet ein rechteckiges Element (- Grau schraffierter Bereich). Dem Anwender wird zudem eine Liste an Relationen zur Verfügung gestellt, die auf Anforderung über dem Interaktionscontainer eingeblendet wird. Über diese Liste kann der Benutzer per Geste die gewünschten Relationen auswählen und auf dem Interaktionscontainer positionieren. Abbildung 6 illustriert dieses Vorgehen, indem die Relation *Artikel* aus der Liste ausgewählt und per Drag & Drop auf dem Manipulationsbereich abgelegt wird.

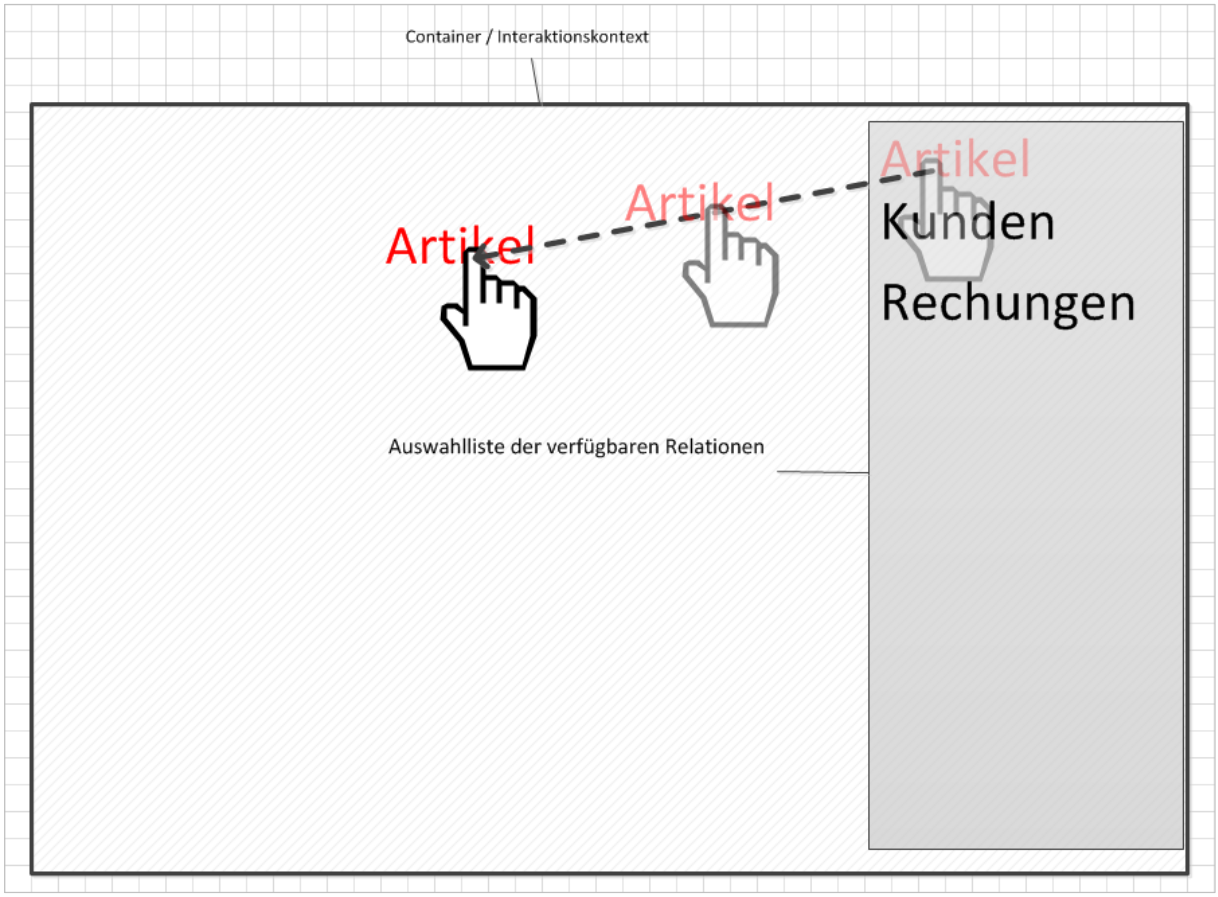


Abbildung 6 - Interaktionscontainer: Hinzufügen einer Relation

Die positionierten Relationen werden auf der Arbeitsfläche wiederum durch rechteckige Elemente dargestellt. Abbildung 7 illustriert den Interaktionscontainer, nachdem einige Relationen der Datenbank auf dem Interaktionscontainer abgelegt wurden.

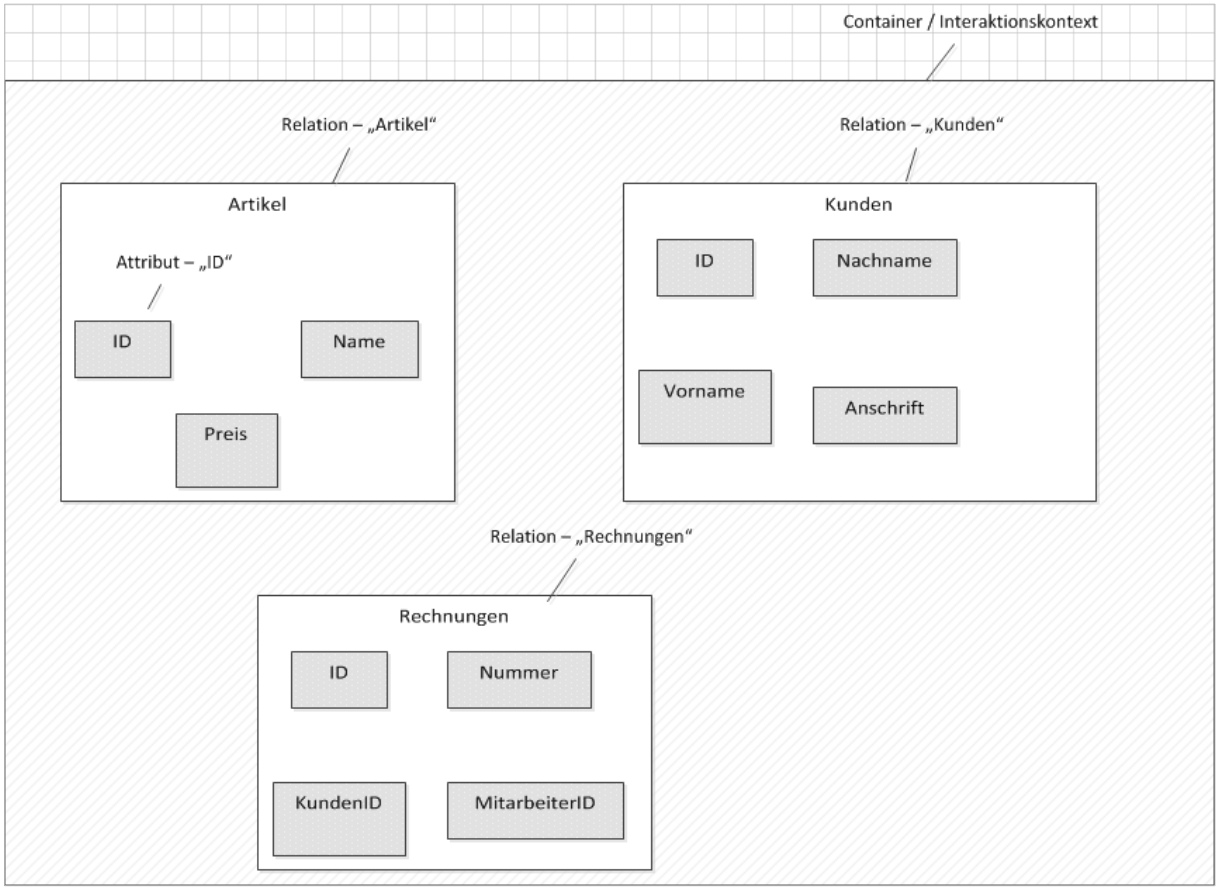


Abbildung 7 - Interaktionscontainer: Vollständig gefüllt

Bei den positionierten Relationen handelt es sich um Instanzen der Ausgangstabellen. Somit kann eine Relation mehrfach auf dem Interaktionscontainer platziert werden. Die Notwendigkeit der Mehrfachplatzierung liegt zum einen darin, einen Selbstverbund (Kapitel 4.4.7) darstellen zu können. Zum anderen werden Relationen mehrfach benötigt, um beispielsweise eine Unteranfrage mit Referenz zu einer schon existierenden Relation formulieren zu können. Der Mehrwert und die Notwendigkeit dieser Spezifikation wird in Kapitel 4.5 verdeutlicht.

Da bestimmte Operationen (Kapitel 4) an Hand der Positionierung der Elemente definiert werden, bietet der Interaktionscontainer die Möglichkeit, alle enthaltenen Elemente unter Berücksichtigung der Elementpositionen zu skalieren. Somit wird die Positionierung aller Elemente untereinander beibehalten und definierte Operationen bleiben bestehen.

3.2 Das Relationselement

Das Relationselement ist der Container der Relationen der Datenbank. Ein solcher Container entsteht automatisch mit dem Platzieren einer Relation auf dem Interaktionscontainer. Dieses Element beinhaltet zwei Darstellungsmöglichkeiten. Zum einen die Darstellung der Attribute (Spalten), die Schemaansicht. Zum anderen die Darstellung der Ergebnismenge einer Anfrage, die Tupelansicht. Das Umschalten der Ansichten erfolgt mit Hilfe einer Tippgeste auf die Titelzeile des Relationselementes.

Um einige Operationen (Kapitel 4.4.1, 4.4.6) per Geste definieren zu können, benötigt das Relationselement, unabhängig vom Interaktionscontainer, die Möglichkeit zur Skalierung, so dass jedes Relationselement unabhängig vom Interaktionscontainer und unabhängig von allen anderen Elementen auf eine gewünschte Größe angepasst werden kann. Ein weiterer Grund dieser Spezifikation ist, dass bei einer großen Menge an Elementen, der Inhalt dieser nicht mehr lesbar ist. Somit ist auch hier die Funktionalität des Heranzoomens und Herauszoomens notwendig.

3.2.1 Die Schemaansicht

In der Schemaansicht werden alle Attribute (Spalten) einer Relation dargestellt. Hier erfolgt die Formulierung der Gesten zur Filterung und Vereinigung. Die nachfolgende Abbildung 8 illustriert das Element der Relation *Artikel* in der Schemaansicht.

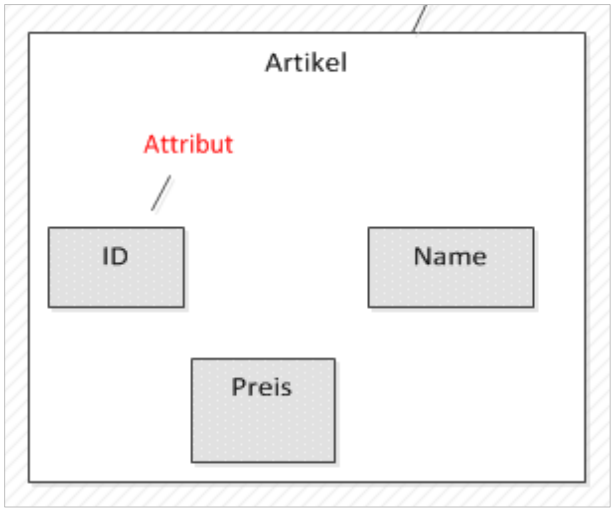


Abbildung 8 - QBG: Schemaansicht

3.2.2 Die Tupelansicht

In der Tupelansicht werden die enthaltenen Werte einer Relation dargestellt. Das gilt nicht nur für die Darstellung der Werte einer Tabelle, sondern auch für die Ergebnismenge einer formulierten Anfrage auf dieser Relation. Die nachfolgende Abbildung 9 skizziert das Element der Relation *Artikel* in der Tupelansicht.

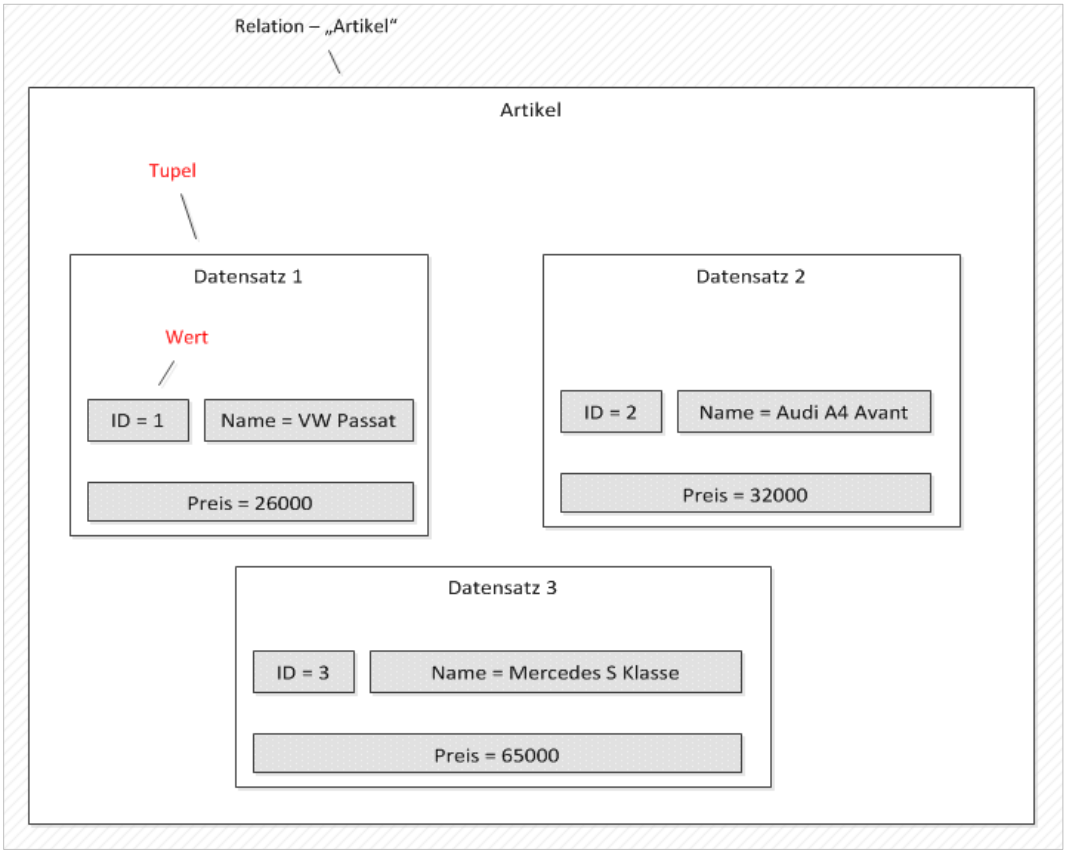


Abbildung 9 - QBG: Tupelansicht

Die Darstellung des Gesamtergebnisses einer Anfrage über mehrere Relationen führt zu der Frage, wo dieses Ergebnis dargestellt wird. In Kapitel 5 wird diese Frage aufgegriffen und an Hand eines Beispiels verdeutlicht. Es werden zudem verschiedene Probleme bei der Lösung dieser Frage und die entsprechenden Lösungsansätze skizziert.

4 Spezifikation der Operationen (Gesten)

In den nachfolgenden Unterkapiteln werden die grundlegenden Operationen der relationalen Algebra kurz vorgestellt. An Hand der Datenbank aus Anhang IV wird anschließend pro Operation jeweils ein Beispiel formuliert. Die Gegenüberstellung der Anfragesprachen SQL, QBE und QBG bildet den Hauptteil der Unterkapitel, wobei pro Anfragesprache das Beispiel aufgegriffen und in der jeweiligen Sprache formuliert wird. Eine ausführliche Erklärung der Geste erfolgt in dem jeweiligen Abschnitt **QBG**.

4.1 Selektion

Die Selektion wählt bestimmte Tupel aus einer Relation aus, wobei die Auswahl mit der Selektionsbedingung eingeschränkt wird.

Am Beispiel der Selektion des Attributs *Nachname* der Relation *Kunden* mit der Selektionsbedingung [*='Mustermann']* werden die Interaktionstechniken SQL, QBE und QBG gegenübergestellt.

SQL:

```
SELECT
	ID, Nachname, Vorname, Anschrift, Alter
FROM
	dbo.Kunden
WHERE
	Nachname = 'Mustermann'
```

QBE:

Der obere Teilbereich des QBE-Editors wird hier nicht dargestellt, da die Formulierung der Selektionsbedingung nur im Tabellengerüst erfolgen kann.

Column	Alias	Table	Output	Sort Type	Sort Order	Filter
Nachname		Kunden	☐			= 'Mustermann'
ID		Kunden	☑			
Nachname		Kunden	☑			
Vorname		Kunden	☑			
Anschrift		Kunden	☑			
[Alter]		Kunden	☑			
			■			

Abbildung 10 - QBE: Selektion

QBG:

Die gewünschte Relation (hier *Kunden*) wird aus der Liste der verfügbaren Relationen ausgewählt und auf den Interaktionscontainer positioniert. Anschließend fokussiert der Anwender das Selektionsattribut und skaliert das Element (hier Attribut *Nachname*), bis es die für die auszuführende Geste benötigte Größe erreicht hat. Letztendlich besteht die Geste aus zwei gleichzeitigen horizontalen Wischbewegungen über dem Attribut-Element (Gleichheitszeichen „=“ zeichnen). Abbildung 11 illustriert diese Gestenformulierung.

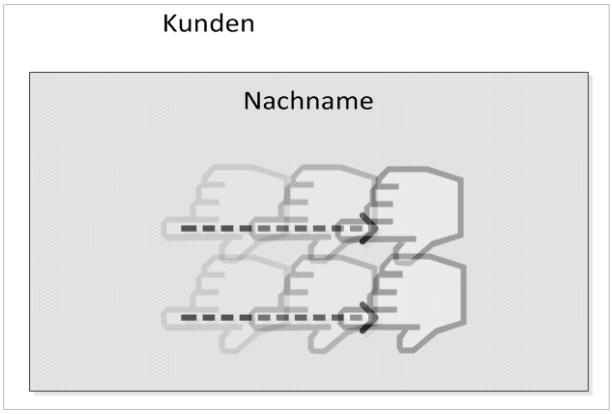


Abbildung 11 - QBG: Selektion

Nach Erkennung der Geste ist es möglich – analog zu QBE – die Bedingung mit Hilfe einer Auswahl zu formulieren. Je nach Datentyp wird entweder eine Auswahlliste oder ein Eingabefeld zur Freitexteingabe bereitgestellt. Abbildung 12 illustriert die Eingabe der Selektionsbedingung an Hand eines Eingabefeldes. Hier wird exemplarisch festgelegt, dass der Wert des Attributes *Nachname* dem Wert [*Mustermann*] entsprechen soll.

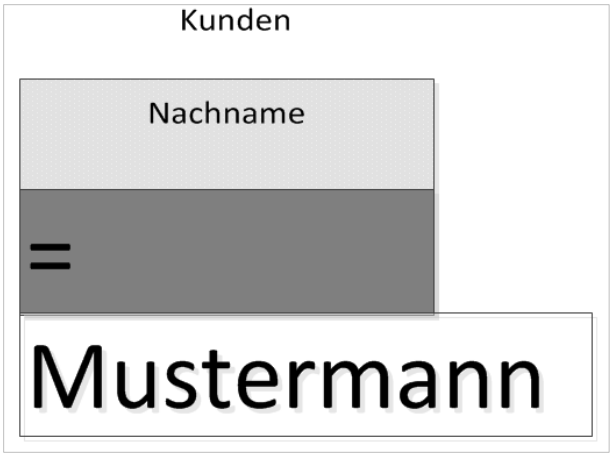


Abbildung 12 - QBG: Selektion - Eingabe Selektionsbedingung

Nach der Formulierung und Bestätigung der Selektionsbedingung, wird das Attribut im Interaktionscontainer entsprechend markiert. Die nachfolgende Abbildung 13 skizziert die Ansicht mit festgelegter Selektionsbedingung.

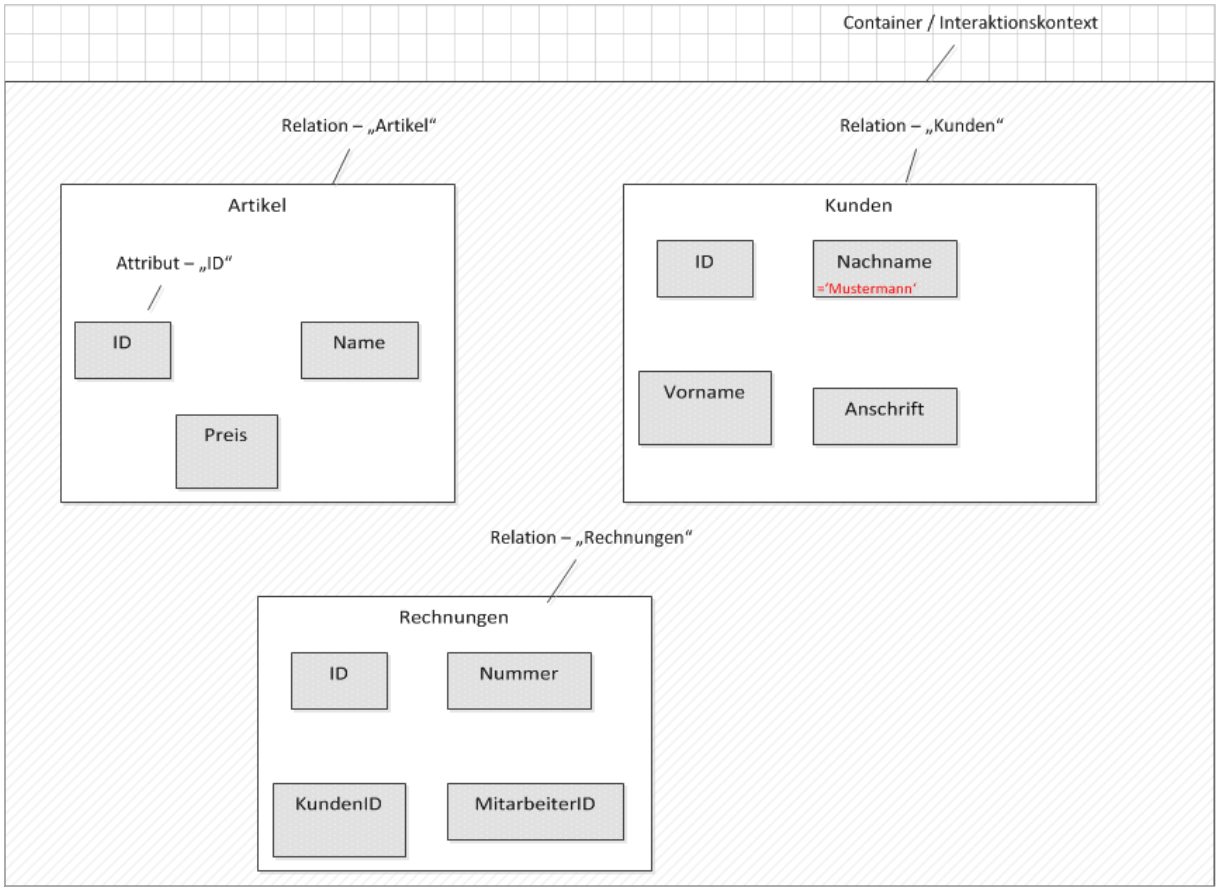


Abbildung 13 - QBG: Selektion - Selektionsbedingung formuliert

Neben der Geste zur Formulierung einer Gleichheitsbedingung, sollen die in Tabelle 1 gelisteten Vergleichsoperatoren unterstützt werden.

Vergleichsoperator	Beschreibung
=	Vergleich auf Gleichheit
<> / != / ≠	Vergleich auf Ungleichheit
<	Vergleich auf Kleiner
≤ / <=	Vergleich auf Kleiner-Gleich
>	Vergleich auf Größer
≥ / >=	Vergleich auf Größer-Gleich

Tabelle 1 - Vergleichsoperatoren

4.2 Projektion

Die Projektion ist die Auswahl bestimmter Attribute, die in der Ergebnismenge angezeigt werden.

Am Beispiel der Projektion der Attribute *Nachname* und *Vorname* der Relation *Kunden* werden die Interaktionstechniken SQL, QBE und QBG gegenübergestellt.

SQL:

```
SELECT Nachname, Vorname FROM dbo.Kunden
```

QBE:

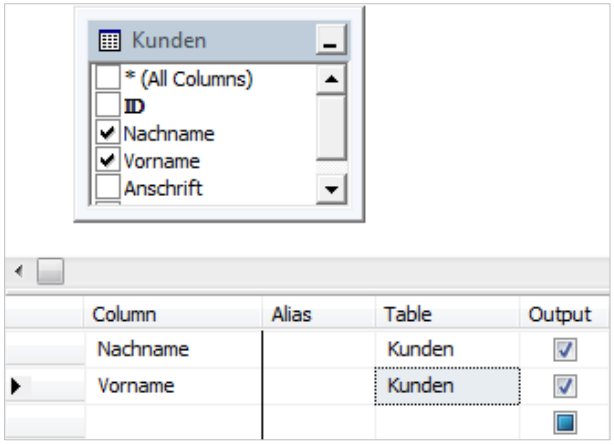


Abbildung 14 - QBE: Projektion

QBG:

Die gewünschte Relation (hier *Kunden*) wird aus der Liste der verfügbaren Relationen ausgewählt und auf den Interaktionscontainer positioniert. Anschließend sucht der Anwender die Attribute und wählt diese mit einer Tippgeste auf das entsprechende Element aus. Das Element wird nach der Auswahl markiert. So ist ersichtlich, welche Attribute in der Ergebnismenge zur Verfügung stehen. Abbildung 15 illustriert die Auswahl des Attributs *Nachname* und die anschließende Auswahl des Attributs *Vorname* der Relation *Kunden*.

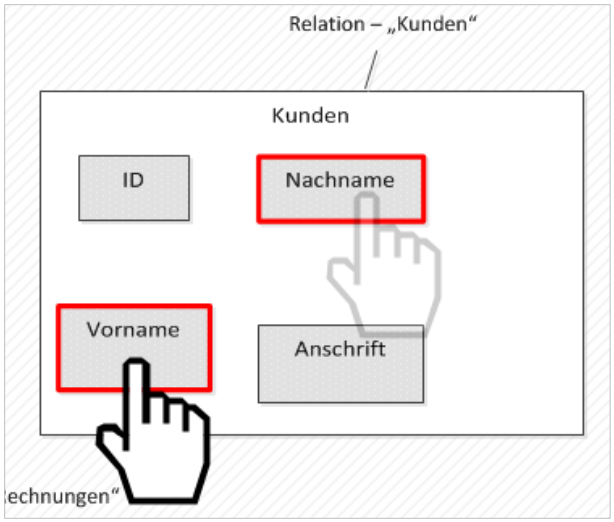


Abbildung 15 - QBG: Projektion

4.2.1 Aggregate

Eine Besonderheit bei der Projektion ist die Nutzung von Aggregatfunktionen. Hier wird die Ergebnismenge vor der Präsentation durch eine Aggregatfunktion manipuliert, so dass beispielsweise die Anzahl der betroffenen Zeilen einer Anfrage ausgegeben wird.

Die in Tabelle 2 aufgelisteten Aggregatfunktionen werden unterstützt.

Aggregatfunktion	Beschreibung
MIN	Berechnet den kleinsten Wert einer Spalte.
MAX	Berechnet den größten Wert einer Spalte.
SUM	Berechnet die Summe der Werte einer Spalte, wobei der Wertebereich/Datentyp der Spalte numerisch sein muss.
COUNT	Berechnet die Anzahl der Werte einer Spalte oder im Spezialfall *COUNT(*)* die Anzahl der Tupel einer Relation.
AVG	Berechnet den arithmetischen Mittelwert der Werte einer Spalte, wobei der Wertebereich/Datentype der Spalte numerisch sein muss.

Tabelle 2 - Aggregatfunktionen [SSH13:318]

Am Beispiel der Projektion des Aggregats *MIN(Kunden.Alter)* wird der jüngste Kunde in der Datenbank ermittelt. Im Folgenden werden an Hand dieses Beispiels die Interaktionstechniken SQL, QBE und QBG gegenübergestellt.

SQL:

```
SELECT MIN(Alter) FROM Kunden
```

QBE:

Die Abbildung 16 illustriert die formulierte Abfrage in QBE. Die Auswahl der Aggregatfunktion (hier *MIN*) erfolgt im Tabellengerüst in der Spalte *Group By*. In diesem Zusammenhang ist die Spaltenbezeichnung nicht optimal gewählt, da hier noch keine Gruppierung vorgenommen werden muss. Es erfolgt lediglich die Ausgabe eines Skalars. Die Gruppierung wird in Kapitel 4.8 näher betrachtet.

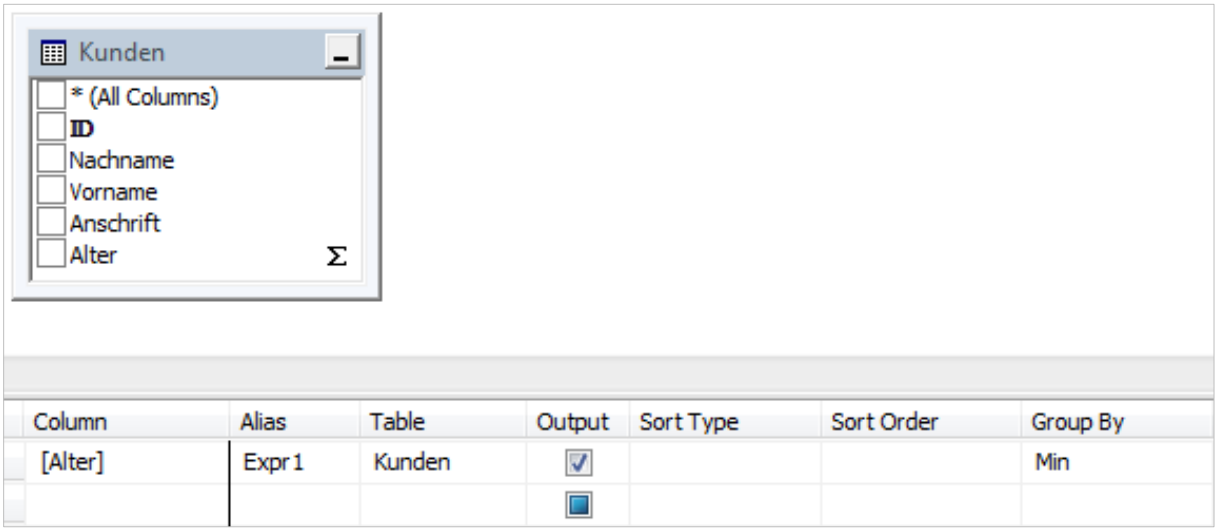


Abbildung 16 - QBE: Aggregat

QBG:

In QBG kann eine Aggregatfunktion nach dem Auswählen (Tippgeste) eines Attributes gewählt werden. Hierzu wird dem Anwender direkt nach der Auswahl ein Menü zur Verfügung gestellt. Dieses beinhaltet die in Tabelle 2 gelisteten Aggregatfunktionen, die wiederum mittels Tippgeste angewendet werden können. Erfolgt innerhalb einer gewissen Zeit keine Auswahl der Aggregatfunktion, wird das selektierte Attribut zur Projektion hinzugefügt.

Abbildung 17 skizziert die Auswahl des Attributs (hier: *Kunden.ID*) mittels Tippgeste auf das Element.

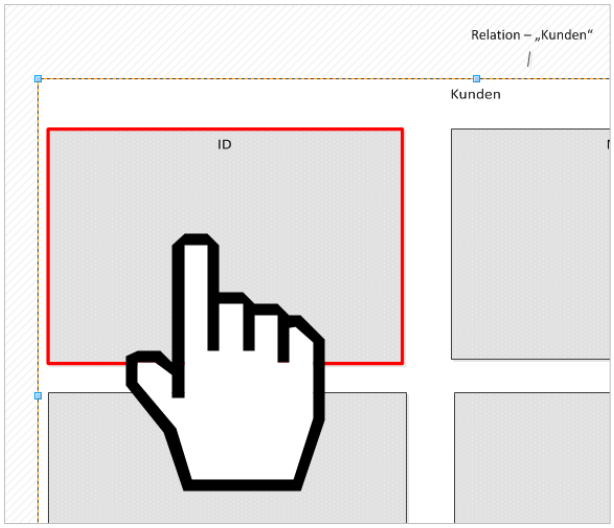


Abbildung 17 - QBG: Aggregat – Attrributauswahl

Nachdem die Auswahl erfolgt ist, wird das in Abbildung 18 illustrierte Auswahlmenü über dem Element des zuvor selektierten Attributs dem Anwender zur Verfügung gestellt.

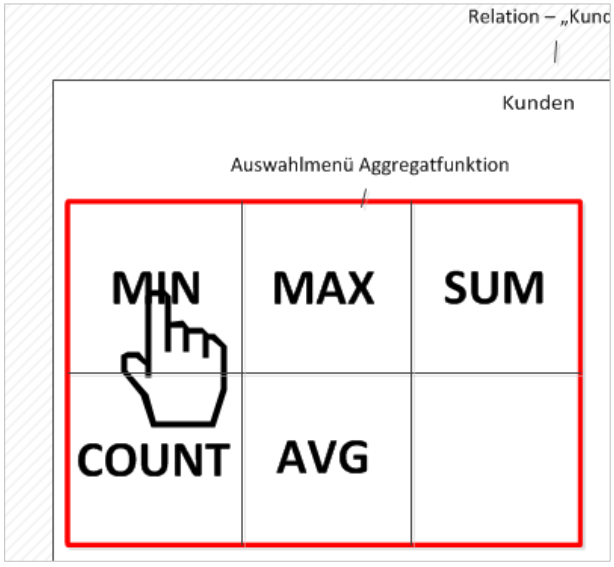


Abbildung 18 - QBG: Aggregat – Funktionsauswahl

Hier hat der Benutzer nun die Möglichkeit eine Aggregatfunktion auszuwählen, oder ohne Auswahl die Projektion des Attributs (hier: *Kunden.ID*) in die Abfrage zu übernehmen.

4.3 Umbenennung

Die Umbenennung ist eine Operation in der relationalen Algebra und wird für die notwendige Namensgleichheit diverser anderer Operationen, zum Beispiel bei der Vereinigung, genutzt. Da eine Umbenennung von modernen, auf SQL basierenden Datenbanksystemen implizit durchgeführt wird, ist die Definition einer Geste in QBG nicht zwingend erforderlich. Auf Grund der Vollständigkeit der Operationen und besseren Lesbarkeit eines Anfrageergebnisses, wird die Umbenennung dennoch in QBG wie folgt definiert.

Am Beispiel der Umbenennung des Attributes *Nachname* zu *Lastname*, werden die Interaktionstechniken SQL, QBE und QBG gegenübergestellt.

SQL:

```
SELECT Nachname AS Lastname FROM Kunden
```

QBE:

Der obere Teilbereich des QBE-Editors wird hier nicht dargestellt, da die Formulierung der Umbenennung (Alias) nur im Tabellengerüst erfolgen kann.

Abbildung 19 - QBE: Umbenennung

QBG:

Die gewünschte Relation (hier *Kunden*) wird aus der Liste der verfügbaren Relationen ausgewählt und auf den Interaktionscontainer positioniert. Anschließend sucht der Anwender das umzubenennende Attribut und führt eine Wischbewegung in der Horizontalen aus. Die Geste erfordert, dass die Bewegung annähernd an dem Punkt endet, an dem diese angefangen hat und mindestens die Hälfte der Breite des fokussierten Elementes auf halbem Weg zurücklegt. Die nachfolgende Abbildung 20 illustriert diese Geste.

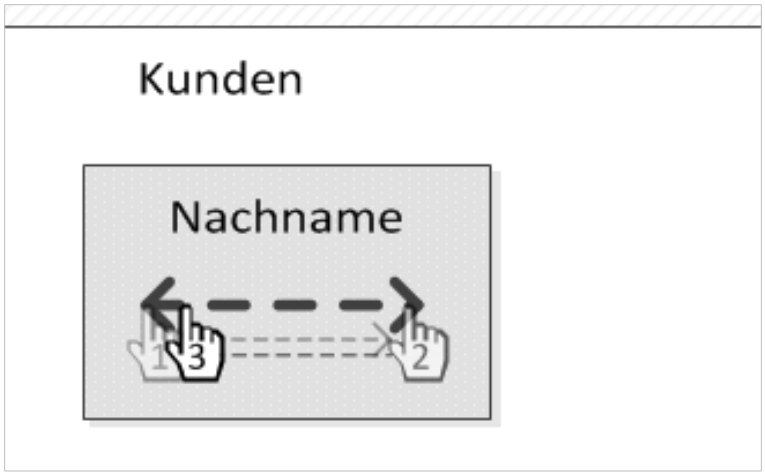


Abbildung 20 - QBG: Umbenennung

Die Geste beginnt hier an Position 1. Es erfolgt eine horizontale Wischbewegung zum rechten Rand des Attribut-Elementes *Nachname* und wieder zurück zum Ausgangspunkt am linken Rand. Die Positionen 1, 2 und 3 werden hier ohne Unterbrechung mit einer Wischbewegung angesteuert.

4.4 Verbund (Join)

Der Verbund verbindet zwei Relationen miteinander und stellt das Ergebnis als neue einheitliche Relation dar. Dieses Ergebnis wird in der Tupelansicht der Quellrelation dargestellt. Man unterscheidet verschiedene Verbund-Operationen, die in den nachfolgenden Unterkapiteln erläutert werden.

4.4.1 Kartesisches Produkt (Cross Join)

Das kartesische Produkt verbindet jede Zeile einer Relation mit jeder Zeile einer anderen Relation. Das Resultat eines kartesischen Produktes ist folglich die Menge aller Kombinationen der Tupel der beteiligten Relationen.

Am Beispiel des kartesischen Produktes der Relationen *Kunden* und *Mitarbeiter* werden die Interaktionstechniken SQL, QBE und QBG gegenübergestellt.

SQL:

```
SELECT * FROM Kunden CROSS JOIN Mitarbeiter
```

QBE:

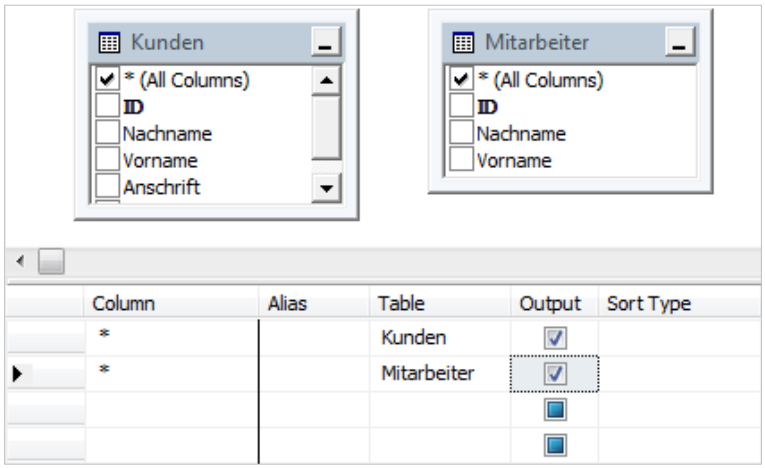


Abbildung 21 - QBE: Kartesischen Produkt

QBG:

Um in QBG ein kartesisches Produkt zu formulieren, werden die gewünschten Relationen auf dem Interaktionscontainer platziert. Anschließend wird die Quellrelation (hier: *Kunden*) mittels der entsprechenden Geste größer skaliert, so dass dieses Element die Elemente der anderen Relationen (hier: *Mitarbeiter*) vollständig überlappen kann. Im letzten Schritt wird das Element der Quellrelation über alle Elemente der anderen beteiligten Relationen verschoben. Die nachfolgenden Abbildungen skizzieren diesen Ablauf.

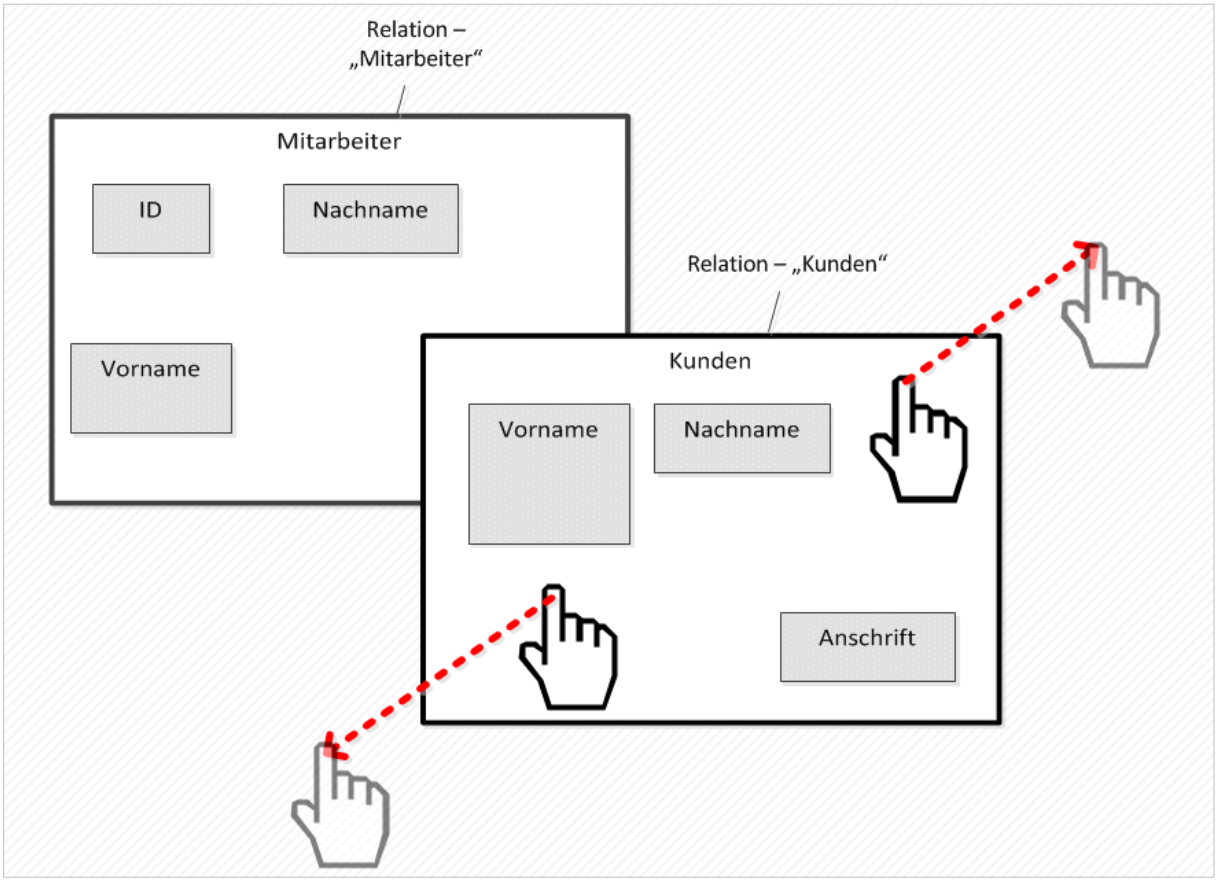


Abbildung 22 - QBG: Kartesisches Produkt – Skalieren 1

Abbildung 22 illustriert die Skalierung des Elementes der Quellrelation (hier: *Kunden*). Zur Skalierung wird dieses Element an zwei Positionen gleichzeitig manipuliert und mit einer jeweils nach außen führenden diagonalen Wischbewegung vergrößert.

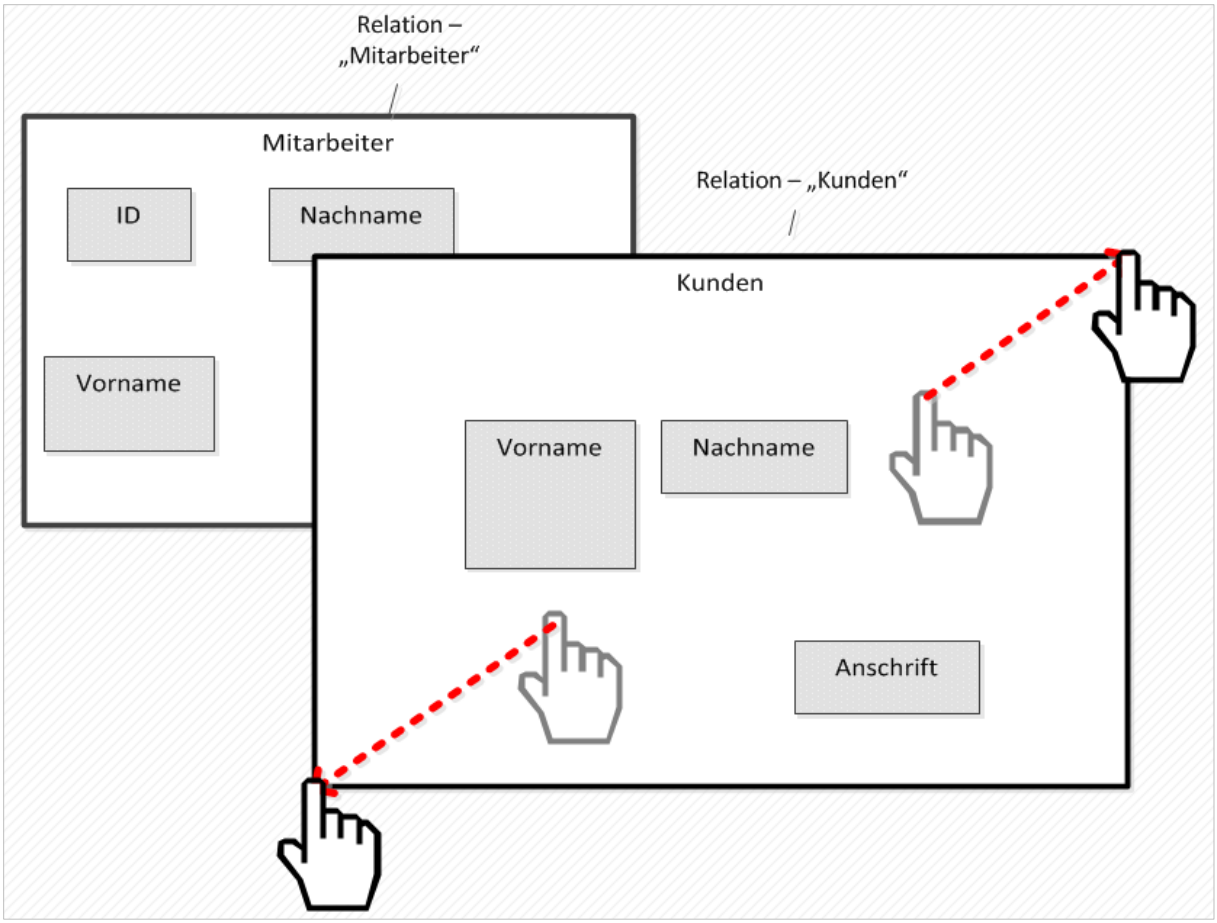


Abbildung 23 - QBG: Kartesisches Produkt – Skalieren 2

Das so vergrößerte Element (Abbildung 23) wird anschließend über das zu beteiligende Relationselement (hier: *Mitarbeiter*) verschoben (Abbildung 24).

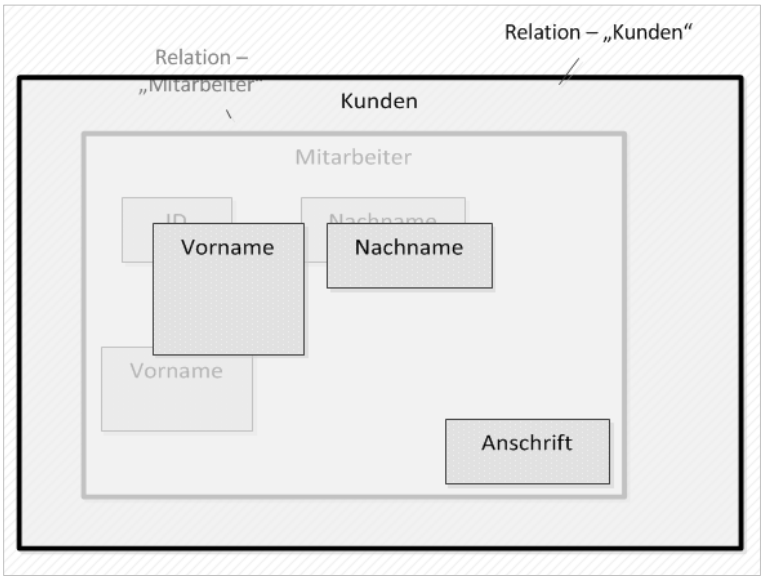


Abbildung 24 - QBG: Kartesisches Produkt – Verschoben

4.4.2 Gleichheitsverbund (Equivalent Join/Inner Join)

Der Gleichheitsverbund verbindet die Datensätze aus zwei Tabellen an Hand eines Selektionsprädikates miteinander. In diesem Prädikat wird angegeben, welcher Attributwert der einen Relation mit welchem Attributwert der zweiten Relation übereinstimmen muss. Alle nicht übereinstimmenden Verknüpfungen der Datensätze werden in der Ergebnismenge ausgeblendet.

Am Beispiel des Gleichheitsverbundes des Attributes *Nachname* der Relationen *Kunden* und *Mitarbeiter* werden die Interaktionstechniken SQL, QBE und QBG gegenübergestellt.

SQL:

```
SELECT * FROM Kunden
INNER JOIN  Mitarbeiter ON Mitarbeiter.Nachname = Kunden.Nachname
```

QBE:

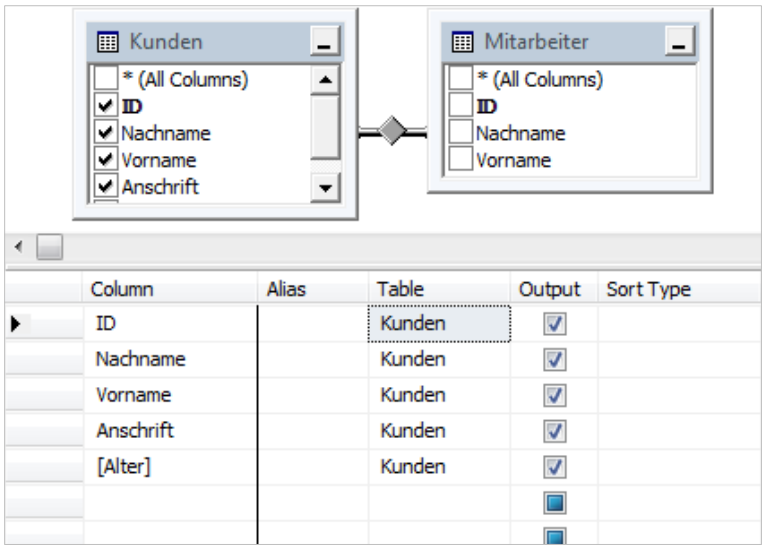


Abbildung 25 - QBE: Gleichheitsverbund

Die Verbindungslinie zwischen den Relationen *Kunden* und *Mitarbeiter* des oberen Teilbereichs stellt das Selektionsprädikat dar. Diese Bedingung wird durch eine Maus-Geste automatisch erzeugt und kann in einem separaten Dialog (Abbildung 26) angepasst werden.

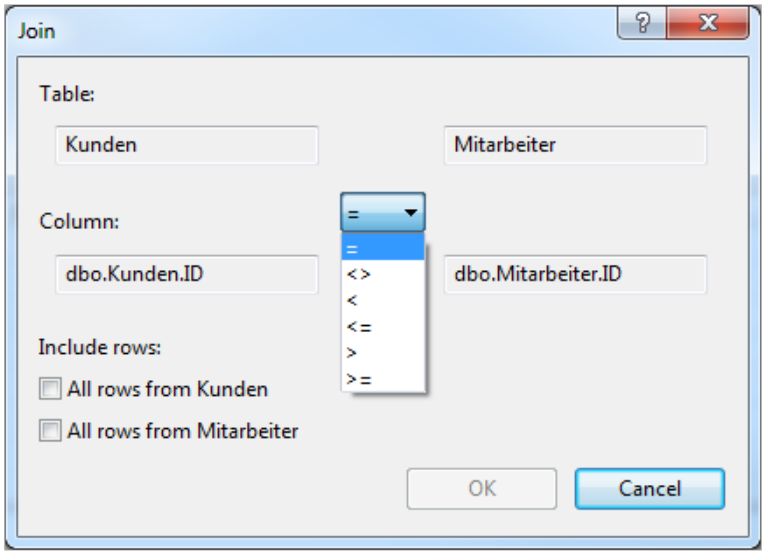


Abbildung 26 - QBE: Dialog Verbundprädikat

Hierbei definiert die Nichtauswahl der Optionen *„All rows from Kunden“* und *„All rows from Mitarbeiter“* den Inner Join.

QBG:

Die Formulierung des Gleichheitsverbunds beginnt mit dem Verbinden von zwei Attributen, die das Selektionsprädikat darstellen. Hierzu wählt man ein Attribut einer Relation mit einer Tippgeste aus und zieht dieses Element, ohne Loslassen, auf das Zielattribut einer anderen Relation. Die nachfolgende Abbildung 27 illustriert diese Geste.

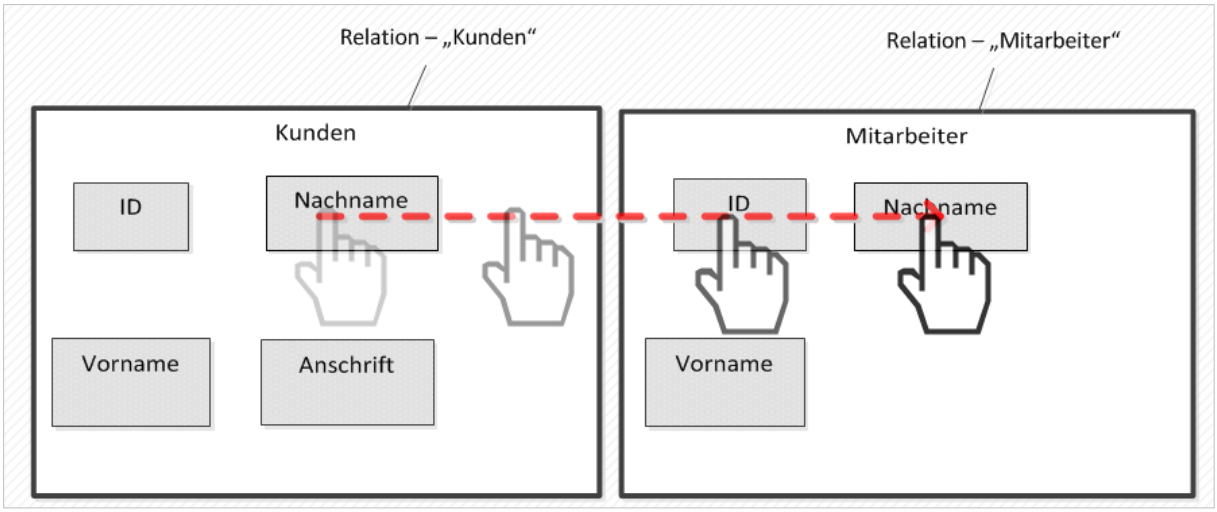


Abbildung 27 - QBG: Gleichheitsverbund – Selektionsprädikat

Hier wird das Attribut *Nachname* der Relation *Kunden* ausgewählt und, ohne Loslassen der Auswahl, auf das Attribut *Nachname* der Relation *Mitarbeiter* gezogen. Anschließend erfolgt, analog zu Abbildung 26, die Auswahl des Vergleichsoperators an Hand einer Auswahlliste (Abbildung 28).

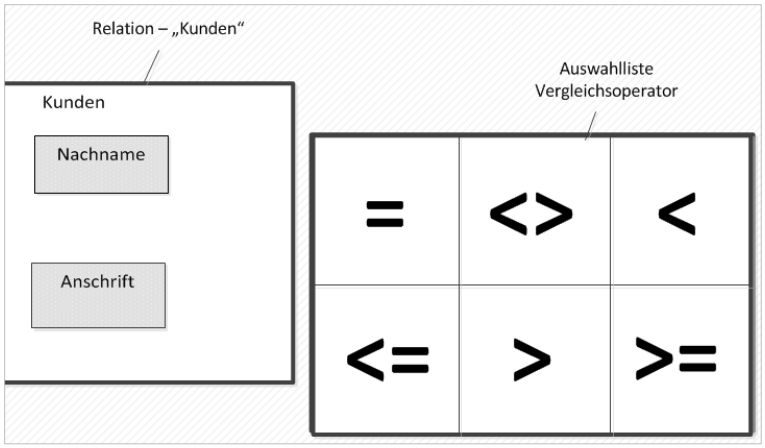


Abbildung 28 - QBG: Gleicheitsverbund – Verbundprädikat

Das Selektionsprädikat (hier *Kunden.Nachname = Mitarbeiter.Nachname*) wird anschließend als Verbindungslinie zwischen den Attributen der Relationen dargestellt (Abbildung 29).

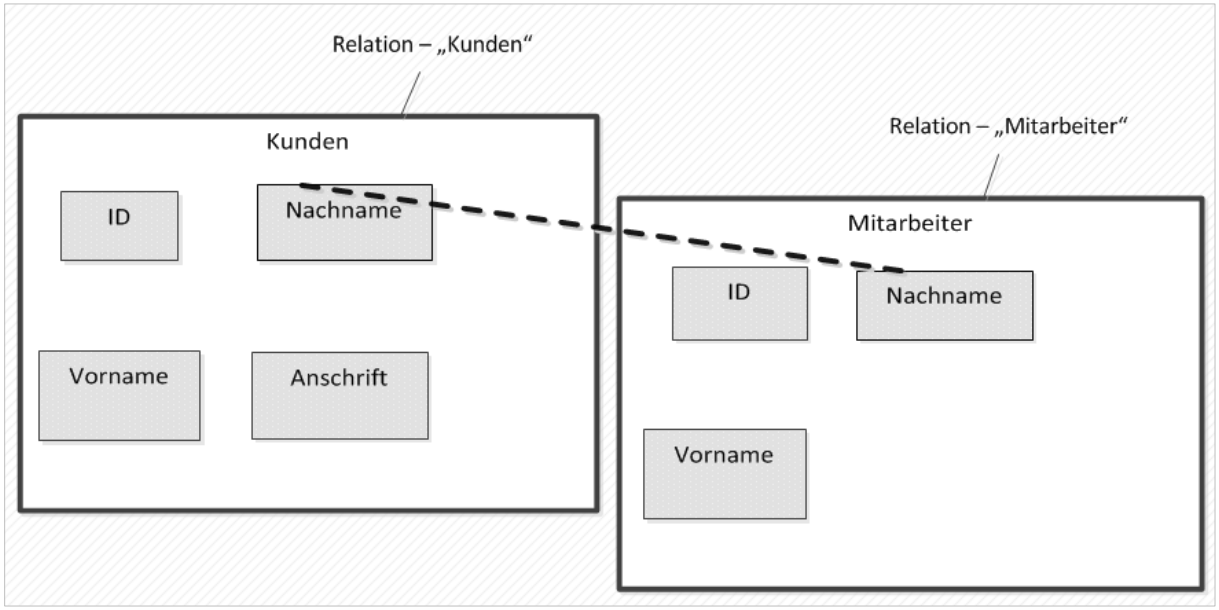


Abbildung 29 - QBG: Darstellung Selektionsprädikat

4.4.3 Natürlicher Verbund (Natural Join)

Ein natürlicher Verbund verknüpft Relationen über gleich benannte Spalten bei gleichen Attributwerten. Im Gegensatz zu einem Gleichheitsverbund wird bei dem natürlichen Verbund kein Verbundprädikat angegeben. Das Datenbanksystem erkennt automatisch das Attribut an Hand der Namensgleichheit der Attribute in allen Relationen und stellt über diese Attribute einen Verbund her. Hierbei werden mehrfach vorkommende Spalten in der Ergebnismenge ausgeblendet.

Da ein natürlicher Verbund über den Gleichheitsverbund abgebildet werden kann, indem gleich benannte Spalten ausgeblendet werden, ist eine Spezifikation einer entsprechenden Geste nicht notwendig. Das System entscheidet automatisch, ob ein formulierter Gleichheitsverbund zu einem natürlichen Verbund umgewandelt werden kann.

Ein Nachteil dieses Automatismus ist, dass immer ein Verbundprädikat formuliert werden muss, obwohl ein natürlicher Verbund dieses Prädikat normalerweise automatisch wählt. Dieser Nachteil ist zu vernachlässigen, da es sonst bei der Formulierung einer Geste zu einer Verwechslung mit dem kartesischen Produkt führt.

Die folgende SQL-Anfrage würde einen natürlichen Verbund zwischen den Relationen *Kunden* und *Mitarbeiter* erzeugen:

```
SELECT * FROM Kunden NATURAL JOIN Mitarbeiter
```

Da sowohl der QBE-Editor „Microsoft SQL-Server 2008 R2 Management Studio“, also auch „Microsoft Access 2010“ keine grafische Unterstützung zur Definition eines natürlichen Verbundes bieten, wird im Folgenden auf den Vergleich der Interaktionstechnik mit QBE verzichtet.

4.4.4 Inklusionsverbund links (Left Join / Left Outer Join)

Der linke Inklusionsverbund gehört zu der Gruppe der äußeren Verbunde. Hier werden alle Datensätze aus der ersten Relation, die dem Selektionsprädikat entsprechen, in die Ergebnismenge aufgenommen, auch wenn keine entsprechenden Werte für den Datensatz in der zweiten Relation existieren.

Am Beispiel des linken Inklusionsverbundes der Relationen *Kunden* und *Mitarbeiter* mit dem Selektionsprädikat *„Kunden.Nachname = Mitarbeiter.Nachname“* werden die Interaktionstechniken SQL, QBE und QBG gegenübergestellt.

SQL:

```
SELECT * FROM Kunden
LEFT JOIN Mitarbeiter ON Kunden.Nachname = Mitarbeiter.Nachname
```

QBE:

Abbildung 30 illustriert den linken Inklusionsverbund. Der Fokus liegt hier auf dem oberen Teilbereich. Hier wird mit Hilfe des Symbols (▷) verdeutlicht, dass es sich um einen Left Join handelt.

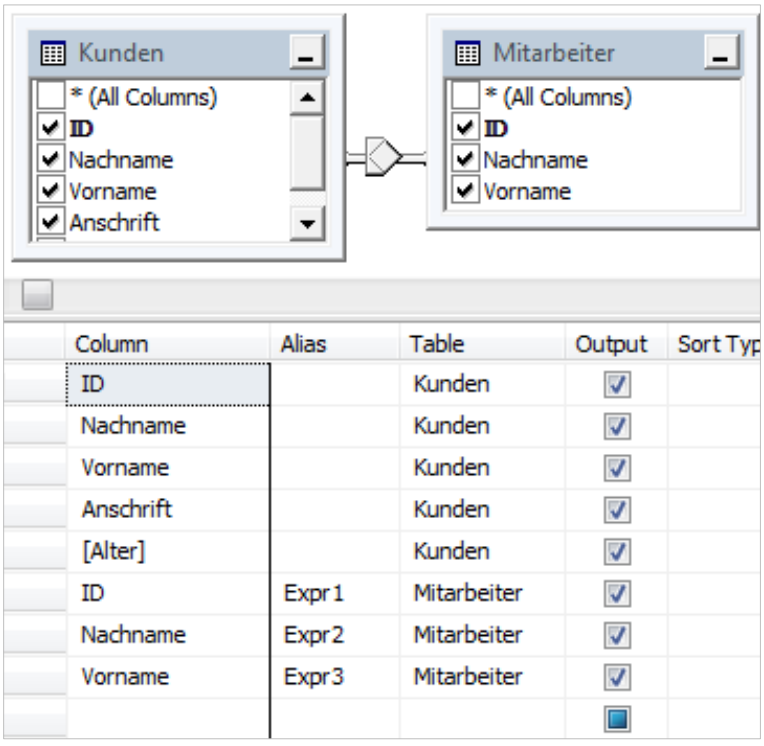


Abbildung 30 - QBE: Inklusionsverbund links

Der Typ des Verbundes kann, analog zu Abbildung 26, in einem separaten Dialog angepasst werden. Hierbei definiert die Option *„All rows from Kunden“* den Left Join. Abbildung 31 demonstriert diesen Dialog.

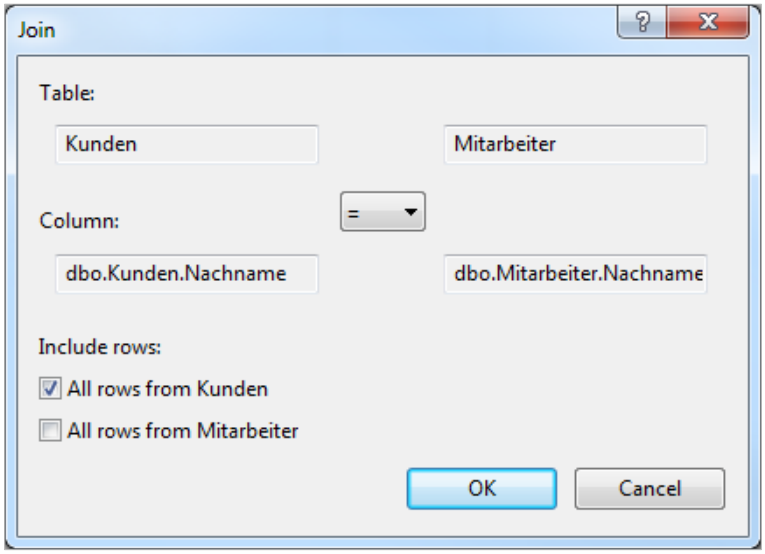


Abbildung 31 - QBE: Dialog Left Join

QBG:

Die Formulierung eines linken Inklusionsverbundes erfolgt ähnlich zu der Formulierung des Gleichheitsverbundes. Hier wird ebenso das Selektionsprädikat per Drag & Drop definiert. Nach der Definition des Selektionsprädikates entsteht automatisch ein Inner Join. Dieser Verbund kann nachfolgend mit dem Verschieben einer der an dem Verbund beteiligten Relationen in einen Left Join umgewandelt werden. Dazu ist es notwendig, dass das Element der Quellrelation (hier: *Kunden*) mit den Elementen der beteiligten Relationen (hier: *Mitarbeiter*) um maximal die Hälfte der Breite **von links** überschnitten wird.

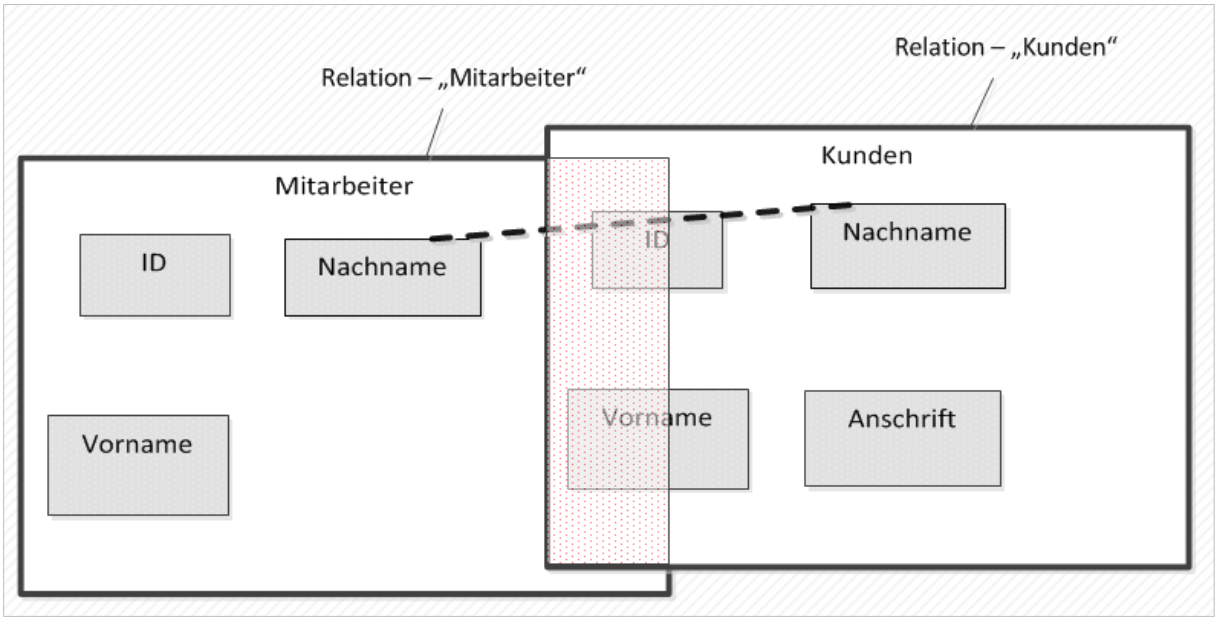


Abbildung 32 - QBG: Left Join

Der in Abbildung 32 rot gepunktete Bereich hebt die Überschneidung der beiden Relationselemente hervor. Zu sehen ist hier, dass das Element der Relation *Mitarbeiter* das Element der Quellrelation der Verbundoperation (hier: *Kunden*) um maximal die Hälfte der Breite des Elementes der Relation *Kunden* **von links** schneidet.

4.4.5 Inklusionsverbund rechts (Right Join / Right Outer Join)

Der rechte Inklusionsverbund gehört zu der Gruppe der äußeren Verbunde. Hier werden alle Datensätze aus der zweiten Relation, die dem Selektionsprädikat entsprechen, in die Ergebnismenge aufgenommen, auch wenn keine entsprechenden Werte für den Datensatz in der ersten Relation existieren.

Am Beispiel des rechten Inklusionsverbundes der Relationen *Kunden* und *Mitarbeiter* mit dem Selektionsprädikat *„Kunden.Nachname = Mitarbeiter.Nachname“* werden die Interaktionstechniken SQL, QBE und QBG gegenübergestellt.

SQL:

```
SELECT * FROM Kunden
RIGHT JOIN Mitarbeiter ON Kunden.Nachname = Mitarbeiter.Nachname
```

QBE:

Abbildung 33 illustriert den rechten Inklusionsverbund. Der Fokus liegt hier auf dem oberen Teilbereich. Hier wird mit Hilfe des Symbols (◁) verdeutlicht, dass es sich um einen Right Join handelt.

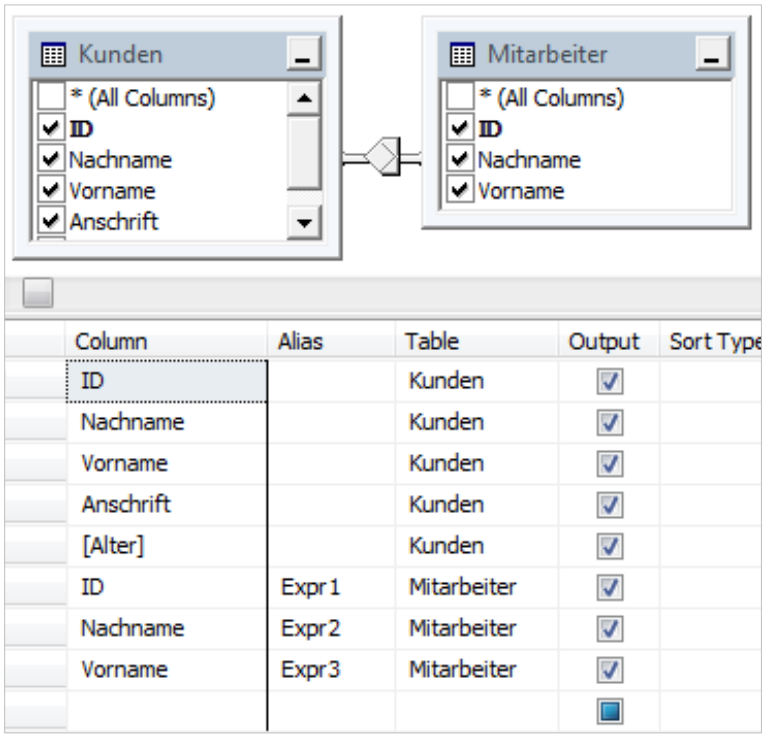


Abbildung 33 - QBE: Inklusionsverbund rechts

Der Typ des Verbundes kann, analog zu Abbildung 26, in einem separaten Dialog angepasst werden. Hierbei definiert die Option *„All rows from Mitarbeiter“* den Right Join. Abbildung 34 demonstriert diesen Dialog.

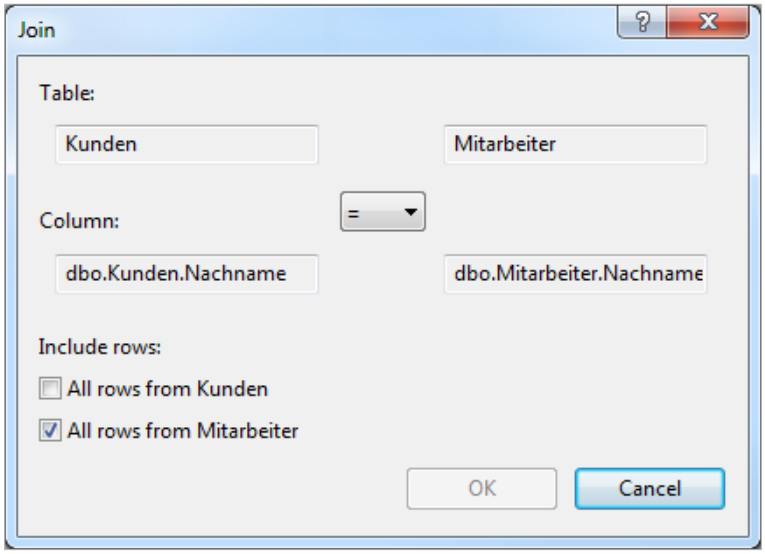


Abbildung 34 - QBE: Dialog Right Join

QBG:

Die Formulierung eines rechten Inklusionsverbundes erfolgt ähnlich zu der Formulierung des Gleichheitsverbundes. Hier wird ebenso das Selektionsprädikat per Drag & Drop definiert. Nach der Definition des Selektionsprädikates entsteht automatisch ein Inner Join. Dieser Verbund kann nachfolgend, mit dem Verschieben einer der an dem Verbund beteiligten Relationen, in einen Right Join umgewandelt werden. Dazu ist es notwendig, dass das Element der Quellrelation (hier: *Kunden*) mit den Elementen der beteiligten Relationen (hier: *Mitarbeiter*) um maximal die Hälfte der Breite **von rechts** überschnitten wird.

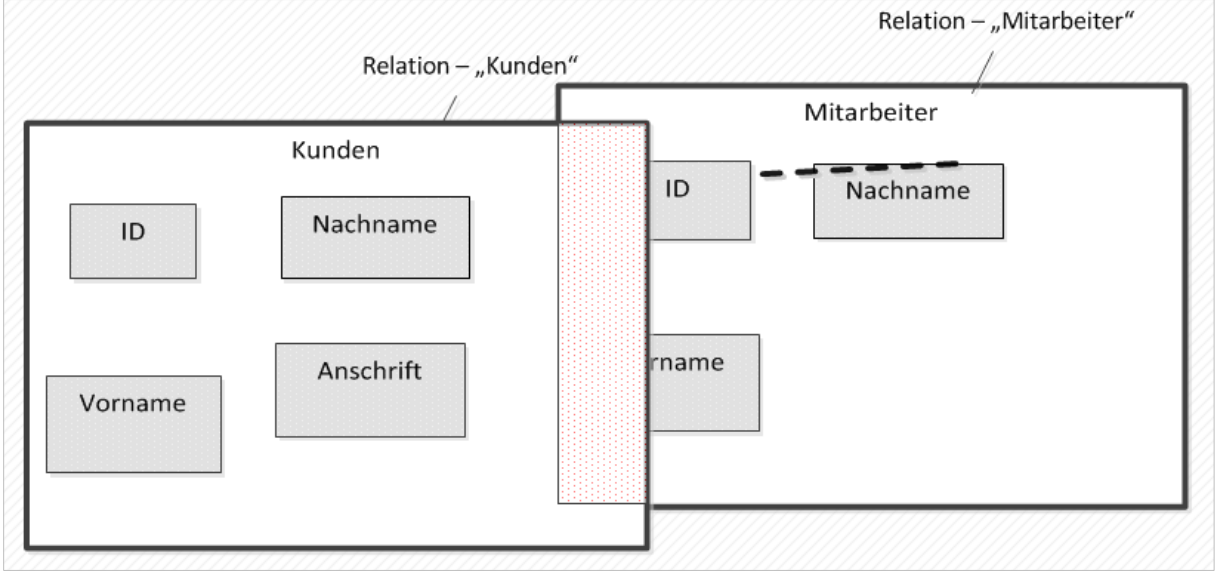


Abbildung 35 - QBG: Right Join

Der in Abbildung 35 rot gepunktete Bereich hebt die Überschneidung der beiden Relationselemente hervor. Zu sehen ist hier, dass das Element der Relation *Mitarbeiter* das Element der Quellrelation der Verbundoperation (hier: *Kunden*) um maximal die Hälfte der Breite des Elementes der Relation *Kunden* **von rechts** schneidet.

4.4.6 Voller Inklusionsverbund (Full Join / Full Outer Join)

Der volle Inklusionsverbund, oder auch Full Join, gehört ebenso zu der Gruppe der äußeren Verbunde. Diese Operation ist die Zusammensetzung des Left Join und des Right Join. Somit werden alle Datensätze der Relationen in die Ergebnismenge aufgenommen, wo entweder der Datensatz aus der ersten Relation oder der Datensatz aus der zweiten Relation dem Selektionsprädikat entspricht.

Am Beispiel des vollen Inklusionsverbundes der Relationen *Kunden* und *Mitarbeiter* mit dem Selektionsprädikat *„Kunden.Nachname = Mitarbeiter.Nachname“* werden die Interaktionstechniken SQL, QBE und QBG gegenübergestellt.

SQL:

```
SELECT * FROM Kunden
FULL JOIN Mitarbeiter ON Kunden.Nachname = Mitarbeiter.Nachname
```

QBE:

Abbildung 36 illustriert den vollen Inklusionsverbund in QBE. Der Fokus liegt hier auf dem oberen Teilbereich. Hier wird mit Hilfe des Pfeil-Symbols (⧄) verdeutlicht, um welchen Verbundtyp es sich handelt.

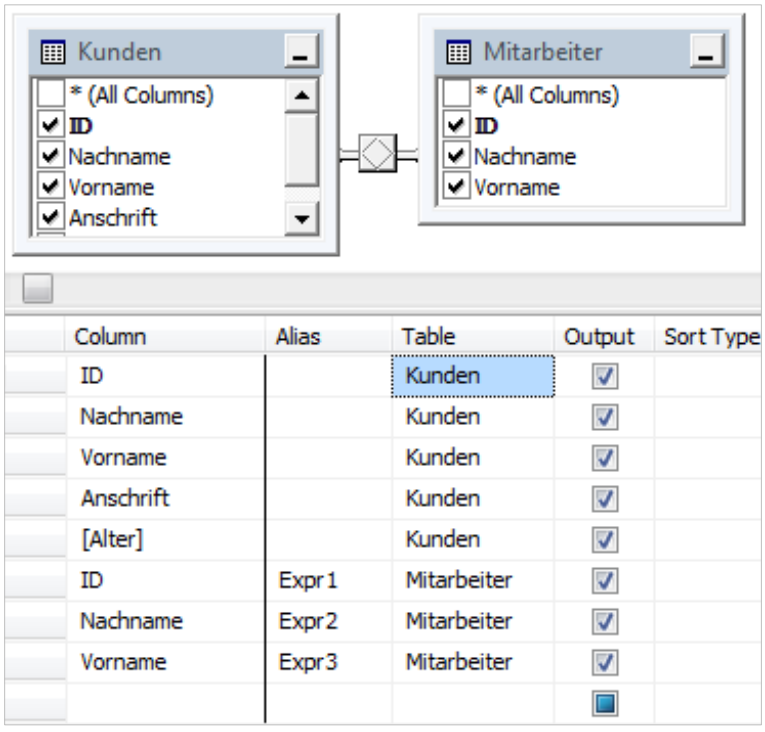


Abbildung 36 - QBE: Voller Inklusionsverbund

Der Typ des Verbundes kann, analog zu Abbildung 26, in einem separaten Dialog angepasst werden. Hierbei definieren die Optionen *„All rows from Kunden“* und *„All rows from Mitarbeiter“* den Full Join. Abbildung 37 demonstriert diesen Dialog.

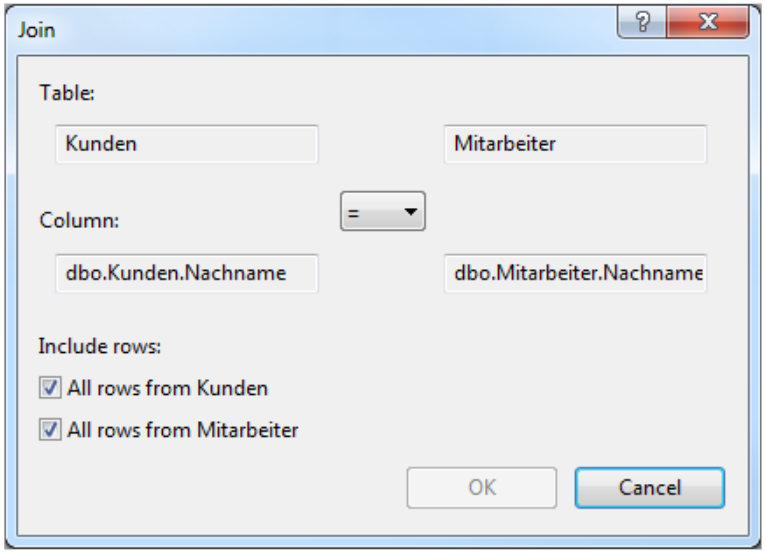


Abbildung 37 - QBE: Dialog Full Join

QBG:

Analog zu dem linken und zu dem rechten Inklusionsverbund, erfolgt zunächst die Formulierung des Selektionsprädikates an Hand einer Drag & Drop –Geste der Attribute. Anschließend wird das erste Relationselement so skaliert, dass man dieses komplett in den Bereich des zweiten Relationselementes verschieben kann.

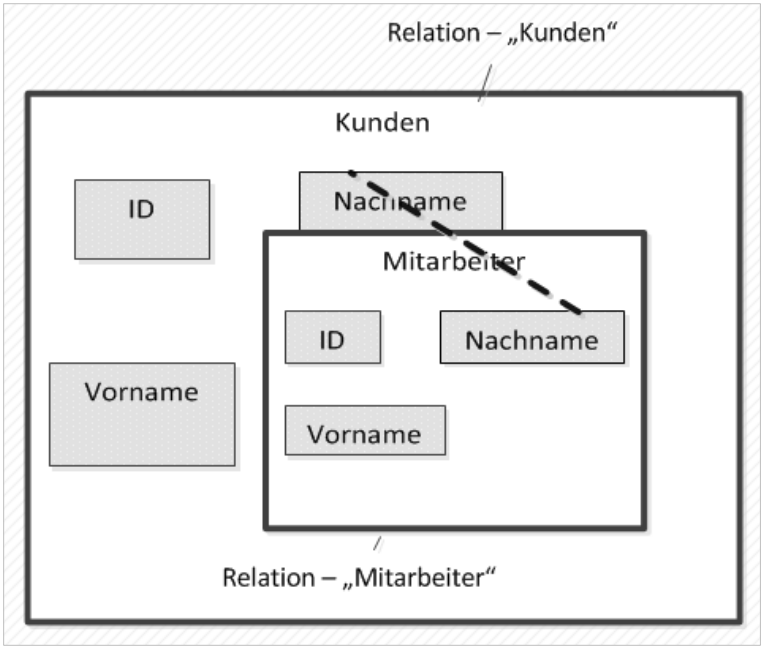


Abbildung 38 - QBG: Full Join

Abbildung 38 illustriert das skalierte Element der Relation *Mitarbeiter* mit anschließendem Verschieben in bzw. über das Element der Relation *Kunden*. Eine Verschiebung des Elements der Relation *Kunden*, würde dieselbe Geste formulieren, da auch dann das Element der Zielrelation (hier: *Mitarbeiter*) innerhalb des Elementes der Quellrelation (hier: *Kunden*) liegen würde.

Grundsätzlich gilt, dass jede der Verbundoperationen Inner Join, Left Join, Right Join und Full Join mit dem Verschieben, ein oder mehrerer, an dem Verbund beteiligten Relationselemente, in jede andere dieser vier Verbundoperationen überführt werden kann.

4.4.7 Selbstverbund (Self Join)

Der Selbstverbund, oder auch Self Join, ist eine Verbundoperation, in der der Verbund über eine einzige Tabelle abgebildet wird. Self Joins können mit allen anderen Verbundoperationen durchgeführt werden, wobei die einzelnen Instanzen der einzigen Relation mit einem Alias angegeben werden müssen. Ein Zugriff auf die Attribute erfolgt anschließend nur über den Alias.

Am Beispiel des Selbstverbunds als Inner Join der Relation *Mitarbeiter* mit dem Selektionsprädikat *„[Alias1].ID = [Alias2].Vorgesetzter“* werden die Interaktionstechniken SQL, QBE und QBG gegenübergestellt.

SQL:

```
SELECT a.*, b.* FROM Mitarbeiter a
INNER JOIN Mitarbeiter b ON a.ID = b.Vorgesetzter
```

QBE:

Die nachfolgende Abbildung 39 illustriert den Selbstverbund in QBE. Hier stellt der QBE-Editor (Microsoft SQL Server Management Studio) die Verbundoperation als Inner Join dar. Dass es sich um einen Selbstverbund handelt, erkennt man an der Verwendung eines Alias-Namens für die einzelnen Instanzen der Relation *Mitarbeiter*. Analog zu den Unterkapiteln 4.4.1, 4.4.2, 4.4.3, 4.4.4, 4.4.5 und 4.4.6 kann der Verbundtyp in dem entsprechenden Bearbeitungsdialog angepasst werden.

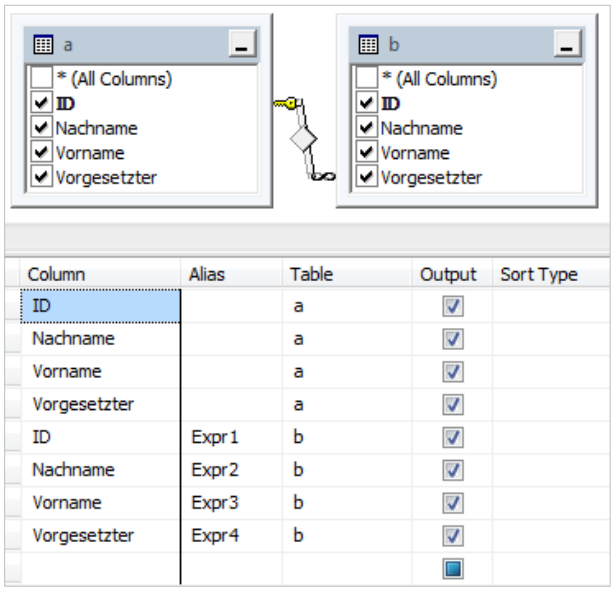


Abbildung 39 - QBE: Selbstverbund

QBG:

Die Formulierung eines Selbstverbunds in QBG erfolgt ähnlich zu der Formulierung aller anderen Verbundoperationen. Der Unterschied liegt jedoch in der Auswahl der Relationen, die auf dem Interaktionscontainer platziert werden müssen. Laut Definition handelt es sich um einen Verbund innerhalb einer Relation, weshalb auf dem Interaktionscontainer die gewünschte Relation mindestens zweimal platziert werden muss. Das System erkennt nach der Platzierung der zweiten Instanz der Relation automatisch, dass ein Alias für die weitere Manipulation erforderlich ist und vergibt anschließend die Alias-Namen der Instanzen automatisch.

Abbildung 40 skizziert die die platzierten Elemente der einzelnen Instanzen der Relation *Mitarbeiter*. Der Alias der Instanzen wird neben dem Namen der Relation im Titel des Elementes dargestellt.

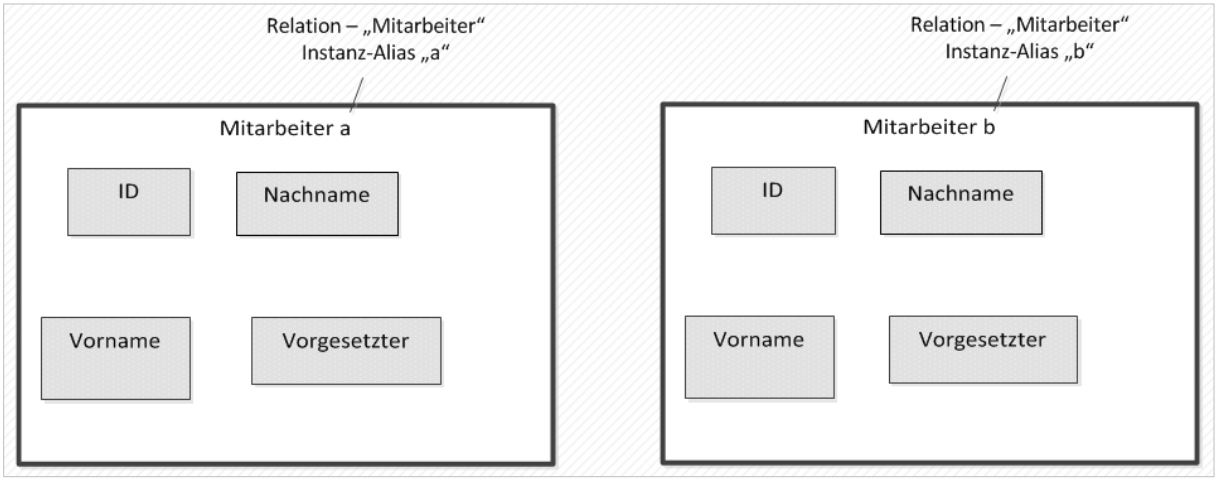


Abbildung 40 - QBG: Selbstverbund - Instanzen der Relation

Nachdem die Platzierung abgeschlossen ist, erfolgt die Formulierung der gewünschten Geste für die Verbundoperation. Diese Formulierung erfolgt wie in den Unterkapiteln 4.4.1, 4.4.2, 4.4.3, 4.4.4, 4.4.5 oder 4.4.6 dargestellt.

Die Darstellung an Hand des Beispiels ist in der nachfolgenden Abbildung 41 illustriert.

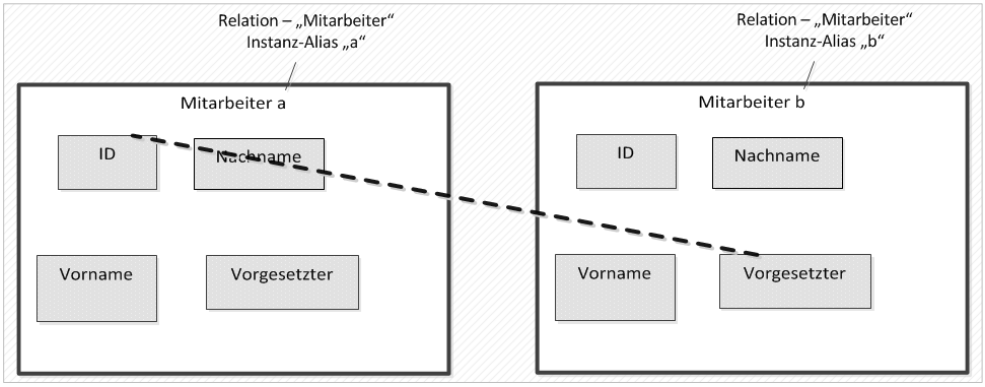


Abbildung 41 - QBG: Selbstverbund - Inner Join

4.5 Mengenoperationen

„Die klassischen Operationen auf Mengen sind die Vereinigung, der Durchschnitt und die Differenz. Diese Mengenoperationen sind auch in der Relationenalgebra enthalten, allerdings mit der Einschränkung, dass die Relationenschemata der beiden beteiligten Relationen gleich sein müssen. Diese Gleichheit ist zwar eine starke Einschränkung, sie kann jedoch oft durch die Umbenennungsoperation erreicht werden.“ [SSH13:101]

Bei allen Mengenoperationen gilt die Vereinigungskonformität. Diese besagt, dass die Anzahl der Attribute der Relationen übereinstimmen muss und die Datentypen der jeweils zu vereinigenden Attribute kompatibel sein müssen. Zudem müssen in der relationalen Algebra die Namen der Attribute übereinstimmen, was durch die automatische Umbenennung der Attribute durch das Datenbanksystem gewährleistet wird.

Da sowohl der QBE-Editor „Microsoft SQL-Server 2008 R2 Management Studio“, also auch „Microsoft Access 2010“ keine grafische Unterstützung zur Definition der Mengenoperationen bieten, wird im Folgenden auf den Vergleich der Interaktionstechniken mit QBE verzichtet.

4.5.1 Vereinigung

Die Vereinigung fügt die Tupel der beteiligten Relationen mit gleicher Attributanzahl und kompatiblem Datentyp in einer Ergebnisrelation zusammen.

Am Beispiel der Vereinigung der Relation K*unden* und den selektierten Attributen *Nachname* und *Vorname* mit der Relation *Mitarbeiter* und den selektierten Attributen *Nachname* und *Vorname* werden im Folgenden die Interaktionstechniken SQL und QBG gegenübergestellt.

SQL:

```
SELECT Kunden.Nachname, Kunden.Vorname FROM Kunden
UNION
SELECT Mitarbeiter.Nachname, Mitarbeiter.Vorname FROM Mitarbeiter
```

QBG:

Um die notwendige Gleichheit der Relationenschemata zu erreichen, werden in QBG die Attribute der Relationen in einer entsprechenden Reihenfolge ausgewählt. Die gewählte Reihenfolge wird mit einer Zahl (hier: „1“, „2“) am Attributelement jeder Relation angezeigt. Anschließend erfolgt der Wechsel der Ansichten der Relationen in die Tupelansicht. Im letzten Schritt wird eine Skalierung mit anschließendem Verschieben durchgeführt, so dass, analog zu Kapitel 4.4.1 Kartesisches Produkt (Cross Join) QBG, alle Elemente der Zielrelation (hier: *Mitarbeiter*) von dem Element der Quellrelation (hier: *Kunden*) überdeckt werden. Mit Abschluss dieser Verschiebeoperation, erfolgt eine automatische Aktualisierung der Ergebnismenge in der Tupelansicht des Elementes der Quellrelation. Die nachfolgenden Abbildungen skizzieren diesen Ablauf.

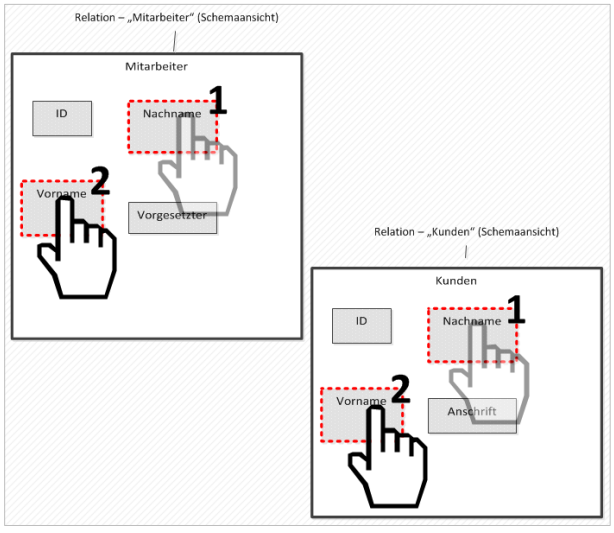


Abbildung 42 - QBG: Vereinigung – Attributauswahl

Abbildung 42 skizziert die Auswahl (Projektion) der Attribute der Relationen. Die Reihenfolge der Auswahl ist mit einer Ziffer in der rechten oberen Ecke jedes ausgewählten Attributes dargestellt.

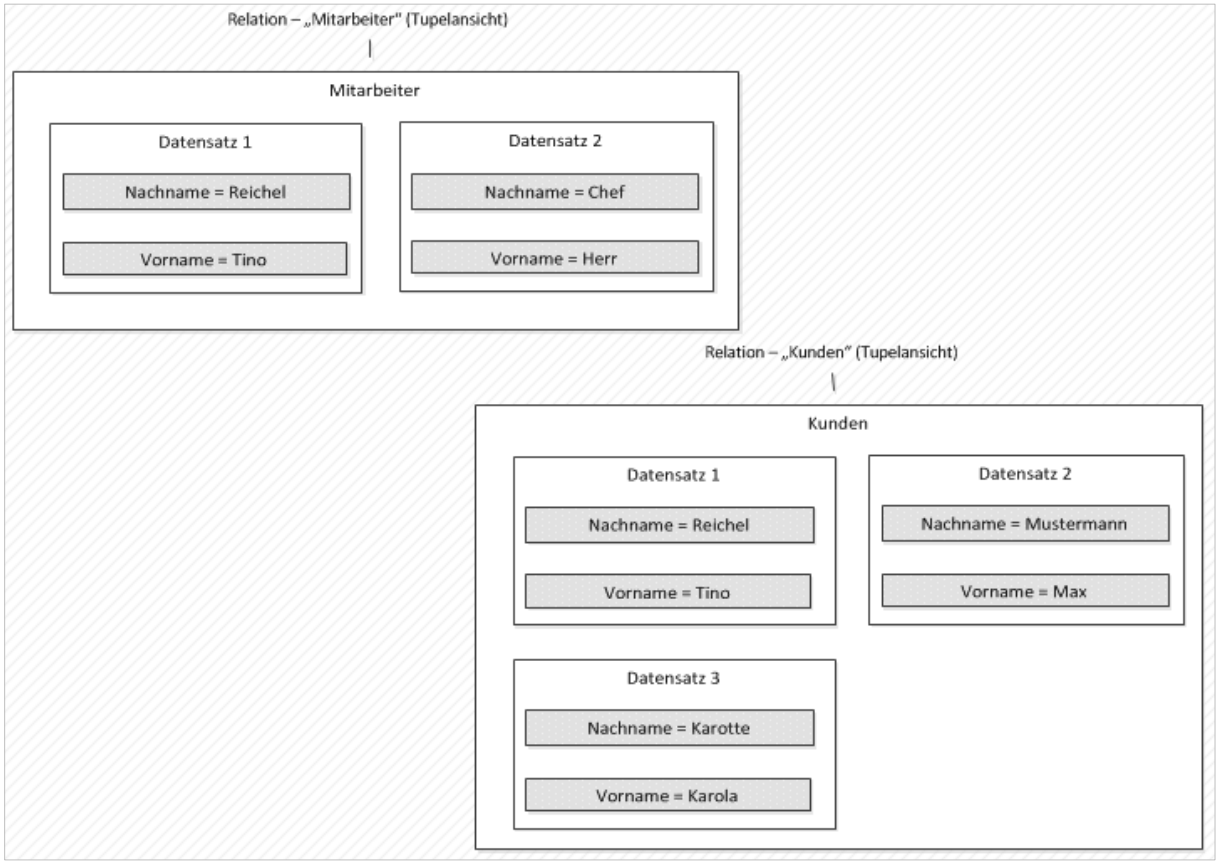


Abbildung 43 - QBG: Vereinigung – Tupelansicht

Der nachfolgende Schritt, der Wechsel in die Tupelansicht, ist in Abbildung 43 illustriert.

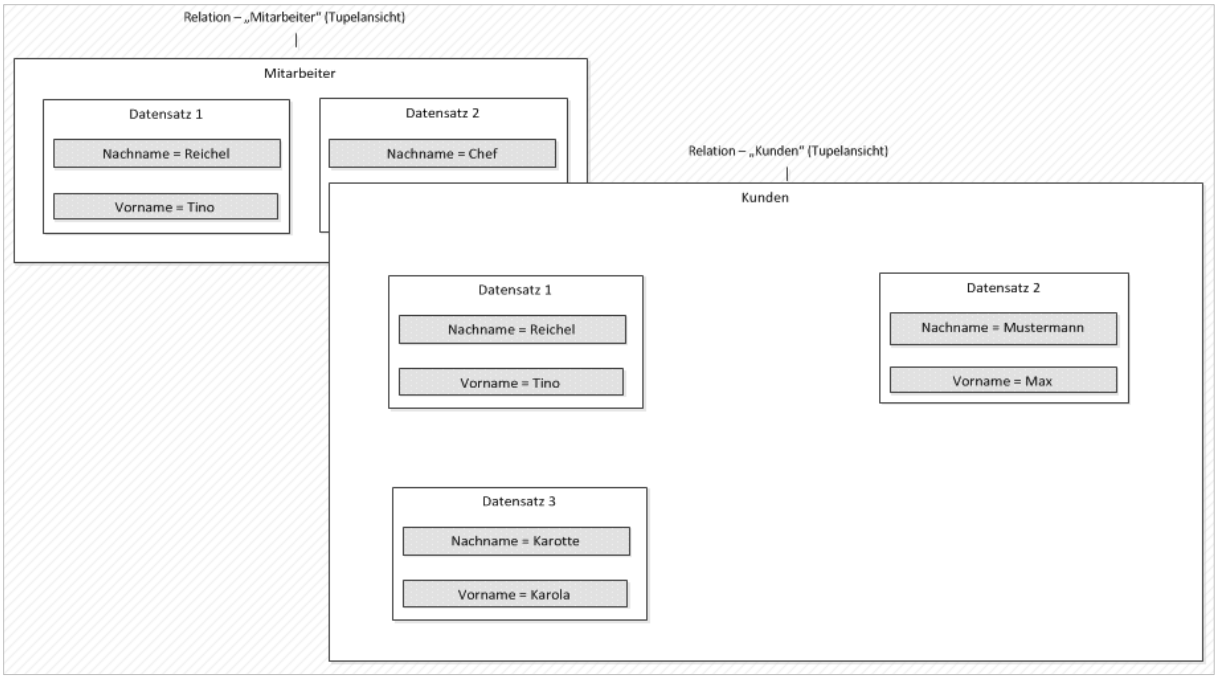


Abbildung 44 - QBG: Vereinigung - Tupelansicht skaliert

In Abbildung 44 erfolgt eine Skalierung des Relationselementes der Relation *Kunden*, so dass im Anschluss, Abbildung 45, dieses Relationselement über das der Relation *Mitarbeiter* verschoben werden kann.

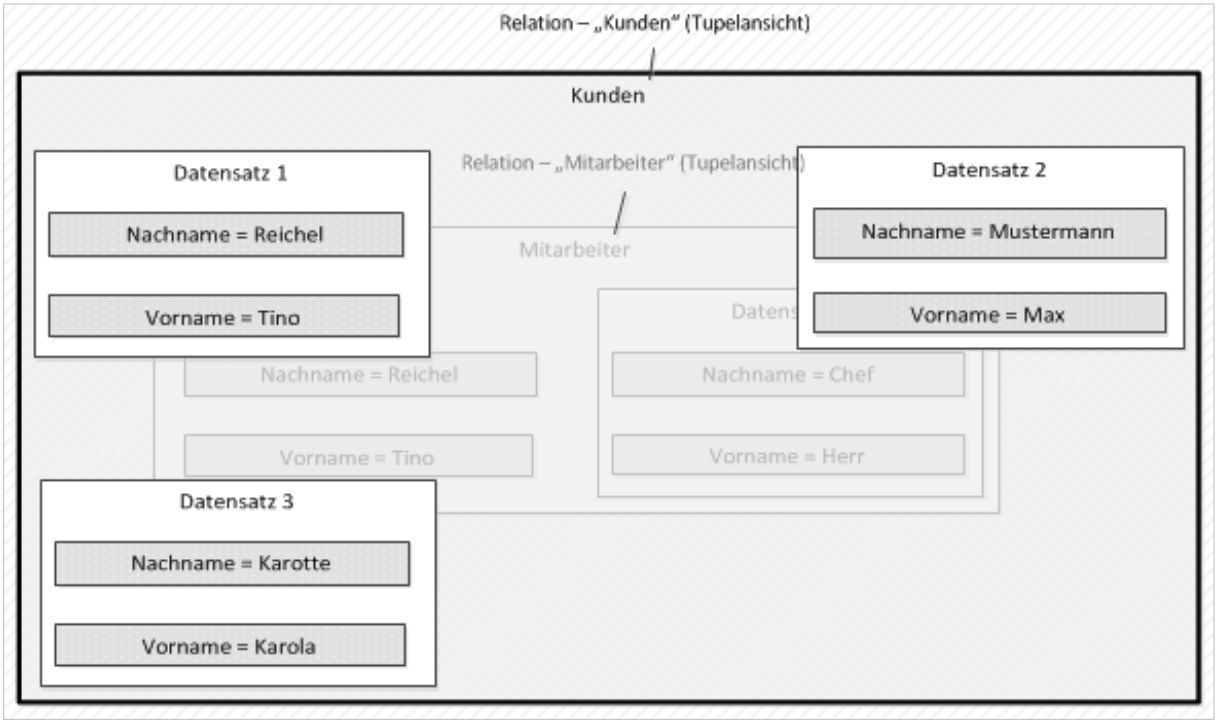


Abbildung 45 - QBG: Vereinigung - Tupelansicht verschoben

Das Relationselement *Kunden* überdeckt hier komplett das Relationselement *Mitarbeiter*. Somit entsteht die Vereinigung der Relationen *Kunden* und *Mitarbeiter*.

4.5.2 Durchschnitt (Schnittmenge)

Der Durchschnitt liefert die Tupel der Relationen, die gleich sind. Die mehrfachen Zeilen werden in der Ergebnismenge nicht angezeigt.

Am Beispiel des Durchschnitts der Relation K*unden* und den selektierten Attributen *Nachname* und *Vorname* mit der Relation *Mitarbeiter* und den selektierten Attributen *Nachname* und *Vorname* werden im Folgenden die Interaktionstechniken SQL und QBG gegenübergestellt.

SQL:

```
SELECT Kunden.Nachname, Kunden.Vorname FROM Kunden INTERSECT
SELECT Mitarbeiter.Nachname, Mitarbeiter.Vorname FROM Mitarbeiter
```

QBG:

Analog zu 4.5.1 Vereinigung QBG, erfolgt im ersten Schritt die Auswahl der Attribute und der Wechsel in die Tupelansicht. Anschließend wird ein Element so verschoben, dass der linke Teilbereich der Quellrelation (hier: *Kunden*) um maximal die Hälfte der Breite mit dem Element der Zielrelation (hier: *Mitarbeiter*) geschnitten wird. Diese Geste ist wiederum eine Analogie zu 4.4.4 Inklusionsverbund links (Left Join / Left Outer Join). Der Unterschied besteht lediglich in der Auswahl der Ansicht und darin, dass die Formulierung des Durchschnitts kein Selektionsprädikat erfordert. Die nachfolgende Abbildung 46 illustriert die Formulierung des Durchschnitts, wobei die Auswahl der Attribute und der Wechsel in die Tupelansicht analog zu Abbildung 42 und Abbildung 43 erfolgt, so dass diese nachfolgend nicht erneut aufgeführt werden.

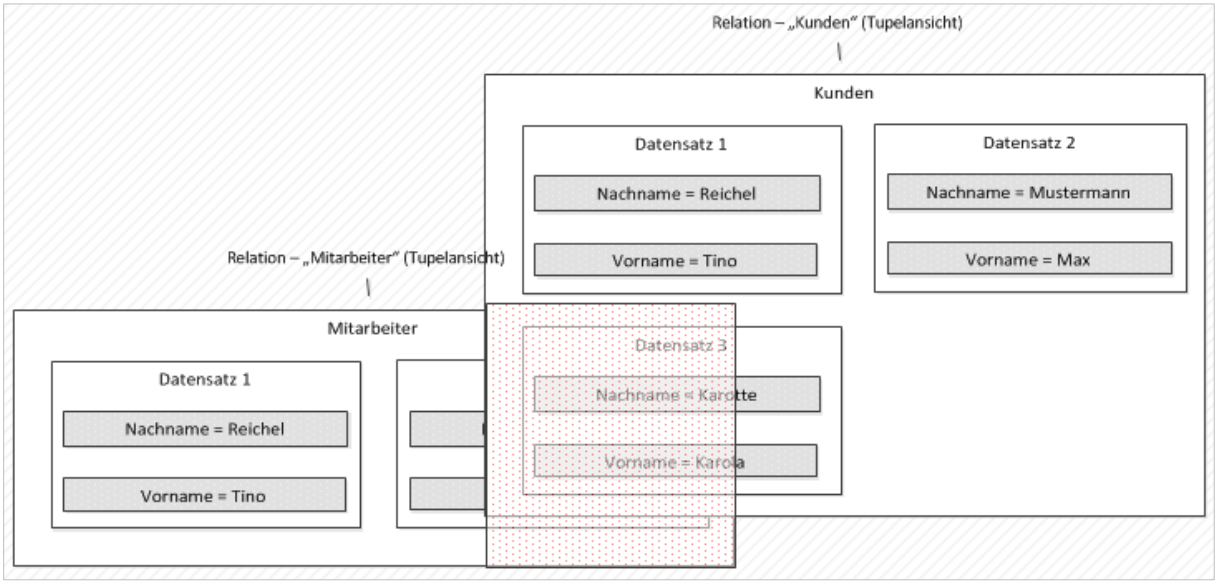


Abbildung 46 - QBG: Durchschnitt

4.5.3 Differenz

Die Differenz liefert die Tupel der Quellrelation, die in der Zielrelation nicht enthalten sind.

Am Beispiel des Durchschnitts der Relation K*unden* und den selektierten Attributen *Nachname* und *Vorname* mit der Relation *Mitarbeiter* und den selektierten Attributen *Nachname* und *Vorname* werden im Folgenden die Interaktionstechniken SQL und QBG gegenübergestellt.

SQL:

```
SELECT Mitarbeiter.Nachname, Mitarbeiter.Vorname FROM Mitarbeiter EXCEPT
SELECT Kunden.Nachname, Kunden.Vorname FROM Kunden
```

QBG:

Analog zu 4.5.1 Vereinigung QBG, erfolgt im ersten Schritt die Auswahl der Attribute und der Wechsel in die Tupelansicht. Anschließend wird ein Element so verschoben, dass der rechte Teilbereich der Quellrelation (hier: *Kunden*) um maximal die Hälfte der Breite mit dem Element der Zielrelation (hier: *Mitarbeiter*) geschnitten wird. Diese Geste ist wiederum eine Analogie zu 4.4.5 Inklusionsverbund rechts (Right Join / Right Outer Join). Der Unterschied besteht lediglich in der Auswahl der Ansicht und darin, dass die Formulierung der Differenz kein Selektionsprädikat erfordert. Die nachfolgende Abbildung 47 illustriert die Formulierung der Differenz, wobei die Auswahl der Attribute und der Wechsel in die Tupelansicht analog zu Abbildung 42 und Abbildung 43 erfolgt, so dass diese nachfolgend nicht erneut aufgeführt werden.

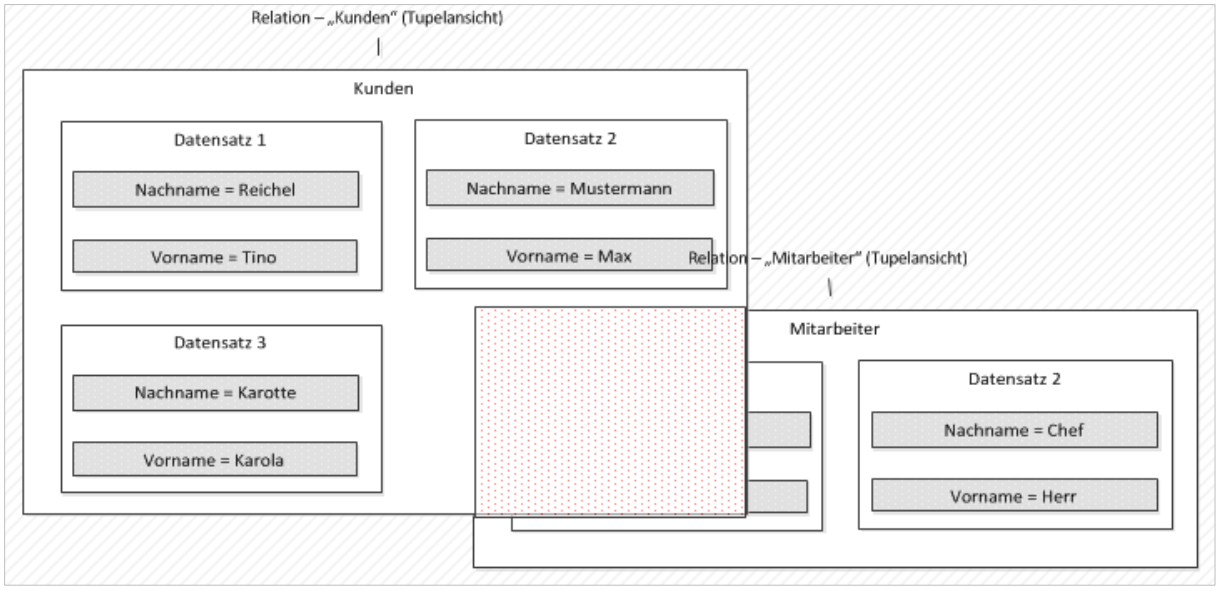


Abbildung 47 - QBG: Differenz

4.6 Verschachtelung

Verschachtelungen bieten die Möglichkeit innerhalb einer Anfrage eine weitere Unteranfrage zu formulieren. Das Ergebnis dieser inneren Anfrage kann anschließend als Filterkriterium der äußeren Anfrage dienen.

Im nachfolgenden Beispiel werden alle in Rechnung gestellten Artikel mit einem Preis größer als 30000€ gesucht. Hier liefert eine Unterabfrage der Relation *Artikel* eine Ergebnismenge aller Artikel mit einem Preis größer als 30000€. Diese Menge dient anschließend als Filterkriterium der Anfrage auf die Relation *RechnungsPositionen*. Das Ergebnis kann anschließend zu Statistikzwecken genutzt werden, aus dem man beispielsweise erkennt, wie häufig ein hochpreisiger Artikel (Preis > 30000€) verkauft wurde. Im Folgenden werden die Interaktionstechniken SQL, QBE und QBG an Hand dieses Beispiels gegenübergestellt.

SQL:

```
SELECT * FROM RechnungsPositionen
WHERE RechnungsPositionen.ArtikelID IN
      (SELECT ID FROM Artikel WHERE Artikel.Preis > 30000)
```

QBE:

Die nachfolgende Abbildung 48 illustriert die Formulierung der Verschachtelung in QBE. Zu beachten ist hier, dass die Formulierung der Unterabfrage im Tabellengerüst in der Spalte *Filter* erfolgt. Eine grafische Unterstützung zur Formulierung existiert hier nicht.

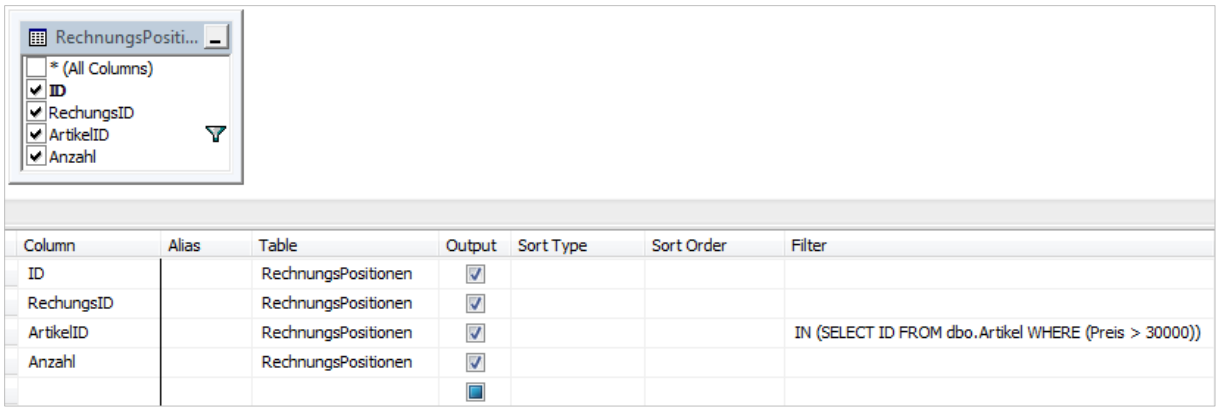


Abbildung 48 - QBE: Verschachtelung

QBG:

Die Formulierung einer verschachtelten Abfrage erfolgt in QBG von innen nach außen. Das heißt, dass zuerst die Unterabfragen mit den bekannten Gesten formuliert werden. Zu beachten ist, dass die Unterabfrage eine kompatible Ergebnismenge zurückliefern muss, so dass eine anschließende Selektion erfolgen kann.

Die Abbildung 49 skizziert die Platzierung der Relationen *RechnungsPositionen* und *Artikel*, sowie die Formulierung der Unterabfrage. Die Selektion (*Preis > 30000*) und Projektion (*Artikel.ID*) erfolgt über die bekannten Gesten aus den Kapiteln 4.1 Selektion und 4.2 Projektion.

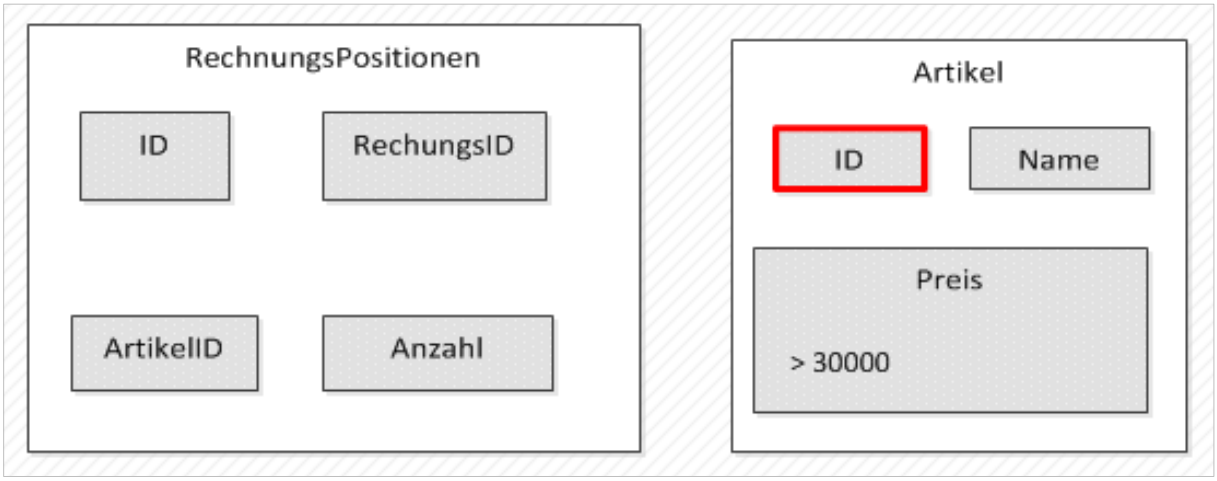


Abbildung 49 - QBG: Verschachtelung - Unterabfrage

Nachdem die Formulierung der Unterabfrage abgeschlossen ist, erfolgt ein Wechsel in die Tupelansicht. Abbildung 50 Illustriert diesen Schritt.

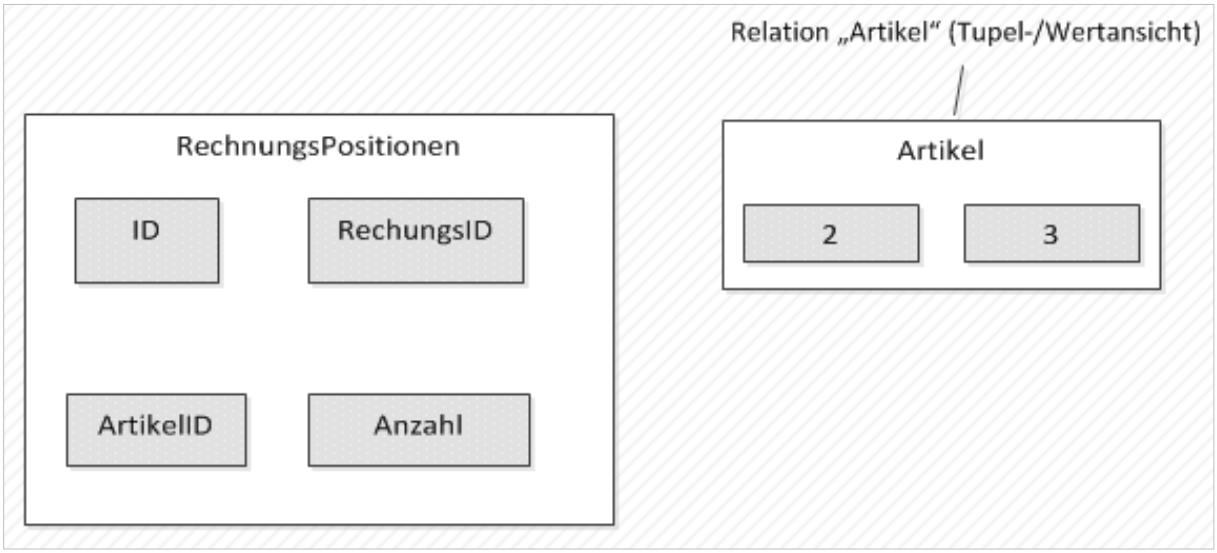


Abbildung 50 - QBG: Verschachtelung - Tupelansicht

Im letzten Schritt erfolgt die Formulierung der Selektionsbedingung der Relation *RechnungsPositionen*. Man unterscheidet hier, ob es sich um eine Vergleichsoperation oder eine Mengenoperation handelt.

Bei einem Vergleich erfolgt die Formulierung des Operators analog zu Kapitel 4.1 Selektion. Nach Erkennung des Vergleichsoperators wird nicht, wie bei der Selektion, der Vergleichswert in das Eingabefeld eingetragen. Hier wird eine Drag&Drop-Geste vom soeben ausgewählten Attribut (hier: *RechnungsPositionen.ArtikelID*) auf die Tupelansicht der inneren Abfrage (hier: *Artikel*) formuliert.

Bei einer Mengenoperation (hier: *IN*) erfolgt die Formulierung hingegen analog zu Kapitel 4.4.2 Gleichheitsverbund (Equivalent Join/Inner Join). Hier wird eine Drag&Drop-Geste vom ausgewählten Attribut (hier: *RechungsPositionen.ArtikelID*), ohne vorherige Selektionsbedingung, auf die Tupelansicht der inneren Abfrage (hier: *Artikel*) formuliert. Abbildung 51 Illustriert dieses Vorgehen.

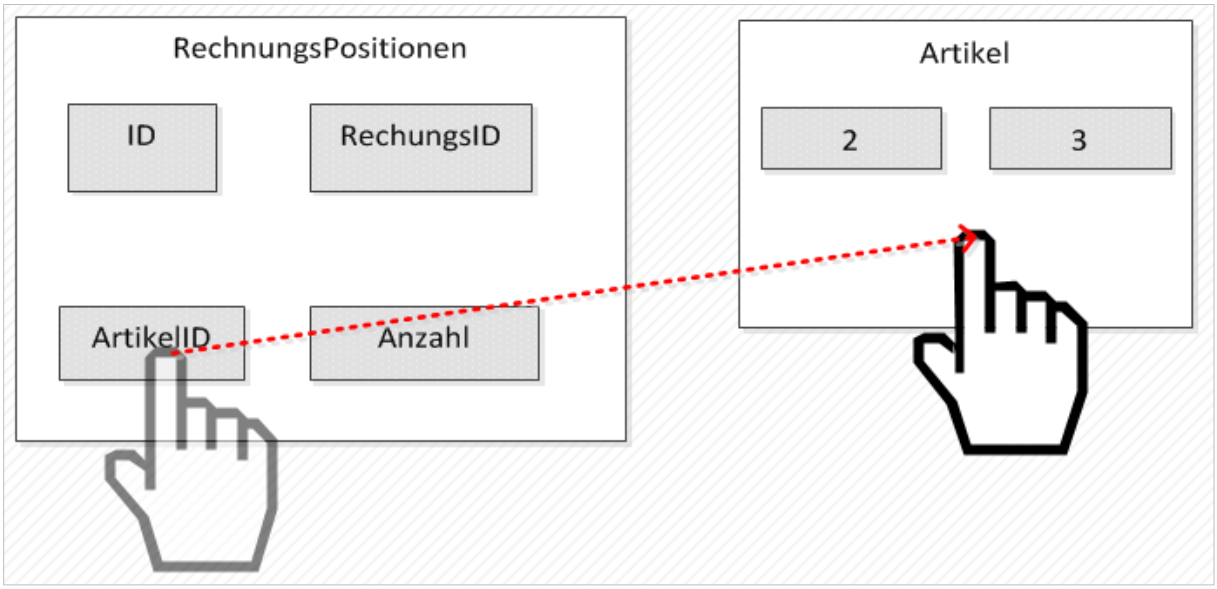


Abbildung 51 - QBG: Verschachtelung – Mengenoperation

In beiden Fällen erfolgt die Auswahl des Operators nach der Formulierung der Drag&Drop-Geste. Hier wird, analog zu Kapitel 4.4.2 Abbildung 28, der gewünschte Operator aus einer Liste verfügbarer Operatoren ausgewählt. Diese Liste wird für dieses Szenario mit Vergleichs- und Mengenoperatoren angepasst (Abbildung 52).

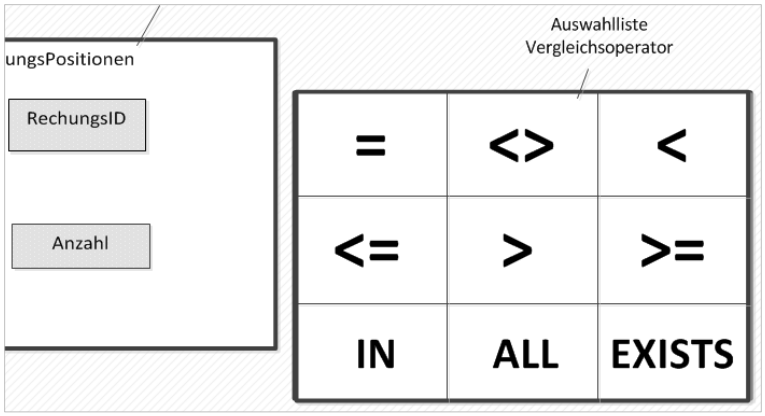


Abbildung 52 - QBG: Verschachtelung – Vergleichsoperatoren

Zusätzlich zu den in Kapitel 4.1 Tabelle 1 beschriebenen Vergleichsoperatoren werden die in Tabelle 3 aufgelisteten Mengenoperatoren unterstützt.

Mengenoperator	Beschreibung
IN	„Ermittelt, ob ein angegebener Wert mit einem Wert aus einer Unterabfrage oder Liste übereinstimmt.“ [MS13_IN]
ALL	„Vergleicht einen Skalarwert mit Werten, die sich in einer einzelnen Spalte befinden." [MS13_ALL]
EXISTS	„Gibt eine Unterabfrage an, die testet, ob Zeilen vorhanden sind.“ [MS13_EX]

Tabelle 3 - Mengenoperatoren

4.7 Sortierung

Die Sortierung ermöglicht eine Änderung der Reihenfolge der Tupel in der Ergebnisliste. Man unterscheidet hier zwischen aufsteigende und absteigende Sortierung.

Am Beispiel der Projektion aller Attribute der Relation *Kunden*, mit der aufsteigenden Sortierung nach dem Attribut *Alter*, werden die Interaktionstechniken SQL, QBE und QBG gegenübergestellt.

SQL:

```
SELECT * FROM Kunden
ORDER BY Alter
```

QBE:

Abbildung 53 illustriert die Formulierung der Sortierung in QBE. Hier wird im Tabellengerüst in Spalte *SortType* die Art der Sortierung (aufsteigend, absteigend) festgelegt.

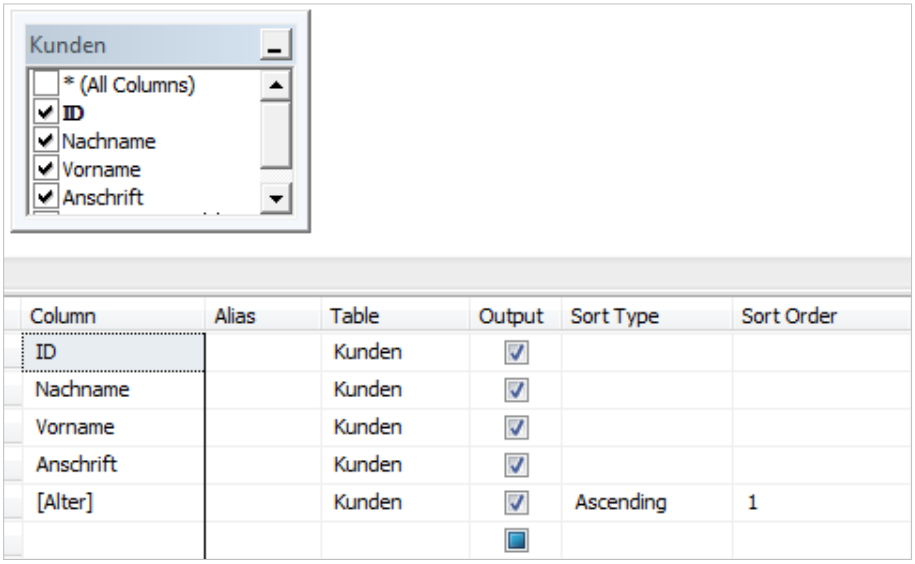


Abbildung 53 - QBE: Sortierung – Tabellengerüst

Eine weitere Möglichkeit die Sortierung festzulegen, bietet der QBE-Editor im Kontextmenü jedes Attributes in der Tabellenansicht (Abbildung 54).

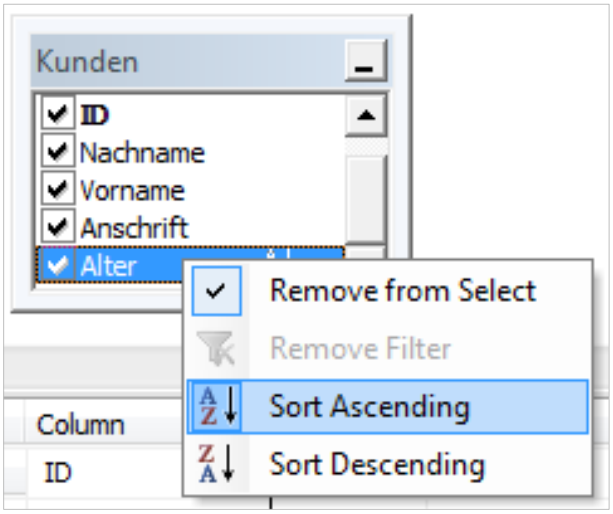


Abbildung 54 - QBE: Sortierung - Kontextmenü

QBG:

In QBG erfolgt die Formulierung der Sortierung in der Tupelansicht des Elementes der Ergebnisrelation. Um eine absteigende Sortierung nach einem Attribut zu formulieren, wird einer vertikale Wischbewegung von oben nach unten, über dem Element des Attributes, nach dem sortiert werden soll, durchgeführt. Start und Ende der Wischgeste müssen hierbei nicht im Element des Attributes liegen. Eine vertikale Wischbewegung von unten nach oben sortiert aufsteigend. Um eine zuvor definierte Sortierung zu löschen, erfolgt eine horizontale Wischbewegung auf dem Element, wobei diese Geste sowohl von links, als auch von rechts beginnen kann. Die nachfolgenden Abbildungen skizzieren die Formulierung einer Sortierung nach dem oben angegebenen Beispiel.

Abbildung 55 illustriert die Formulierung der aufsteigenden Sortierung der Werte der Relation *Kunden* nach dem Attribut *Alter*.

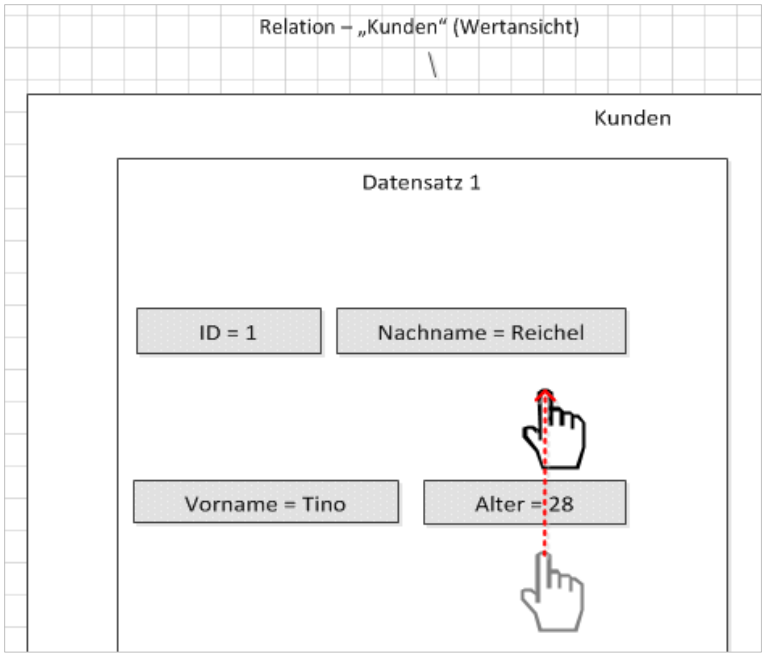


Abbildung 55 - QBG: Sortierung

Abbildung 56 skizziert die Darstellung einer zuvor formulierten Sortierung. Diese Abbildung beinhaltet zusätzlich zur aufsteigenden Sortierung nach dem Attribut *Alter* (Abbildung 55) eine zusätzliche absteigende Sortierung nach dem Attribut *Nachname*. Der rote Pfeil identifiziert die Art der Sortierung (aufsteigend oder absteigend), wobei die Ziffer auf dem Pfeil die Reihenfolge der Sortierkriterien darstellt.

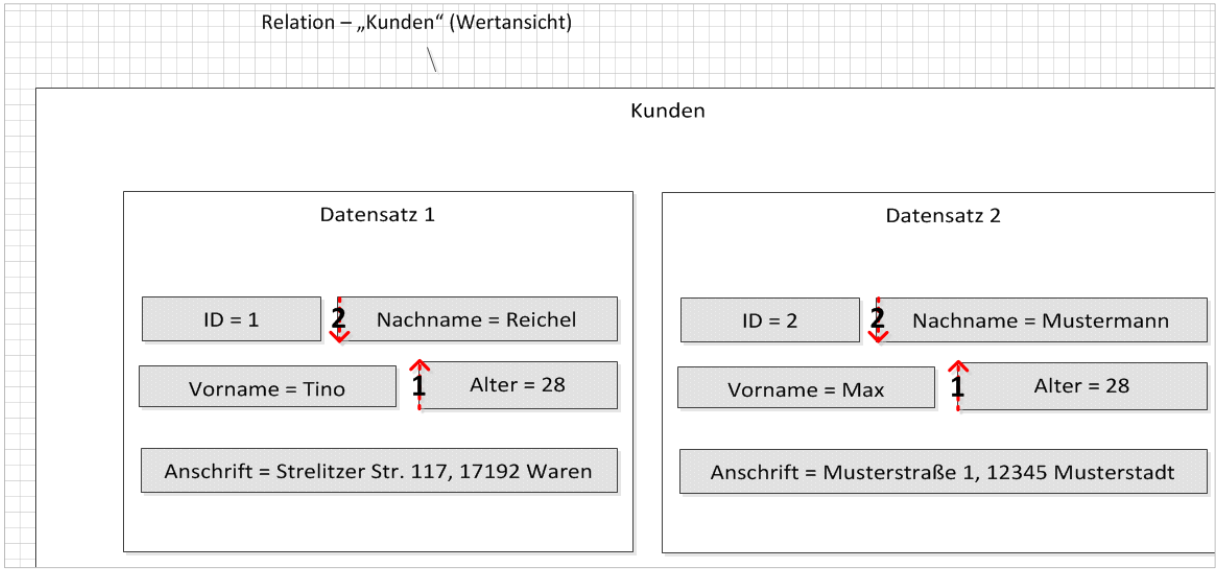


Abbildung 56 - QBG: Sortierung – Darstellung

Die in Abbildung 56 dargestellte Sortierung entspricht dem folgenden SQL-Ausdruck:

```
SELECT * FROM Kunden
ORDER BY Alter ASC, Nachname DESC
```

4.8 Gruppierung

Eine Gruppierung dient zur Zusammenfassung mehrerer Tupel einer Relation in jeweils eine Gruppe. Zu beachten ist, dass eine Gruppierung nur dann möglich ist, wenn die Projektion eine gruppenweise Auswertung zulässt. Dies ist wiederum der Fall, wenn eine Aggregatfunktion (Kapitel 4.2.1) über ein Attribut angewendet wird.

Am Beispiel der Gruppierung nach *Artikel.Name* mit Projektion des Attributes *Artikel.Name* und Projektion des Ergebnisses des Aggregats *SUM(RechnungsPositionen.Anzahl)* werden im Folgenden die Interaktionstechniken SQL, QBE und QBG gegenübergestellt.

SQL:

```
SELECT Artikel.Name, SUM(RechnungsPositionen.Anzahl)
FROM
      RechnungsPositionen
INNER JOIN
      Artikel ON RechnungsPositionen.ArtikelID = Artikel.ID
GROUP BY Artikel.Name
```

QBE:

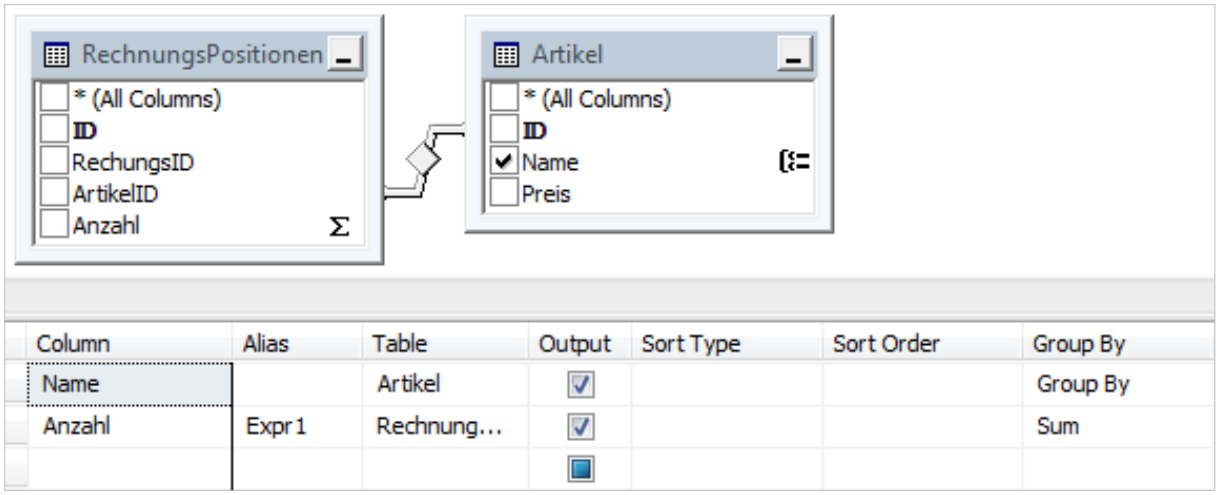


Abbildung 57 - QBE: Gruppierung

QBG:

Um eine Gruppierung in QBG zu formulieren, erfolgt im ersten Schritt die Auswahl des Attributes in der Schemaansicht des Relationselementes (hier: *RechnungsPositionen.Anzahl*). Nach der Auswahl mit einer Tippgeste wird die Aggregatfunktion *SUM* aus dem zur Verfügung gestellten Menü gewählt. Abbildung 58 skizziert die Auswahl der Attribute *Artkel.Name* und *RechnungsPositionen.Anzahl*, wobei über die Spalte *RechnungsPositionen.Anzahl* die Aggregatfunktion *SUM* ausgeführt werden soll.

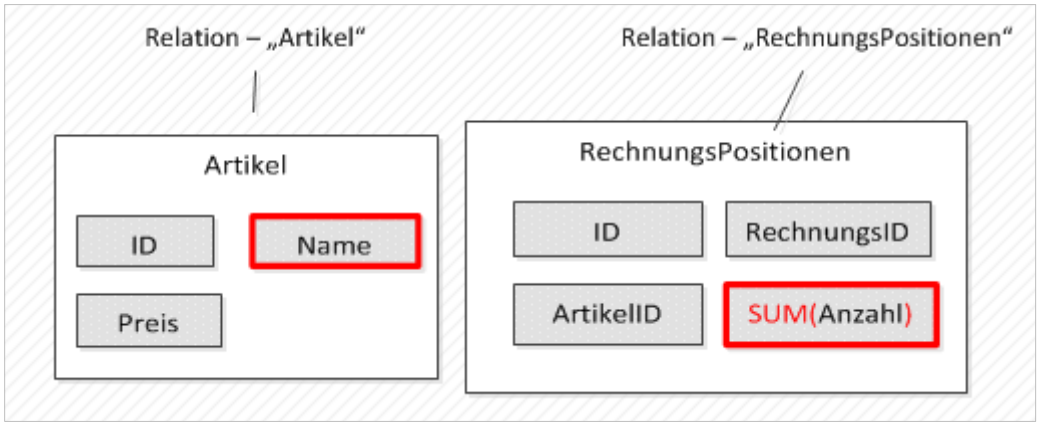


Abbildung 58 - QBG: Gruppierung – Projektion

Im nächsten Schritt erfolgt die Formulierung eines Gleichheitsverbunds (Inner Join – Kapitel 4.4.2) mit dem Verbundprädikat *RechnungsPositionen.ArtikelID = Artikel.ID*. Abbildung 59 illustriert die so entstandene Ansicht des Interaktionscontainers mit den Relationselementen *Artikel* und *RechnungsPositionen,* dem definierten Gleichheitsverbund und den pro Relation ausgewählten Attributen.

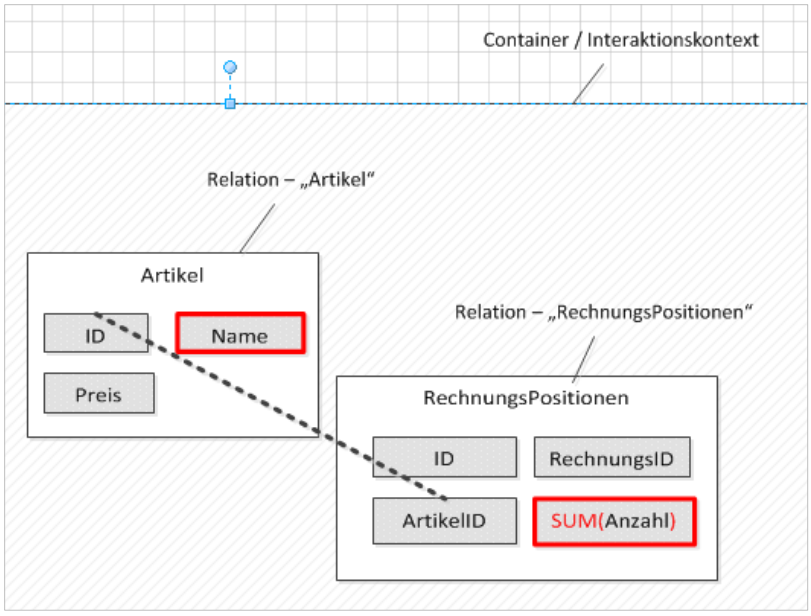


Abbildung 59 - QBG: Gruppierung – Schemaansicht

Nach den bisherigen Definitionen entsteht so die SQL-Abfrage:

```
SELECT Artikel.Name, SUM(RechnungsPositionen.Anzahl)
FROM
      RechnungsPositionen
INNER JOIN
      Artikel ON RechnungsPositionen.ArtikelID = Artikel.ID
```

Diese Abfrage führt bei der Ausführung zu einem Fehler, da hier eine Aggregatfunktion zur Projektion genutzt wird. Das Datenbanksystem kann solche Abfragen nur auswerten, wenn eine Gruppierung nach den Attributen der nicht aggregierten Attribute erfolgt.

Die Anwendungslogik von QBG erkennt die Notwendigkeit einer Gruppierung und vervollständigt automatisch die an das Datenbanksystem abzusetzende SQL-Abfrage. Mit dem Wechsel in die Tupelansicht des Relationselementes *RechnungsPositionen*, wird somit folgende SQL-Abfrage ausgeführt, die anschließend zur in Abbildung 60 skizzierten Ergebnisdarstellung führt:

```
SELECT Artikel.Name, SUM(RechnungsPositionen.Anzahl)
FROM
      RechnungsPositionen
INNER JOIN
      Artikel ON RechnungsPositionen.ArtikelID = Artikel.ID
GROUP BY Artikel.Name
```

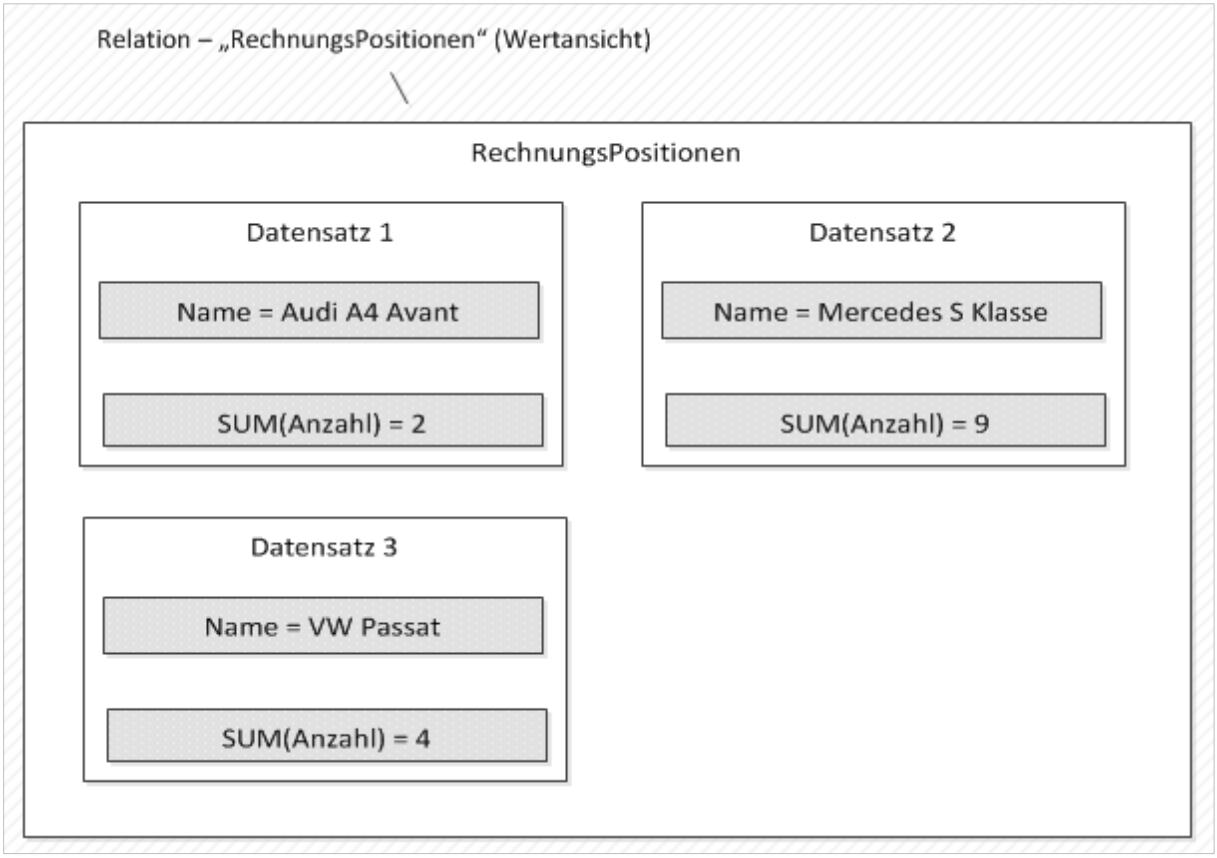


Abbildung 60 - QBG: Gruppierung – Tupelansicht

4.9 Fazit

Mit den in Kapitel 6 festgelegten Definitionen und Abfolgen können einfache Datenbankanfragen mittels Touch- und 3D-Gesten formuliert werden. Die hier festgelegten Spezifikationen erlauben sogar die Erstellung von Abfragen, die mit dem hier genutzten QBE-Editor, nicht ohne Eingabe der SQL-Syntax (Kapitel 4.5.2, 4.5.3), möglich ist. Trotzdem ist es auch hier nicht möglich, komplett auf Tastatureingaben verzichten zu können, da beispielsweise Selektionsbedingungen nicht zwingend eine endliche Menge darstellen und somit nicht in Menüs oder Listen abbildbar sind.

5 Spezifikation der Darstellung der Anfrageergebnisse

Bei der Darstellung der Anfrageergebnisse muss man verschiedene Kriterien berücksichtigen, die Einfluss auf die Spezifikation nehmen. Eine Ausnahme bildet hier die Darstellung der Ergebnisse einer Anfrage, die genau eine Relation beinhaltet. Hier kann grundsätzlich das Ergebnis der Anfrage in der Tupelansicht des gewählten Relationselementes angezeigt werden.

Handelt es sich bei einer formulierten Anfrage um eine Anfrage auf mehreren Relationen, so gibt es verschiedene Möglichkeiten das Gesamtergebnis darzustellen.

Nehmen wir an, dass ein Anwender alle gekauften Artikel mit einem Preis größer als 30.000€ des Kunden *Max Mustermann* als Anfrageergebnis erhalten möchte. Um dieses Ergebnis darzustellen, formuliert der Nutzer die Selektionsbedingung *Kunden.Nachname = 'Mustermann'* auf dem Attribut *Nachnahme* der Relation *Kunden* (Kapitel 4.1 Selektion), um in einer Zwischenanfrage den bestimmten Kunden zu erhalten. Eine weitere Selektionsbedingung, *Artikel.Preis > 30000,* führt auf dem Relationselement *Artikel* zu dem zweiten Zwischenergebnis. Abbildung 61 skizziert diese formulierten Anfragen zum Erhalt der beiden Zwischenergebnisse.

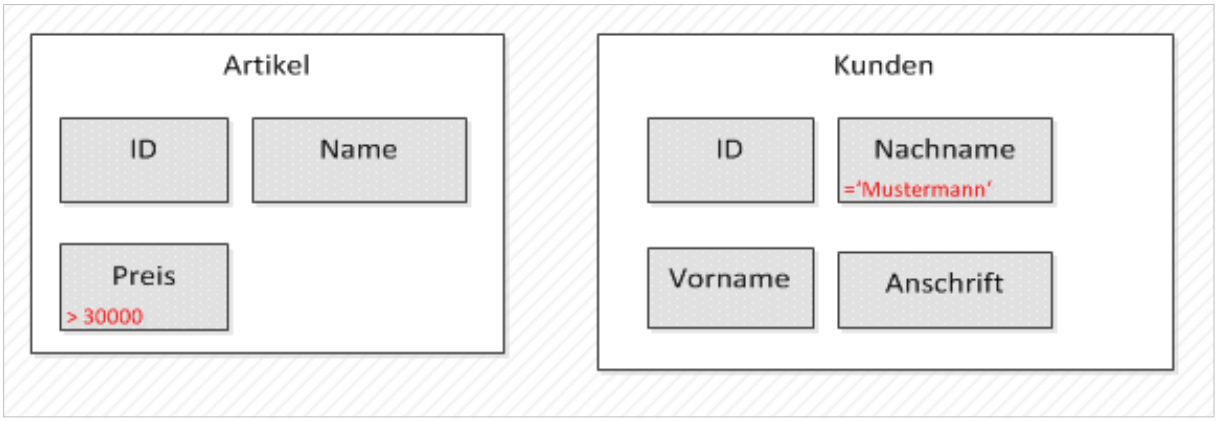


Abbildung 61 - Ergebnisdarstellung: Zwischenergebnisse

Da es in der Beispieldatenbank keine direkte Beziehung zwischen der Relation *Kunden* und *Artikel* gibt, müssen nachfolgend die Relationen *Rechnungen und Rechnungspositionen* auf dem Interaktionscontainer platziert werden, um die Beziehung mit Hilfe von Verbundoperationen (Kapitel 4.4 Verbund (Join)) darzustellen. Die nachfolgende Abbildung 62 illustriert diese platzierten Verbundrelationen unterhalb der zuvor platzierten Elemente.

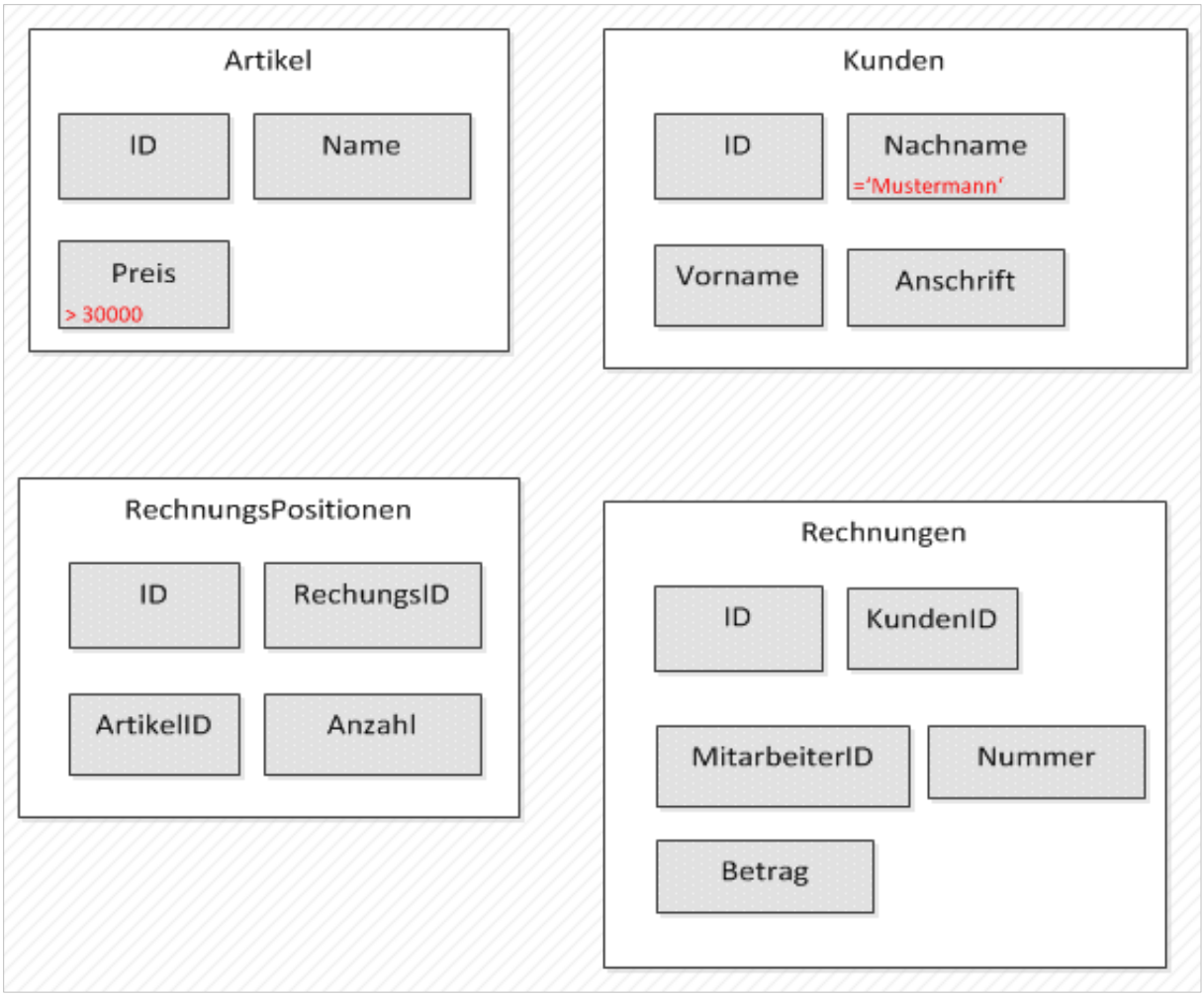


Abbildung 62 - Ergebnisdarstellung: Platzierte Verbundrelationen

Der Nutzer muss nun entsprechende Verbundoperationen ausführen, um eine Beziehung zwischen der Relation *Artikel* und der Relation *Kunden* herzustellen. Eine abgeschlossene Formulierung dieser Operationen ist in der nachfolgenden Abbildung 63 skizziert.

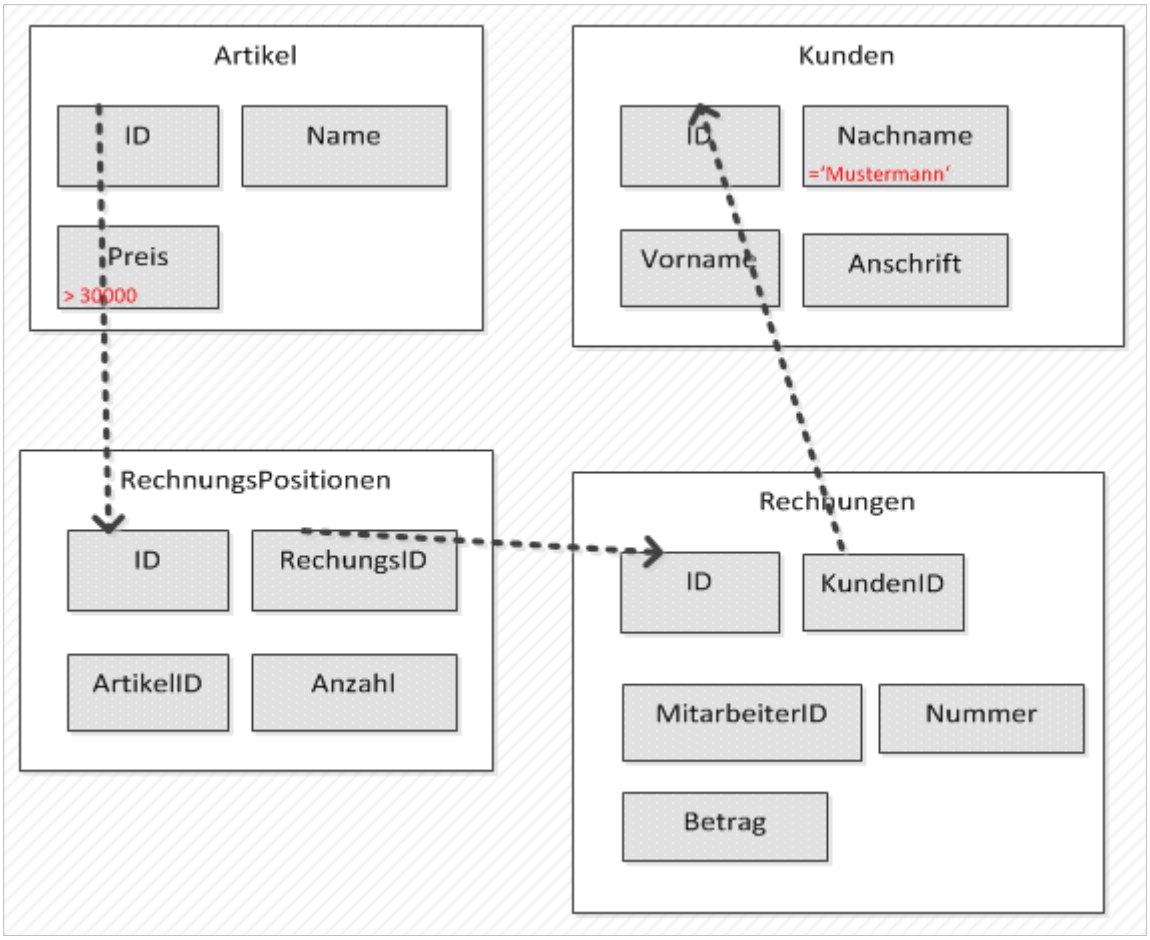


Abbildung 63 - Ergebnisdarstellung: Verbundene Relationen

Die in Abbildung 63 formulierte Anfrage wirft die Frage auf, wo das Gesamtergebnis dieser Anfrage dargestellt wird.

Einige dieser Darstellungsmöglichkeiten werden in den nachfolgenden Unterkapiteln kurz erläutert.

5.1 Anfrageergebnisse im ersten Relationselement

Die Anzeige der Anfrageergebnisse im ersten auf dem Interaktionscontainer positionierten Relationselement stellt eine Möglichkeit dar. Diese ist jedoch inkonsistent, da der Anwender vor der Formulierung der Anfrage entscheiden muss, welches Element das Gesamtergebnis anzeigen soll. Eine Lösung hierfür ist die automatische Erstellung einer Ergebnisrelation. Analog zum Verhalten einiger QBE-Editoren, werden alle positionierten Relationen mittels Kreuzprodukt verbunden und in der Ergebnisrelation dargestellt. Wenn Manipulationen an den Relationen vorgenommen werden, dann wird die Ergebnisrelation entsprechend aktualisiert.

Der Nachteil dieser Lösung ist, dass man keine sofortigen Zwischenergebnisse zu Verfügung hat und somit nicht mit diesen weiterarbeiten kann.

5.2 Anfrageergebnisse im Quellrelationselement

Eine zweite Möglichkeit ist die Anzeige eines Anfrageergebnisses in der Quellrelation. Somit können auch Zwischenergebnisse genutzt werden. Um aber wiederum Gesamtergebnisse darstellen zu können, muss bei der Formulierung einer Verbundoperation die Richtung der Geste beachtet werden. Da die Quellrelation das Ergebnis darstellt, muss die Geste demzufolge immer von der Quellrelation zur Zielrelation formuliert werden.

Der Nachteil dieser Lösung ist, dass der Anwender bei der Formulierung der Anfragen auf die Richtung der zu formulierenden Gesten achten muss.

5.3 Problemfall – Ringverbund

Grundsätzlich existiert der Problemfall eines möglichen Ringverbundes. Dieses Problem kann auftreten, wenn eine Relation als Quellrelation genutzt wird und anschließend, im weiteren Verlauf der Anfrageformulierung, diese Relation als Zielrelation ausgewählt wird. Die Lösung dieses Problems ergibt sich aus der Spezifikation, dass keine Quellrelation im weiteren Verlauf der Anfrageformulierung als Zielrelation genutzt werden darf. Diese Regel muss bei der Implementierung der Spezifikation beachtet werden.

Lösung:

Durch die Möglichkeit, eine Relation (genauer: eine Instanz einer Relation) mehrfach auf den Interaktionscontainer platzieren zu können, sind solche Anfragen dennoch möglich.

5.4 Fazit

Neben den zwei genannten Möglichkeiten zur Ergebnisdarstellung existieren weitere Ansätze, die noch spezifiziert und dokumentiert werden müssen. Wie in den Kapiteln 5.1 und 5.2 werden aber auch diese Darstellungsformen Vor- und Nachteile gegenüber den Lösungsansätzen haben.

Eine konkrete Lösung der Darstellungsproblematik bleibt in der vorliegenden Arbeit offen, wobei der Ansatz aus Kapitel 5.2 bei der Implementierung des Prototyps genutzt wird.

6 Implementierung / Prototyp

Auf der Grundlage der Spezifikation aus den vorherigen Kapiteln, führt die Umsetzung zu der hier dokumentierten Anwendung. Hierbei handelt es sich um einen Prototyp, der nicht den vollen Funktionsumfang der Spezifikation liefert.

6.1 Entwicklungsumgebung

Als Entwicklungsumgebung des Prototyps dient die Anwendung *Microsoft Visual Studio 2012 Professional.* Hiermit steht ein Werkzeug zur Verfügung, das mit Hilfe von Steuerelementen und Bibliotheken zur Auswertung der Kamera-Sensoren [KIN13] erweitert werden kann. Des Weiteren werden Zusatzkomponenten der Bibliothek *Microsoft Surface SDK 2.0* ***Es ist eine ungültige Quelle angegeben.*** verwendet. Diese Steuerelemente sind speziell für die Touch-Bedienung auf größeren Geräten entwickelt worden und komplettieren somit die Komponentenpalette der eingesetzten Entwicklungsumgebung.

6.2 Struktur

Die Implementierung des Prototyps basiert auf dem MVVM-Design-Pattern [MVVM13]. Auf Grund dieses Patterns erfolgt die Aufteilung der einzelnen Programmkomponenten auf verschiedene *Microsoft Visual Studio*- Projekte.

Die nachfolgende Abbildung 64 illustriert die vorhandenen Projekte.

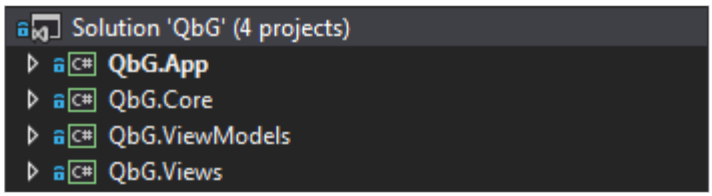


Abbildung 64 - Prototyp: Struktur

6.2.1 QbG.App

Das Projekt *QbG.App* definiert die ausführbare Datei des Prototyps. Hier werden alle weiteren Projekte referenziert. Hier wird sowohl der Haupteinsprungspunkt der Anwendung, als auch die Liste der Konfigurationsparameter definiert. Die in dem Projekt vorhandenen Dateien können der Abbildung 65 entnommen werden.

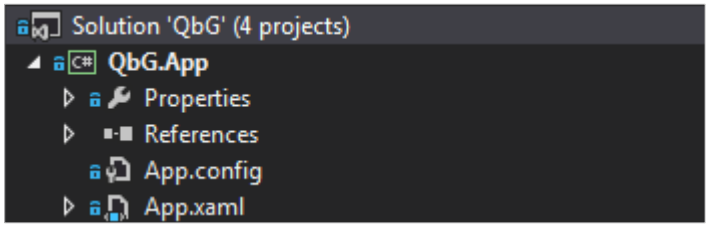


Abbildung 65 - Prototyp: QbG.App

6.2.2 QbG.Core

In dem Projekt QbG.Core werden Klassen und Methoden zur Datenbankkonnektivität und Gestenerkennung definiert. Demzufolge wird hier die Implementierung der Gestenerkennung an Hand des Hidden Markov Models [WF12:28 - 31] mit Hilfe des Accord.NET-Frameworks [ACC13] verwendet. Die in dem Projekt vorhandenen Dateien können der Abbildung 66 entnommen werden.

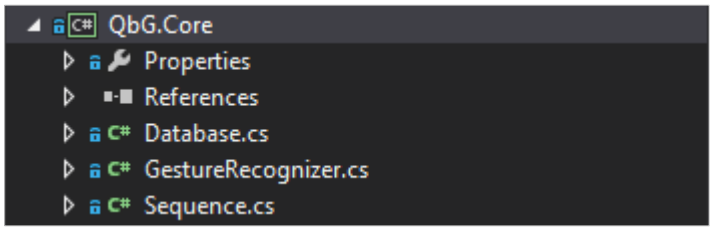


Abbildung 66 - Prototyp: QbG.Core

6.2.3 QbG.ViewModels

Dieses Projekt beinhaltet alle Dateien, in denen die Konstrukte zur Datenhaltung, Datenmanipulation und zur Ereignisverarbeitung definiert sind. Beispielsweise existiert hier die Klasse *TableViewModel*. Diese beinhaltet die Methoden und Eigenschaften eines Relationselementes. Somit werden hier die Methoden definiert, die bei der Manipulation eines Relationselementes ausgeführt werden. Die in dem Projekt vorhandenen Dateien können der Abbildung 67 entnommen werden.

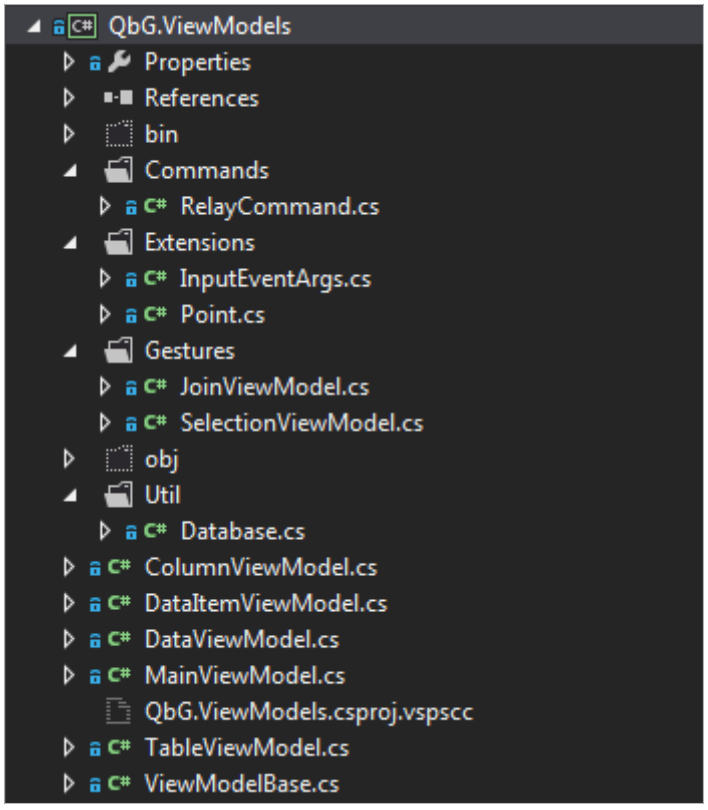


Abbildung 67 - Prototyp: QbG.ViewModels

6.2.4 QbG.Views

In dem Projekt *QbG.Views* sind alle Steuerelemente des Prototyps definiert. Hier erfolgt die Definition des Interaktionscontainers (*MainView.xaml*), der Relationselemente (*TableView.xaml*) und der in den Relationselementen unterschiedlichen Sichten. Es wird die Schemaansicht in der Datei *ColumnView.xaml* und die Tupelansicht in der Datei *DataView.xaml* definiert. Die einzelnen Wertansichten in der Tupelansicht werden durch eine Instanz des Elementes aus *DataItemView.xaml* dargestellt. Diese fünf unterschiedlichen Steuerelemente/Ansichten ermöglichen den Aufbau des Protoyps und werden durch weitere Steuerelemente für die Auswahl der Vergleichsoperatoren erweitert. Die in diesem Projekt vorhandenen Dateien können der Abbildung 68 entnommen werden.

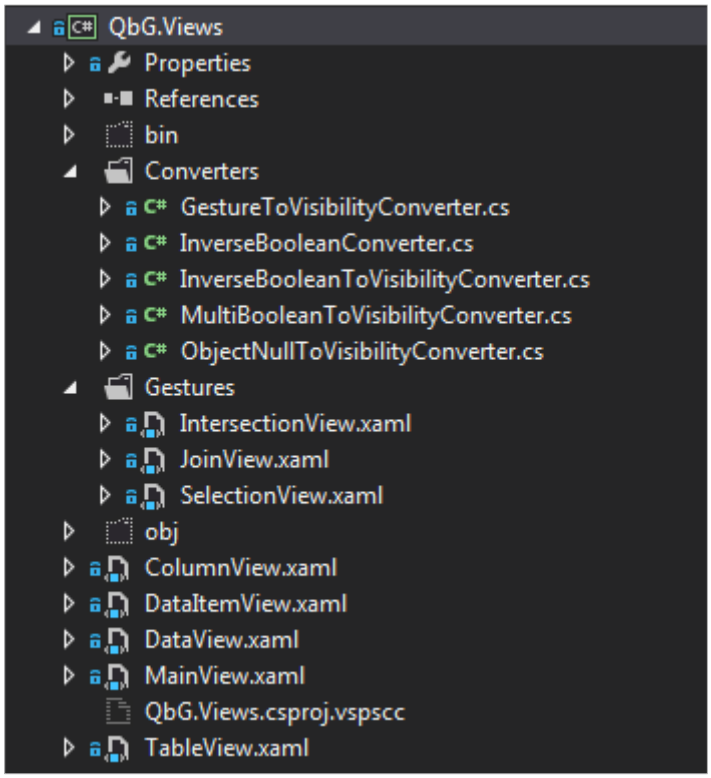


Abbildung 68 - Prototyp: QbG.Views

6.3 Funktionsweise

An Hand der Beispielanfrage aus Kapitel 4.4.2, wird in diesem Kapitel die Funktionsweise des Prototyps kurz dargestellt.

6.3.1 Interaktionscontainer und Auswahlliste

Der Interaktionscontainer ist das Hauptformular der Anwendung und wird direkt nach dem Start des Prototyps dargestellt. Der Anwender blendet, mittels einer Tippgeste auf den rechten Rand des Formulars, die Auswahlliste der verfügbaren Relationen ein. In Abbildung 69 ist das Hauptformular mit eingeblendeter Auswahlliste illustriert.

Abbildung 69 - Prototyp: Interaktionscontainer mit Auswahlliste

6.3.2 Positionierung der Relationselemente

Um eine Relation zur Erstellung einer Anfrage auf dem Interaktionscontainer zu platzieren, erfolgt die Auswahl und das anschließende Platzieren der gewünschten Relation mit Hilfe einer Drag&Drop-Geste von der Auswahlliste auf einen freien Bereich des Interaktionscontainers. In der nachfolgenden Abbildung 70 wird die Relation *Kunden* per Drag&Drop-Geste auf den Interaktionscontainer platziert. Anschließend erfolgt die Platzierung der Relation *Mitarbeiter* mit Hilfe der gleichen Geste.

Abbildung 70 - Prototyp: Positioniertes Relationselement

Abbildung 71 stellt die anschließende Positionierung des Elementes der Relation *Mitarbeiter* dar.

Abbildung 71 - Prototyp: Relationselemente

6.3.3 Ausführung der Operation (Gleicheitsverbund / Inner Join)

In der Beispielanfrage aus Kapitel 4.4.2 wird ein Gleichheitsverbund zwischen den beiden Relationen auf den Attributen *Kunden.Nachname* und *Mitarbeiter.Nachname* formuliert. Diese Formulierung erfolgt auf Grundlage der Spezifikation im Prototyp mittels einer Wischgeste. Diese Geste beginnt auf dem Verbundattribut der Quellrelation (hier: *Kunden.Nachname*) und endet auf dem Verbundattribut der Zielrelation (hier: *Mitarbeiter*). Nach Beendigung der Wischgeste wird dem Anwender eine Auswahlliste von Verbundoperatoren zur Verfügung gestellt, so dass er, mittels Tippgeste auf eines der Operatoren, die Erstellung des Verbundes abschließen kann. Die nachfolgende Abbildung 72 illustriert die Auswahl des Verbundoperators im Prototyp.

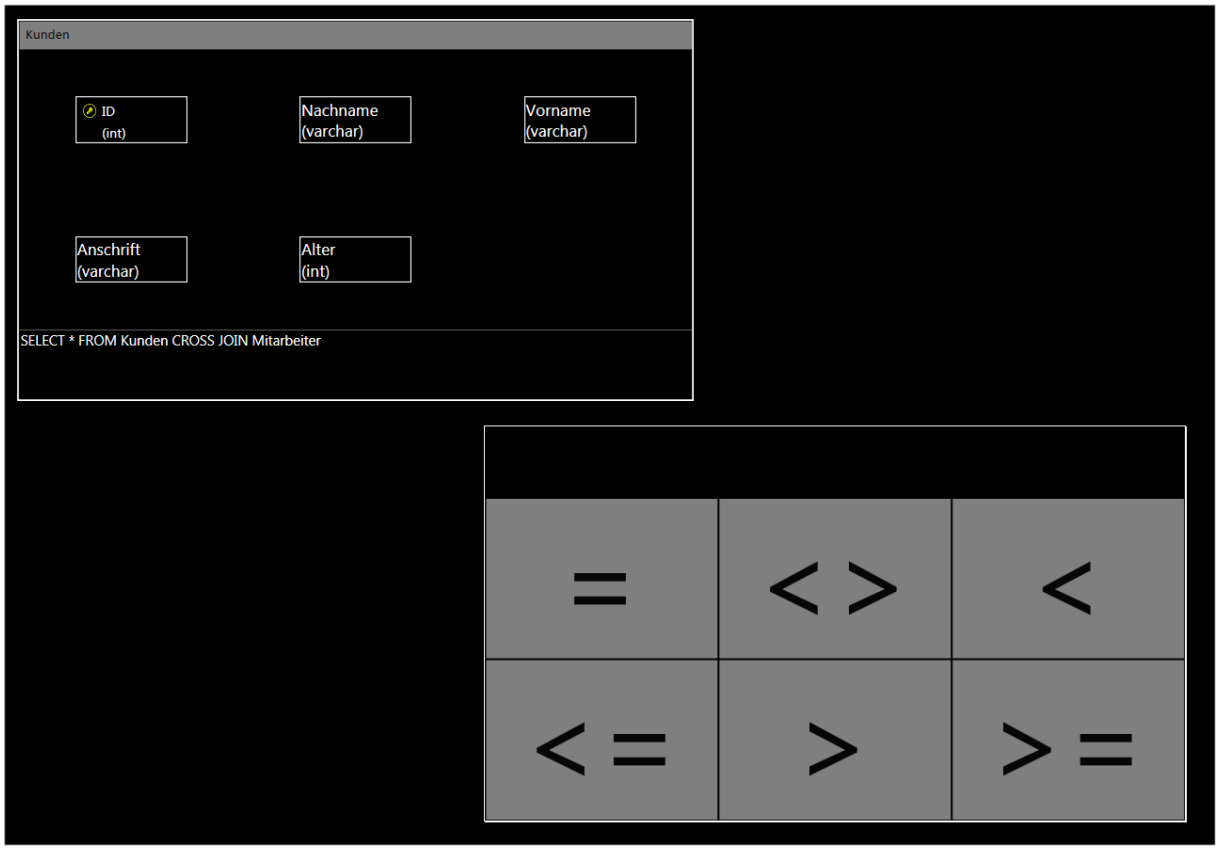


Abbildung 72 - Prototyp: Auswahl Verbundoperator

Es erfolgt die Auswahl des Gleichheitsoperators („=“) mit Hilfe einer Tippgeste, wodurch anschließend die Erstellung der Verbundoperation abgeschlossen wird. Eine Verbindungslinie zwischen den Verbundattributen stellt im Anschluss diese formulierte Verbundoperation auf dem Interaktionscontainer dar (Abbildung 73).

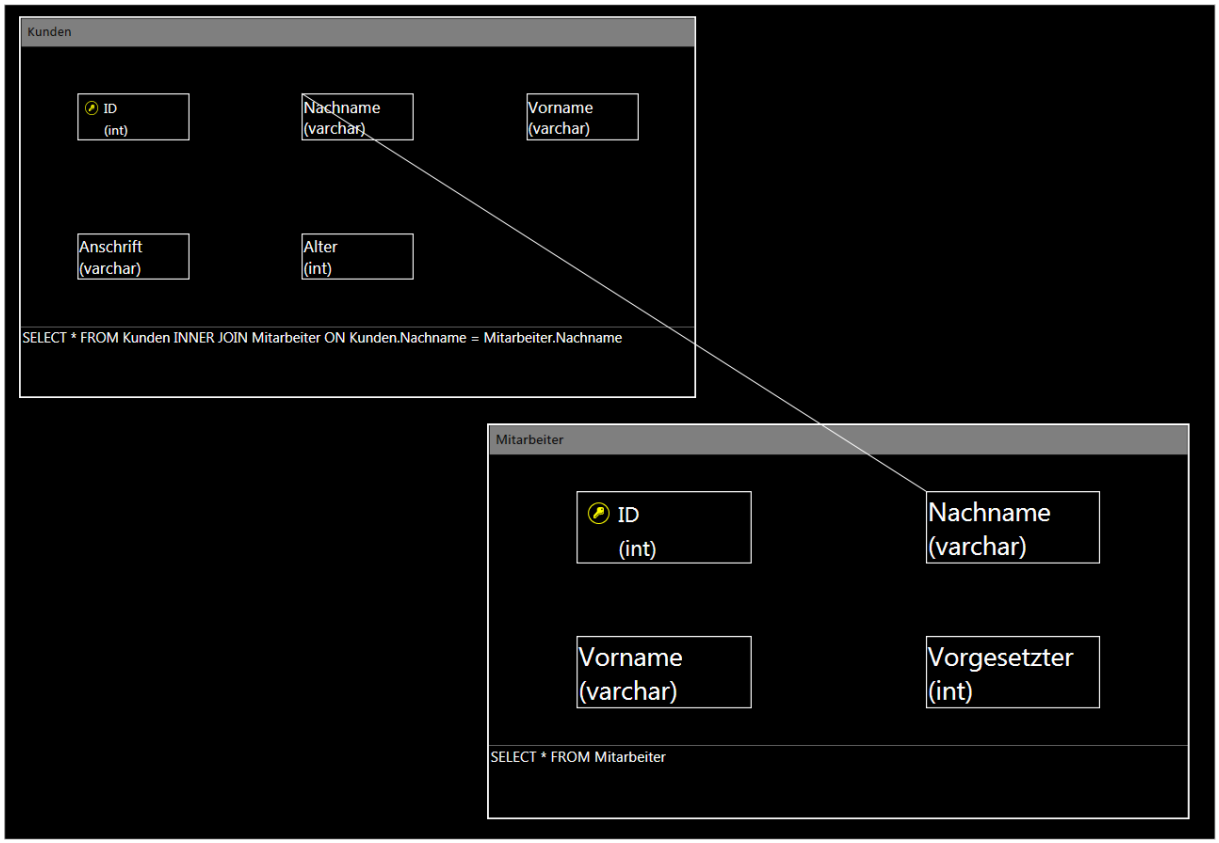


Abbildung 73 - Prototyp: Formulierte Verbundoperation

6.3.4 Ergebnisdarstellung

Die Ergebnisdarstellung erfolgt im Prototyp nach dem Prinzip der Darstellung in der Quellrelation (Siehe Kapitel 5.2), da diese die am schnellsten zu implementierende Darstellungsmöglichkeit ist und im Rahmen des Prototyps ausreichend erprobt wurde.

Da die Relation Kunden im Beispiel die Quellrelation ist und von dem in diesem Relationselement enthalten Attributelement *Nachname* die Geste zur Verbundoperation begann, erfolgt die Ergebnisdarstellung im Relationselement *Kunden*. Wie in Kapitel 3.2.2 spezifiziert führt eine Tippgeste auf die Titelleiste des Relationselementes zum Ansichtswechsel. Da das Relationselement im Standardfall, also auch hier, die Schemaansicht darstellt, führt die Tippgeste zur Anzeige der Tupelansicht. Diese Ansicht ist in Abbildung 74 dargestellt und beinhaltet einen Datensatz/Tupel mit allen Attributen und den entsprechenden Werten aus den beiden Relationen.

Abbildung 74 - Prototyp: Ergebnis Verbundoperation

6.3.5 Darstellung der SQL-Anfrage

Aus Gründen der Validierung und Nachvollziehbarkeit der Ergebnisse, wurde im Prototyp, sowohl in der Schemaansicht als auch in der Tupelansicht, ein Textfeld implementiert, welches die von der Anwendungslogik generierte SQL-Anfrage anzeigt. Aus Gründen der Vollständigkeit ist dieses Textfeld in der nachfolgenden Abbildung 75 zusätzlich dargestellt.

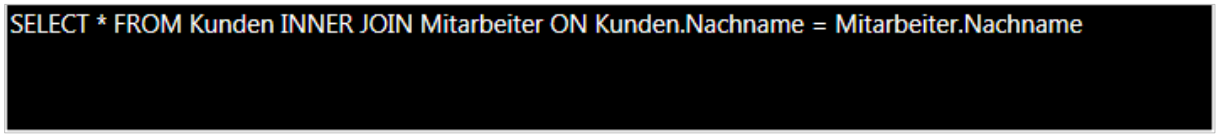


Abbildung 75 - Prototyp: Anzeige der SQL-Syntax

7 Zusammenfassung

7.1 Fazit

Durch die Spezifikation von *Query-by-Gesture (QBG)* ist eine neue Anfragesprache auf Datenbanken entstanden. Die Spezifikation der Anfragesprache, sowie die Spezifikation einer geeigneten Interaktionsumgebung mittels des hier definierten Interaktionscontainers und den Interaktionselementen stellt gleichzeitig eine Alternative zur Anfrageformulierung in QBE dar. Die definierten Gesten und die Darstellung der Relationen und Operationen ermöglichen die einfache und schnelle Erstellung von Datenbankanfragen mittels der modernen Interaktionsmöglichkeiten.

Im Vergleich zu QBE benötigt der Nutzer hier keine Kenntnisse der Anfragesprache SQL. Die Nutzung, einer auf QBG basierten Anwendung, erfordert lediglich ein Grundverständnis für die Definition von Anfragen auf relationale Datenmengen. Es werden intuitive, und aus dem heutigen alltäglichen Umgang mit modernen Interaktionstechniken, bekannte Gesten zur Navigation innerhalb der Anfragegestaltung und Definition der Anfrageoperationen genutzt.

Zusammenfassend bietet QBG eine neue Möglichkeit, ohne Programmierkenntnisse zu einer bestimmten Programmier- oder Anfragesprache und mittels bekannter Gesten (Wischen, Zoomen, Verschieben,…) aus einer großen Datenmenge Informationen zu filtern, zusammenzufassen und darzustellen.

Zielgruppe zur Nutzung von QBG könnte somit der Nutzerkreis der derzeitigen QBE-Anwendungen sein. Des Weiterem kann eine QBG-Anwendung die Akzeptanz und Nutzerfreundlichkeit zum Beispiel bei der Recherche innerhalb einer großen Datenbank (zum Beispiel: Bibliothek) steigern, da die potentiellen Anwender hier eine Schnittstelle zur Verfügung gestellt bekommen, die sie aus ihrem Alltag bereits kennen.

7.2 Ausblick

Da es auf Grund der festgelegten Definitionen der Gesten zu vielen Überschneidungen der Interaktionselemente auf dem Interaktionscontainer kommen kann, wird eine Interaktionstechnik benötigt, die eine zuvor formulierte Anfrage als Zwischenergebnis (ähnlich einer *Sicht* in SQL) speichern kann. Diese neue, temporäre Relation könnte anschließend wie eine „normale“ Relation auf dem Interaktionscontainer positioniert und manipuliert werden.

Um die Akzeptanz und Nutzerfreundlichkeit noch weiter zu steigern, könnte die automatische Erstellung von Verbunden dienen. Hierzu müsste die QBG-Anwendung an Hand der Attribute der zu verbindenden Relationen automatisch ein Verbundprädikat auswählen, ohne dass der Anwender hier reagieren muss.

Des Weiteren wäre es sinnvoll den Interaktionscontainer so zu erweitern, dass man Konstrukte wie Schleifen, Bedingungen und Variablen erstellen kann. Hierdurch würde man die Grundlagen zur Erstellung von Stored Procedures (SPs) und User Defined Functions (UDFs) schaffen, die wiederum in Selektionsbedingungen und Projektionen nutzbar wären.

Durch Nutzung des 3D-Sensors *Microsoft Kinect* und der integrierten Spracherkennung ist es auch denkbar, dass die Auswahl bestimmter Vergleichs- und Mengenoperatoren mit Hilfe der Sprache durchgeführt wird. Diesbezüglich müssten „Wörterbücher“ mit den benötigten Schlüsselwörtern definiert werden, so dass je nach Anwendungsfall die entsprechenden Schlüsselwörter zur Auswahl stehen.

Trotz aller Verbesserungsvorschläge besitzt die QBG-Spezifikation einen hohen Funktionsumfang und lässt sich in den Arbeitsbereichen von QBE einsetzen. Mit eventueller Umsetzung der hier genannten Vorschläge, könnte das Einsatzfeld von QBG vergrößert, die Funktionalität erweitert und das Arbeiten mit QBG noch weiter erleichtert werden.

I. Tabellenverzeichnis

II. Abbildungsverzeichnis

III. Literaturverzeichnis

Alle nachfolgend gelisteten Internetquellen sind auf dem beiliegenden Datenträger offline zur Verfügung gestellt. Zudem sind die Vorlesungsskripte und Berichte (Berichte, Projektarbeiten, Bachelorarbeiten, …) ebenso auf dem beigefügten Datenträger hinterlegt.

[ACC13] (2013): *Accord.NET Framework*. http://code.google.com/p/accord/, zuletzt geprüft am 19.06.2013.

[BBBH13] Beier F.; Baumann S.; Betz H. et al. (2013): *Gesture-Based Navigation in Graph Databases - The Kevin Bacon Game* -. (GI-Edition Lecture Notes in Informatics S. 511 - 514) Ilmenau University of Technology, Germany.

[IAO12] Spath D.; Weisbecker A.; Laufs U. et al. (2012): *Multi-Touch Technologie, Hard-/Software und deren Anwendungsszenarien*. (Studie) Stuttgart: Fraunhofer IAO.

[IIY13] *IIYAMA*. http://www.iiyama.com/gl_en/products/prolite-t2452mts-1/, zuletzt geprüft am 20.06.2013.

[JNM13] Jiang L.; Nandi A. und Mandel M. (2013): *Youtube - GestureQuery: A Multitouch Database Query Interface*. http://www.youtube.com/watch_popup?v=P9GP_P3FBLs&vq=hd720, zuletzt geprüft am 23.Juni.2013.

[KIN13] (2013): *KINECT for Windows*. Microsoft. http://www.microsoft.com/en-us/kinectforwindows/develop/developer-downloads.aspx, zuletzt geprüft am 21.Juni.2013.

[LG13] *LG 23ET83V*. http://www.lg.com/de/monitore/lg-23ET83V, zuletzt geprüft am 13.Juni.2013.

[MH11] Meyer H. (Vorlesung “Multimedia Datenbanken”, WS 2010/2011): *Inhaltsbasierte Bildretrieval-Systeme*. Universität Rostock.

[MS13_ALL] *msdn - ALL (Transact-SQL)*. http://msdn.microsoft.com/de-de/library/ms178543.aspx, zuletzt geprüft am 23.Juni.2013.

[MS13_EX] *msdn - EXISTS (Transact-SQL)*. http://msdn.microsoft.com/de-de/library/ms188336.aspx, zuletzt geprüft am 23.Juni.2013.

[MS13_IN] *msdn - IN (Transact-SQL)*. http://msdn.microsoft.com/de-de/library/ms177682.aspx, zuletzt geprüft am 23.Juni.2013.

[MS13_SO] *msdn - Surface Overview*. http://msdn.microsoft.com/en-us/library/ff727864.aspx, zuletzt geprüft am 22.Juni.2013.

[MVVM13] *MSDN Magazin*. http://msdn.microsoft.com/de-de/magazine/dd419663.aspx, zuletzt geprüft am 22.Juni.2013.

[NA13_1] Nandi A. (2013): *Querying Without Keyboards*. (Bericht) The Ohio State University, Computer Science and Engineering.

[NA13_2] Nandi A. (2013): *The Interactive Join: Recognizing Gestures for Database Queries*. (Bericht) The Ohio State University, Computer Science and Engineering.

[ONI13] *OpenNI*. http://www.openni.org/, zuletzt geprüft am 21.Juni.2013.

[QBG02] Byun H. und Ko B.C. (2002): *dblp - computer science bibliography - Journal of Visual Languages and Computing, Volume 13, 2002*. http://dblp.uni-trier.de/db/journals/vlc/vlc13.html#KoB02, zuletzt geprüft am 23.Juni.2013. Query-by-Gesture: An Alternative Content-Based Image Retrieval Query Scheme.

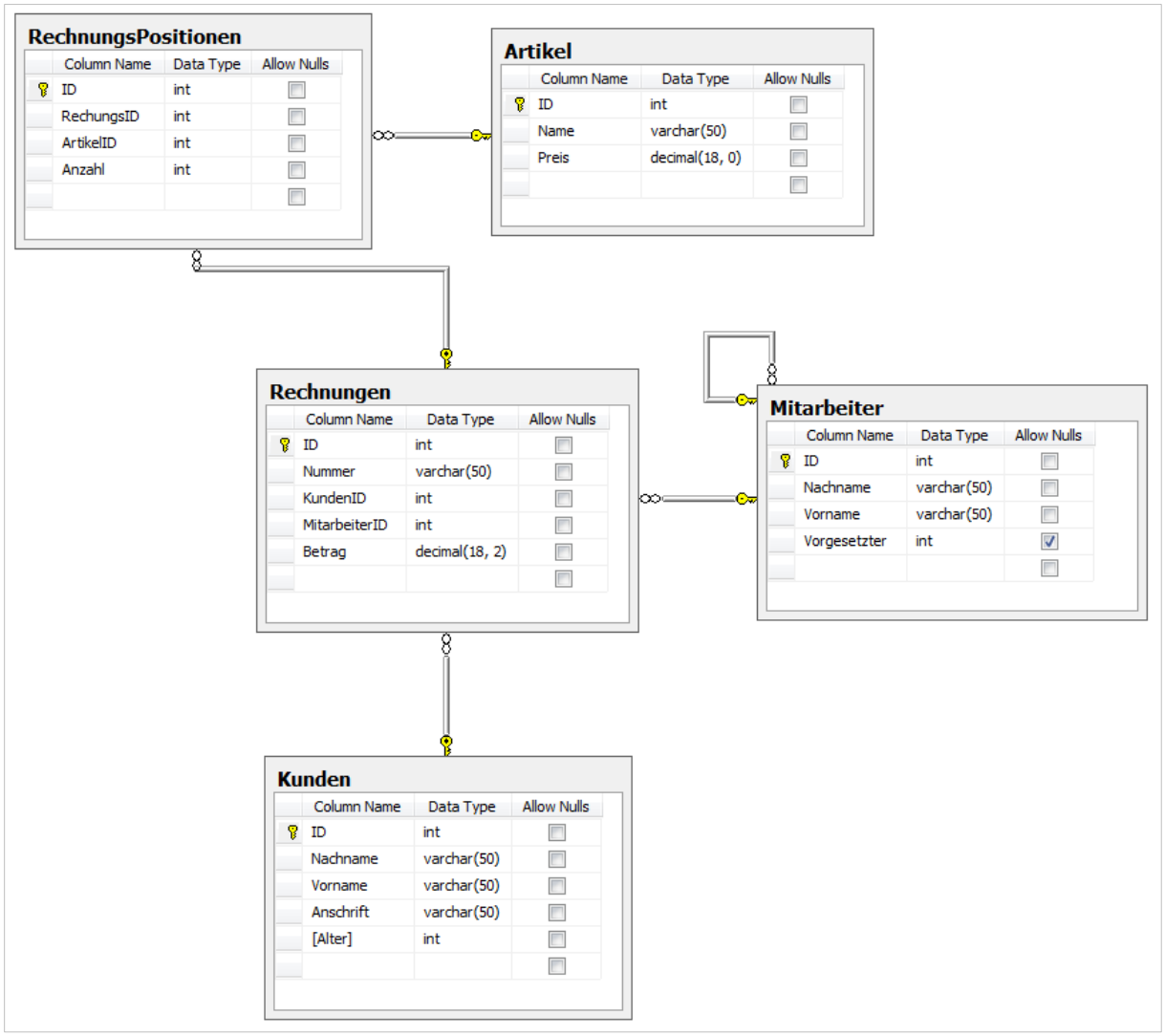
[SSH13] Saake G.; Sattler K.-U. und Heuer A. (2013): *Datenbanken - Konzepte und Sprachen*. 5. Aufl. mitp.

[SWY13] *Swype - Type Fast, Swype Faster*. http://www.swype.com/, zuletzt geprüft am 23.Juni.2013.

[WF12] Winter F. (2012): *Konzeption und Umsetzung eines Gestenerkennungssystems für Intelligente Umgebungen auf Basis von 3D-Kamerasensorik*. (Projektarbeit) Universität Rostock, Fakultät für Informatik und Elektrotechnik, Institut für Informatik, Lehrstuhl für Mobile Multimediale Informationssysteme.

IV. Anhang

Beispieldatenbank – Schema



Beispieldatenbank – Daten Tabelle „Kunden“

ID	Nachname	Vorname	Anschrift	Alter
1	Reichel	Tino	Strelitzer Str. 11...	28
2	Mustermann	Max	Musterstraße 1, ...	32
3	Karotte	Karola	Auf dem Feld 1, ...	1
NULL	NULL	NULL	NULL	NULL

Beispieldatenbank – Daten Tabelle „Mitarbeiter"

ID	Nachname	Vorname	Vorgesetzter
1	Reichel	Tino	2
2	Chef	Herr	NULL
NULL	NULL	NULL	NULL

Beispieldatenbank – Daten Tabelle „Artikel"

ID	Name	Preis
1	VW Passat	26000
2	Audi A4 Avant	32000
3	Mercedes S Klasse	65000
4	VW Golf	23000
5	Opel Adam	15000
6	Dacia Logan	8000
NULL	NULL	NULL

Beispieldatenbank – Daten Tabelle „Rechnungen"

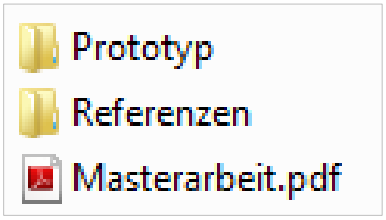
ID	Nummer	KundenID	MitarbeiterID	Betrag
1	00001	1	1	90000,00
2	00002	2	1	403000,00
3	00003	3	2	260000,00
NULL	NULL	NULL	NULL	NULL

Beispieldatenbank – Daten Tabelle „RechnungsPositionen"

ID	RechungsID	ArtikelID	Anzahl
1	1	1	1
2	1	2	2
3	2	1	3
4	2	3	5
5	3	3	4
NULL	*NULL*	*NULL*	*NULL*

V. Datenträger

Der beigefügte Datenträger beinhaltet folgende Orderstruktur:



Das Wurzelverzeichnis beinhaltet dieses Dokument in digitaler Form (PDF).

Im Verzeichnis P*rototyp* sind sowohl die Beispieldatenbank als Backup-Datei, als auch alle Quellcode-Dateien des Prototyps. Zudem beinhaltet dieser Ordner eine kompilierte Version des Prototyps.

Der Ordner *Referenzen* enthält alle Berichte, Bachelorarbeiten, Projektarbeiten und Vorlesungsskripte, die bei der Erstellung der vorliegenden Arbeit dienten. Zudem beinhaltet dieses Verzeichnis eine Offline-Version aller in dem Dokument als Referenz genutzten Webseiten.

Eidesstattliche Erklärung

Ich versichere, dass ich meine Masterarbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Rostock, den 27.06.2013

Nachname: Reichel Vorname: Tino

Matrikelnummer: 211209575 Unterschrift: _____