Prinzipien der quellennahen Datenverarbeitung exemplifiziert an einem Forschungsprojekt des Schweizerischen Nationalfonds

Autor(en): **Grassi, Giulio**

Objekttyp: **Article**

Zeitschrift: **Geschichte und Informatik = Histoire et informatique**

Band (Jahr): **11 (2000)**

PDF erstellt am: **27.05.2024**

Persistenter Link: https://doi.org/10.5169/seals-8926

Prinzipien der quellennahen Datenverarbeitung exemplifiziert an einem Forschungsprojekt des Schweizerischen Nationalfonds[1]

Giulio Grassi

Als im Sommer '96 mit der Zusage des Nationalfonds das Projekt «Soziale Beziehungen im Alltag einer spätmittelalterlichen Stadt, Zürich 15. Jahrhundert»[2] gestartet werden konnte und ich als Datenbankmanager mit der Aufgabe betraut worden war, eine Datenbank für die von uns in Betracht gezogenen Quellen – die *Rats- und Richtbücher*, die *Steuerbücher* und die *Eingewinnerverzeichnisse* – aufzubauen, stellte sich sogleich die Frage, welches *DBMS* (Datenbank Management System) für unsere Bedürfnisse am geeignetsten sei. Worauf sollte ich bei der Wahl des Datenbankprogrammes achten, welches waren unsere Abfragebedürfnisse, welche Systemvoraussetzungen hatten wir am Historischen Seminar in Zürich? Auf dem Datenbank-Markt findet man eine ganze Palette von Programmen, einfach zu bedienende «Karteikästchen»-Programme[3] wie z.B. FileMaker 2.x, AskSam, *relationale Datenbanksysteme* wie z.B. das weitverbreitete Access, das professionellere FoxPro oder das neuere FileMaker etc.; alles Applikationen, die für Datenbanken für den Hausgebrauch etc. ohne allzu grosse Sicherheits- und Performance-Anforderungen implementiert wurden. Daneben stiess ich im Internet auf das DBMS Kleio, das von Prof. Dr. Manfred Thaller[4] entwickelt und vom Max-Planck-Institut für Geschichte

1 Betreut wurde das Projekt von Prof. Dr. Hans-Jörg Gilomen, Ordentlicher Professor am Historischen Seminar der Universität Zürich. Die Teammitarbeiter möchten sich bei dieser Gelegenheit ganz herzlich bei Professor Gilomen bedanken. Zu diesem Projekt siehe auch den Artikel Bosshard/Malamud/Sutter in diesem Heft.

2 Die Untersuchungen im Rahmen des Gesamtprojekts sollen eine umfassende Sicht alltäglicher sozialer Beziehungen geben und deren Bedeutung für das Funktionieren der Gesellschaft, der Politik und der Wirtschaft in einer spätmittelalterlichen Stadt darstellen. Für genauere Angaben zum Projektziel vgl. die zugehörige Internet-Seite: *http://www.hist.unizh.ch/gilomen/NFProjekt/NF-Projekt.html*.

3 Unter Karteikästchen-Programmen verstehe ich ein rudimentäres Datenbankprogramm, das nur eine Entität/Tabelle/einen Karteikartentyp verwalten kann. Es können damit keine Beziehungen zwischen den Entitäten verwaltet werden.

4 Das DBMS Kleio ist aus einem langjährigen Forschungsprojekt des Max-Planck-Institutes für Geschichte in Göttingen hervorgegangen, dessen letzte Kapitel noch nicht geschrieben sind. Manfred Thaller war in den langen Jahren des Forschungsprojektes «Historical Workstation Project» des Max-Planck-Instituts der massgebliche, wenn nicht einzige Entwickler von Kleio. Die in Kleio implementierten Konzepte der quellennahen Datenverarbeitung sowie der in der Programmiersprache C abgefasste, sehr umfangreiche Source-Code stammt praktisch vollständig von Thaller

in Göttingen angeboten wurde[5], ein nicht kommerzielles Produkt, das aus einer langjährigen Forschungstätigkeit im Bereich der Historischen Fachinformatik hervorgegangen ist.

Nach einer längeren Evaluationsphase hat sich das Projektteam für Kleio entschieden, weil das DBMS für die Bedürfnisse von HistorikerInnen und nach den Prinzipien der quellennahen Datenverarbeitung implementiert worden war. Auch wenn die Arbeit mit dem Betriebssystem Unix[6], für das Kleio ursprünglich konzipiert wurde, auf den ersten Blick abschreckend gewirkt hat, so werden die folgenden Ausführungen hoffentlich zur Einsicht beitragen, dass es sich lohnt, ein DBMS, das die Prinzipien der quellennahen Datenverarbeitung abbildet, kennenzulernen.[7] Denn die Komplexität ergibt sich aus dem Bemühen, historische Quellen strukturell abbildbar (Modell) und intelligent durchforschbar (Retrieval) zu machen. Wer der Komplexität seiner Quellen gerecht werden will, *muss* ein System zur Hand nehmen, mit dem diese Komplexität auch abgebildet werden kann. Dies bedeutet allerdings nicht, dass die Benutzerfreundlichkeit und der Support[8] hintenan stehen müssen, eine Kritik, die sich Kleio trotz vielfältiger Vorzüge im strukturell-funktionalen Bereich gefallen lassen muss. Wenn also in den folgenden Ausführungen die Prinzipien der quellennahen Datenverarbeitung am Beispiel des Kleio-DBMS vorgestellt werden, dann deshalb, weil diese Prinzipien in Kleio effektiv umgesetzt worden

---

und wird auch heute noch ausschliesslich von ihm unterhalten und weiterentwickelt. Er ist heute Professor an der Universität Köln.

5 Die Göttinger Kleio-Homepage ist nach wie vor aktiv: *http://gwdu19.gwdg.de/kleio/manual/*. Eine neue Homepage ist, soweit dem Autor bekannt, in Köln vorgesehen.

6 Wir benutzen AIX auf dem RS 6000-Cluster des Rechenzentrums der Universität Zürich. Kleio ist in der Zwischenzeit auch als Windows-Programm erhältlich, wobei der X-Windows-Emulator separat beschafft werden muss. Macintosh hingegen wird nicht unterstützt.

7 HistorikerInnen ohne spezielle Computerkenntnisse empfinden die kommandoorientierte Arbeit in den Shells als unüberwindbare Hürde. Das Revival Unix-verwandter Systeme, v.a. Linux, zeigt m.E. hingegen, dass immer mehr Benutzer den Aufwand nicht mehr scheuen. Erweiterte Möglichkeiten, Stabilität u.a.m. kompensieren den Mehraufwand auf längere Sicht bei weitem.

8 Seit einigen Jahren ist von Thaller ein HTML-basiertes Benutzerhandbuch in Aussicht gestellt worden, es liegt allerdings immer noch nicht vor. Die Benutzer sind nach wie vor gezwungen, das völlig veraltete und für den Normalanwender viel zu technische Handbuch aus dem Jahre 1993 zu verwenden: Thaller, Manfred: «Kleio. A Database System». In: Halbgraue Reihe zur Historischen Fachinformatik B 11, Serie B: Softwarebeschreibungen. St. Katharinen 1993. In der selben Reihe erschienen und für den Anfänger sehr nützlich: Woollard, Matthew; Denley, Peter: «Source-Oriented Data Processing für Historians: a Tutorial for Kleio». In: Halbgraue Reihe zur Historischen Fachinformatik A 23. St. Katharinen 1993.

sind und weil unser Team seit nunmehr vier Jahren erfolgreich damit arbeitet. Auf das schwierige Unterfangen, die Implementierung von Kleio selbst zu kritisieren oder Kleio generell mit anderen DBMS zu vergleichen, möchte ich mich im Folgenden nicht einlassen – zu vielfältig sind die zu berücksichtigenden Faktoren. Die HistorikerInnen, die sich, wie ich vor vier Jahren, für ein Datenbankprogramm bezüglich eines bevorstehenden Projekts entscheiden müssen, werden ihre Voraussetzungen minuziös überprüfen und die vielfältigen spezifischen Bedürfnisse in Bezug auf ihre Quellen einzeln gewichten müssen. Ein in jeder Hinsicht zu empfehlendes DBMS für HistorikerInnen gibt es nicht!

1. Die in die Datenbank eingeflossenen Quellen

Im Rahmen des Nationalfondsprojektes sind von uns drei spätmittelalterliche Zürcher Gerichts- und Wirtschaftsquellen aus dem Zeitraum von 1450 bis 1470 als Kleio-Datenbanken abgebildet worden: die *Rats- und Richtbücher*, die *Steuerbücher* und die *Eingewinnerverzeichnisse*. Alle drei Quellenkorpora werden im Staatsarchiv Zürich aufbewahrt.

Die von 1376 bis 1798 beinahe lückenlos überlieferten Rats- und Richtbücher enthalten unterschiedliche Quellenkategorien zur Zürcher Gerichtsbarkeit: Geständnisse der Täter vor dem Blutgericht, an das Ratsgericht eingereichte Klagen von Privatpersonen, vom Rat angestrengte Nachgänge, Gnadengesuche, Kundschaften, Urfehden, originale Inhaltsverzeichnisse der halbjährlichen Richtperiode und Fragmente verschiedensten Inhalts. Die protokollierten Fälle des Ratsgerichts beinhalten zum Teil detaillierte Informationen zu Auseinandersetzungen, zu deren Verlauf, Kontext und Hintergrund sowie über verschiedene Konfliktformen und deren informelle und institutionalisierte Lösungen. Sie liefern aber auch wertvolle Hinweise zum Alltagsleben und zur Mentalität der Bewohner im spätmittelalterlichen Zürich.

Als wichtigstes Instrument zur sozialtopographischen Verortung und ökonomischen Schichtung der Einwohnerschaft Zürichs dienen die gedruckten Steuerbücher des 15. Jahrhunderts.[9] Weder für jedes Steuerjahr noch für alle sechs Verwaltungseinheiten (Wachten) liegen die Steuerverzeichnisse vollständig vor. Adrian Corrodi-Sulzer rekonstruierte anhand

9 Staatsarchiv des Kantons Zürich (Hg.), Die Steuerbücher von Stadt und Landschaft Zürichs des XVI. und XV. Jahrhunderts, bearb. v. Hans Nabholz, Friedrich Hegi, Edwin Hauser und Werner Schnyder, 8 Bde., Zürich 1918-1958.

des Murerplans[10] einen zu den Listen korrespondierenden Häuserplan, der die kartographische Grundlage zur Bestimmung der Wohnorte bildet.

Bei den Eingewinnerverzeichnissen handelt es sich um Betreibungs- oder Zwangsvollstreckungslisten, die in den Rats- und Richtbüchern von 1375 bis 1487 beinahe lückenlos überliefert sind. Eine Liste umfasst die halbjährliche Regierungszeit einer Ratsrotte und teilt diese in mehrere Aufnahmeperioden. Die umfangreichen Verzeichnisse enthalten hauptsächlich Schuldverhältnisse und nennen den Schuldner, den Gläubiger und den Schuldbetrag. Bei vielen Personen finden sich Berufsbezeichnungen, Wohnortsangaben und Informationen über verwandtschaftliche oder andere Beziehungen. Neben Geldschulden (v. a. Konsumationsschulden) sind auch Getreide-, Wein- und andere Lebensmittelschulden sowie Sachschulden (oder eine Kombination derselben) verzeichnet.

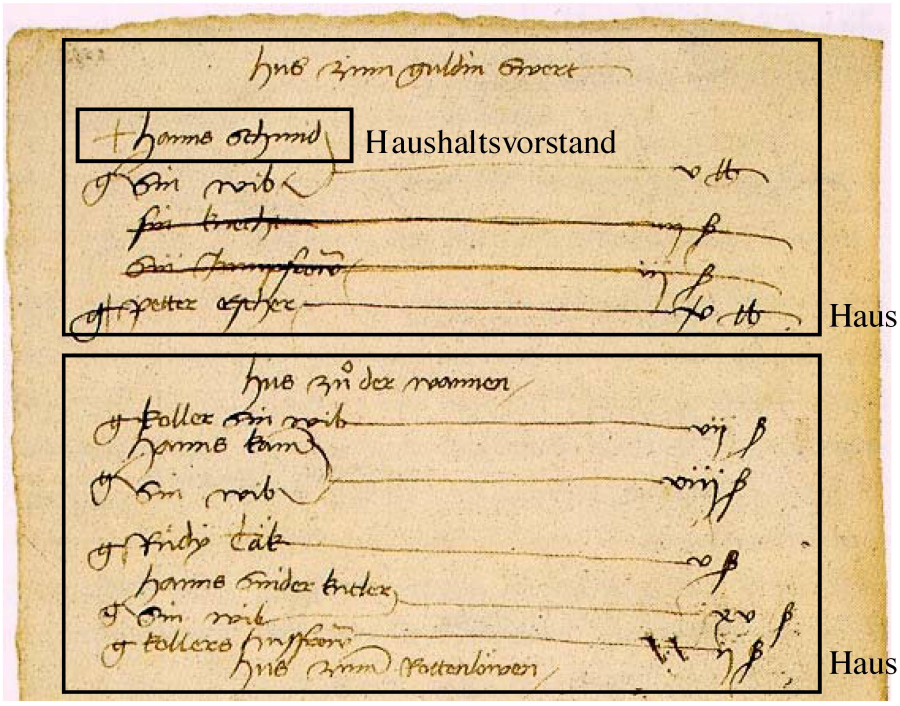


Abbildung 1: Auszug aus dem Steuerbuch-Original, Hervorhebungen durch den Autor

10 Beim Plan der Stadt Zürich von Jos Murer (1576) handelt es sich um eine recht zuverlässige Darstellung der baulichen Verhältnisse der spätmittelalterlichen Stadt Zürich.

2. Strukturmodellierung der Datenbank

Im Folgenden möchte ich nun zur Beschreibung der Prinzipien quellennaher Datenverarbeitung übergehen. Die beschriebenen Konzepte sind bei der Abbildung der oben vorgestellten Quellen allesamt mit den Werkzeugen von Kleio umgesetzt worden.

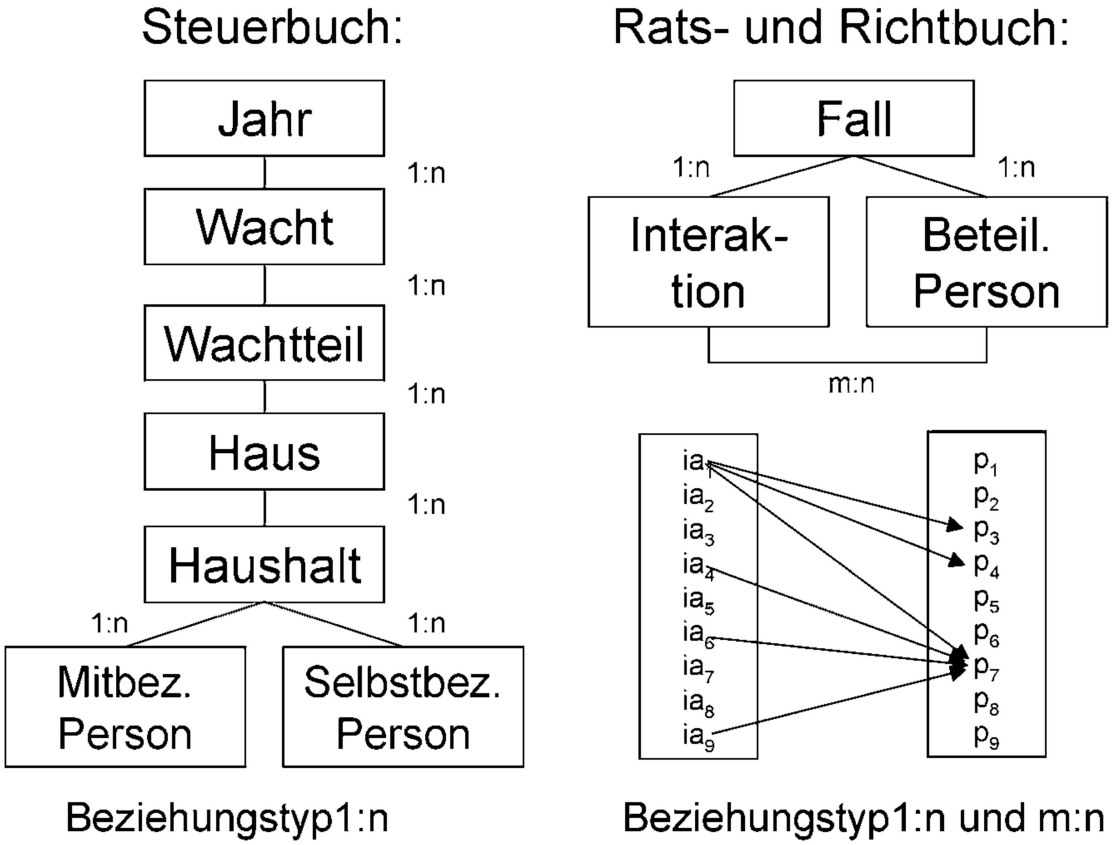
Am Anfang einer jeden Erfassung historischer Quellen in einem DBMS steht die Erstellung der Datenbankstrukturen, d.h. logischer Entitäten oder Gruppen und deren Beziehungen untereinander. Die in der zu erstellenden Datenbank enthaltenen Informationseinheiten mit ihren Beziehungen – die sogenannte Mini-Welt – sollten nun aber bei der historischen Forschungsarbeit nicht bei der Datenaufnahme im Hinblick auf spezielle Abfragebedürfnisse eingeschränkt werden, da nicht absehbar ist, welche zukünftigen Abfragen an die Datenbank gerichtet werden sollen. Während bei einer Betriebsverwaltungsdatenbank (Lagerverwaltung, Abrechungssystem etc.) die Datenbankstrukturen (Entitäten und Relationen der Mini-Welt) sämtliche innerbetrieblichen Informationsbedürfnisse abbilden sollten, liegen die Quellen, Überbleibsel vergangener Zeiten, Dokumente mit einer eigenen inneren Dokumentenstruktur vor. Die sogenannte logische Dokumentenstruktur muss anhand einer vertieften Quellenanalyse aufgefunden werden. Die besten KennerInnen der Quellen müssen zusammen mit dem Datenbankmanager, der sich in den Möglichkeiten des ausgewählten DBMS auskennt, einen Weg finden, strukturelle Regelmässigkeiten der einzelnen Quellen zu definieren. Das Resultat dieses Prozesses ist das Auffinden logischer Gruppen (Tabellen im relationalen Konzept), deren Beziehungen untereinander und der Informationselemente (Felder) in den Gruppen.[11]

Hierarchische Beziehungstypen (1:n) werden dabei von *Netzwerkbeziehungen* (m:n) unterschieden (vgl. Graphik 1).[12] Alle drei in unserem Projekt in Betracht gezogenen Quellen besitzen eine explizite *logische Dokumentenstruktur* die rein hierarchisch ist, d.h. die in ihnen enthaltenen Informationselemente können zu Gruppen zusammengefasst werden, die jeweils in einer 1:n-Beziehung stehen. Geht man von der obersten (oder untersten, je nach Betrachtung) Gruppe aus (root = Wurzel) und bewegt

11 In dem von mir verfassten, unveröffentlichten Manual sind sämtliche Datenbankstrukturen, Kodierungslisten und Konventionen der Datenaufnahme beschrieben: Grassi, Giulio: Kleio-Manual. Soziale Beziehungen im Alltag einer spätmittelalterlichen Stadt. Zürich 1450-70: Rats- und Richtbücher, Steuerbücher, Eingewinnerverzeichnisse. Zürich 1999.

12 Die hier verwendete Terminologie ist an die Struktur von Kleio angelehnt und weicht damit von der klassischen Datenbankterminologie ab.

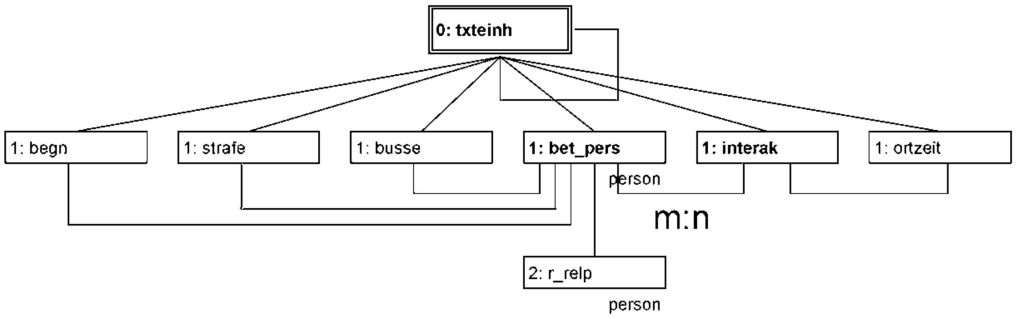
man sich von Gruppe zu Gruppe zu den äussersten Verästelungen des Baumes, so besteht immer eine *eins zu mehrfach-Beziehung* zwischen den jeweiligen Gruppen: Ein bestimmtes *Jahr* der Steuerbücher besteht aus mehreren *Wachten*; jede Wacht hat mehrere *Wachtteile*; in einem Wachtteil hat es mehrere *Häuser*; in einem Haus gibt es mehrere *Haushalte* (Hauptsteuerpflichtige); jeder *Hauptsteuerpflichtige* hat mehrere *Personen, für die er mitbezahlt* und gleichzeitig *mehrere Personen in seinem Haushalt, die ihre Steuern selbst bezahlen*. In umgekehrter Richtung gehört ein Haushalt immer genau zu einem Haus.



Graphik 1: Hierarchische und netzwerkartige Beziehungstypen am Beispiel der Steuerbücher und der Rats- und Richtbücher

Während die einzelnen Interaktionen in den Texteinheiten (Gerichtsverfahren) der Rats- und Richtbücher ebenfalls in einer hierarchischen Beziehung zu den Texteinheiten stehen, kommt man bei der Kategorisierung der Gerichtsverfahren in den Rats- und Richtbüchern in *Interaktionen* (z.B. Delikte) und den daran *beteiligten Personen* mit einer hierarchischen Struktur nicht mehr aus (vgl. auch Graphik 2). Das Beispiel in Graphik 1 zeigt ein Verfahren, das aus mehreren Delikten (Interaktionen) besteht, bei

denen mehrere Personen beteiligt sind. In der ersten Interaktion (ia_1) sind die Personen p_3, p_4 und p_7 beteiligt; p_7 ist aber neben der Interaktion 1 auch in den Interaktionen 4, 6 und 9 beteiligt.



Graphik 2: Das Strukturmodell der Rats- und Richtbuch-Gerichtsverfahren

Obwohl nun die meisten historischen Quellen eine hierarchische logische Dokumentenstruktur aufweisen, muss ein jedes einigermassen brauchbare DBMS beide Beziehungstypen vorsehen. Datenbankprogramme, die nach dem Karteikästchen-Prinzip aufgebaut sind, d.h. nur eine Entität/Gruppe von Informationselementen zulässt und damit weder die Definition von hierarchischen noch von Netzwerk-Beziehungen zwischen verschiedenen Entitäten zulassen, können im Allgemeinen den Anforderungen einer sauberen Abbildung historischen Quellenmaterials nicht genügen.
Ein weiteres Problem besteht in der Unregelmässigkeit der Quellenstruktur. Bei der Arbeit mit historischem Quellenmaterial hat man es fast nie mit konsistenten Datensätzen zu tun. Hätte man auf der anderen Seite keine genügend regelmässige Struktur in einer Quelle, könnte man bei einer Datenmodellierung keine Strukturen mehr definieren. Die Wahrheit bei der Arbeit mit Quellen liegt meistens in der Mitte: Man wird historischem Quellenmaterial in den meisten Fällen nur dadurch gerecht, dass man auf der einen Seite dieses Quellenmaterial strukturiert, soweit Regelmässigkeiten ersichtlich sind, *Lückenhaftigkeit* jedoch von vornherein vorsieht und *datensatzspezifische Mehrinformation* zulässt. Es muss also die Möglichkeit bestehen, ein Entitäten/Gruppen- und Feldergerüst für die zu erwartenden (fast immer vorkommenden) Entitäten und Felder zu definieren, ohne dass dabei eine Gruppe oder ein Feld tatsächlich in den verschiedenen Datensätzen vorkommen *muss*. Die Existenz der Gruppen und Felder ist somit *datensatzsensitiv*. Gleichzeitig soll es auch möglich sein, Felder in den Datensätzen einzubringen, die in der Strukturdefinition nicht vorge-

sehen sind. Spezielle Informationseinheiten werden somit nur für bestimmte Datensätze definiert.

Die Vorteile der Datensatzsensitivität werden evident, sobald man einerseits damit beginnt, Datensätze in grosser Menge einzugeben, und andererseits bei der Report-Erstellung. Bei der Dateneingabe[13] genügt es, denjenigen Feldbezeichner zu benutzen, für die es tatsächlich im entsprechenden Datensatz auch einen Eintrag gibt. Besitzt ein Datensatz eine Mehrinformation, die nur dort vorkommt, wird für diesen spezifischen Datensatz ein Feld definiert, in dem dann die Mehrinformation aufgenommen werden kann, ohne dass die Gesamtstruktur davon betroffen wäre. Auf der anderen Seite erscheint bei der Generierung von Reports[14] niemals ein Feld, das für den entsprechenden Datensatz nicht existiert. Dies hat den Vorteil, dass Listen mit vielen Feldbezeichnern und wenig Inhalten vermieden werden.

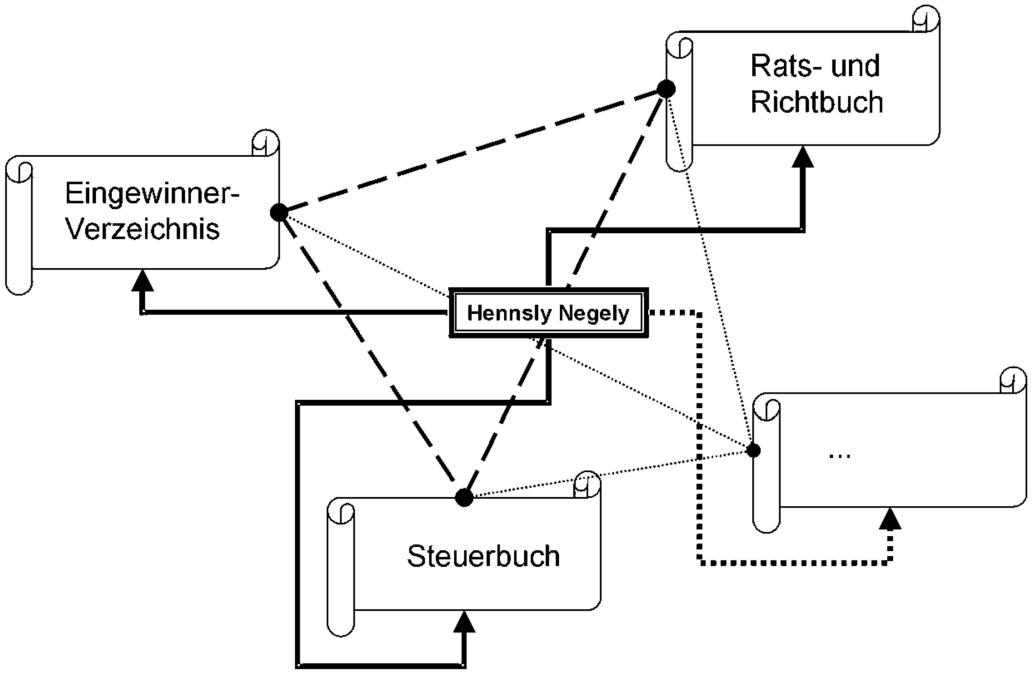
Kommerzielle relationale Datenbanken besitzen im Normalfall keine derartigen flexiblen Struktur-Handhabungsmöglichkeiten. Die Strukturdefinition sowie die Generierung von Sichten bzw. Reports sind im Wesentlichen statisch, obwohl Funktionen für die nachträgliche Unterdrückung leerer Felder meist vorhanden sind. Eine Flexibilisierung der Strukturen kann zwar auf der Ebene der Programmierung (Skript-Sprachen, Visual Basic u.a.m.) über den statischen Strukturen erreicht werden, die Datensatzsensitivität sollte jedoch auch aus Performance-Gründen auf der Ebene der Datenspeicherung implementiert sein.

Der prosopographische Ansatz bei der Arbeit mit verschiedenen Quellen, die Informationen zu identifizierbaren Personen enthalten, bildet ein Musterbeispiel für den Nutzen eines modularen Datenbankaufbaus mit nachträglicher fixer Verbindung durch Netzwerk-Brücken. Die drei von uns aufgenommenen Quellen beinhalten auf verschiedensten Ebenen Personeninformationen, die zunächst in keiner Verbindung zueinander stehen, die es aber für prosopographische Fragestellungen zusammenzuführen gilt (vgl. Graphik 3). Der in der Historischen Fachinformatik unter der Be-

13 In Kleio werden die Daten in einem beliebigen Text-Editor eingegeben, wobei die Strukturierung über Spezialzeichen verwirklicht wird. Der Parser vergleicht die Datenstruktur des eingelesenen Datenfiles mit der explizit definierten Datenbankstruktur und speichert intern die effektive Struktur der Datenbank. Der Verzicht auf eine Dateneingabe auf graphischer Ebene hat jedoch den Nachteil, dass das Eingegebene nicht während der Eingabe kontrolliert werden kann, sondern erst beim Einlese-Prozess.

14 In Kleio gibt es keinen graphischen Report-Editor, in dem die anzuzeigenden Felder angegeben werden. Der Output einer Abfrage erscheint als Text-File, in dem für einen Datensatz nicht existente Felder ausgelassen werden (default).

zeichnung *Nominal Record Linkage* bekannte Vorgang hat zum Ziel, durch das Setzen von festen Verbindungszeigern die in den verschiedenen Quellen enthaltenen Informationselemente zusammenzuführen. Der Kleio-spezifische Vorgang soll hier nicht weiter beschrieben werden, hingegen kann noch darauf hingewiesen werden, dass es sich hier in der Regel um ein schwieriges Unterfangen handelt, da oft nicht genügend eindeutige Identifikatoren vorhanden sind. Ein wohlüberlegter interaktiver Prozess muss durchlaufen werden, um die potentiell identischen Personen in den verschiedenen Quellen aufzufinden. Durch den Einsatz von speziellen *Namensausgleichsverfahren* sowie *Kodierungslisten* und weiteren Filterbedingungen können jedoch die potentiell identischen Personen automatisch aufgefunden werden. In einem weiteren Schritt wird dann auf der Ebene der Personen-Informationsinhalte mit der Erfahrung des Quellenkundigen darüber befunden, ob die Identitäten tatsächlich übereinstimmen.



Graphik 3: Personenidentifikation – Verbindung der Quellen mit Netzwerk-Brücken[15]

15 Die gestrichelte Linie in der Grafik verdeutlicht die Verbindung der verschiedenen Quellen (Datenbanken in Kleio) miteinander. Die Pfeile von Hennsly Negely zeigen an, dass diese identifizierte Person in den einzelnen Quellen vorkommt. Zwischen den Datenbanken besteht somit ein Personennetzwerk. Jede Person bekommt eine ID, mit deren Hilfe Kleio das Netzwerk aufbaut. Die feinen Linien zum leeren Quellendokument deuten an, dass die Datenbank beliebig erweiterbar ist.

3. Sekundäre, benutzerspezifische Kategorisierung über den Feldinhalten

Ein weiteres wichtiges Konzept innerhalb der quellennahen Datenverarbeitung ist die Trennung zwischen der Datenbankstrukturierung bei der Datenaufnahme und der anschliessenden Interpretationsebene. Die Regeln, die es für die Bildung der Datenbankstrukturen zu beachten gilt, sind bereits skizziert worden. Im Wesentlichen geht es darum, die logische Dokumentenstruktur zu erfassen und genau so, wie sie vorliegt, abzubilden. Die nachfolgende Interpretationsebene mit den dazugehörigen Abfragen wird in diesem Stadium überhaupt nicht beachtet. Tatsächlich ist die Interpretationsebene vollständig von der Datenbankstrukturierung entkoppelt.

Die Vorteile dieser Vorgehensweise können anhand eines Beispiels in einer unserer Datenbanken illustriert werden, der Eingewinnerverzeichnisse. Das zugrunde liegende Konzept taucht in der historischen Fachinformatik unter der Bezeichnung *Postkodierung* auf.

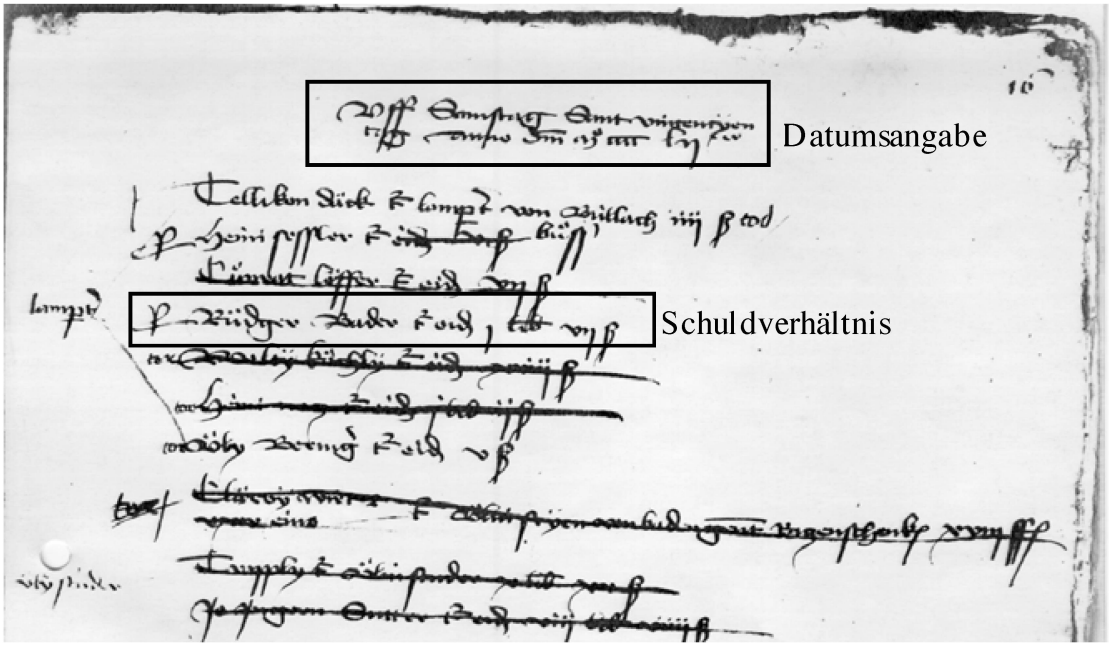


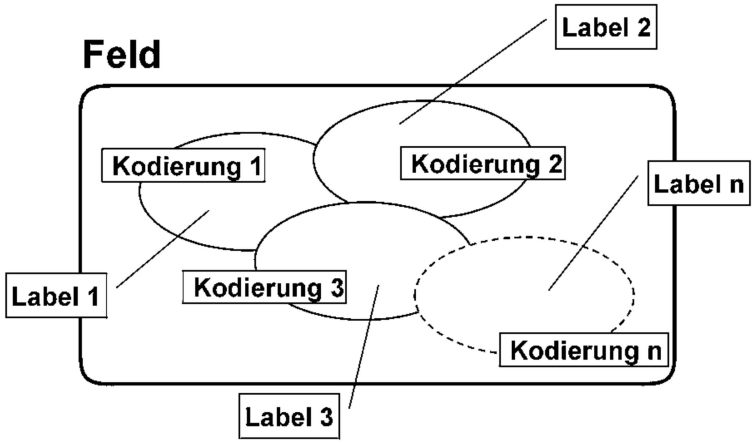
Abbildung 2: Auszug aus dem Eingewinnerverzeichnis-Original, Hervorhebungen durch den Autor

Bei den Eingewinnerverzeichnissen handelt es sich um eine serielle Quelle, die man mit einem modernen Begriff als Betreibungsamtslisten bezeichnen kann. Im Wesentlichen besteht die Quelle aus einer Auflistung von Schuldverhältnissen, in denen Schuldner, Gläubiger und der geschuldete Betrag aufgeführt werden. Ab und zu erscheint jedoch nicht ein geschuldeter

Geldbetrag, sondern eine materielle Schuld, wobei es sich dabei zum Teil um irgendwelche Gegenstände handelt. Nach den Regeln der quellennahen Datenverarbeitung wird man in der Datenbank ein Feld vorsehen, in das die geschuldeten Gegenstände der jeweiligen Schuldverhältnisse eingetragen werden, ohne einen Gedanken daran zu verlieren, wie diese Gegenstände gruppiert werden können. Die Gruppierung der Gegenstände ist aber für die nachträgliche Interpretation von zentraler Bedeutung, und man wird nicht darum herumkommen, diese auch vorzunehmen. Die Gruppierung der Eintragungen hängt nun aber von den Absichten der/s Interpretierenden ab und kann damit nicht allgemein festgelegt werden.

Die Lösung dieses fundamentalen Problems im Rahmen der Arbeit mit historischen Quellen kann dadurch erreicht werden, dass diese Gruppierungen eben nicht – wie gesagt – von den Erstellern der Datenbank vorgenommen werden, sondern von den Interpretierenden, durch das Erstellen von Kodierungslisten («Codebooks», in der Terminologie von Kleio). In unserem Beispiel werden sämtliche Einträge des Feldes, das die geschuldeten Gegenstände beinhaltet, herausgegeben und mit einer Kodenummer versehen. Durch eine geschickte Verwendung der einzelnen Ziffern des Kodes ist es nun möglich, eine für den Interpretierenden sinnvolle Kategorisierung der Gegenstände vorzunehmen, z.B. eine Gruppierung der Gegenstände in Werkzeuge, Textilien, Arbeits- und Rohmaterial, Nahrungsmittel, Haushaltsgegenstände, Schmuck und Brennmaterial. In den Abfragen wird man nun anhand dieser Kodeziffern die Gegenstände der einzelnen Gruppen ansprechen können. Hinzu kommt, dass durch die Übergabe dieser Kodes an professionelle Statistikprogramme (z.B. SPSS) komplexe Auswertungen vorgenommen werden können.

Das Konzept der Postkodierung eignet sich darüber hinaus gut für den Ausgleich von Schreibungsvarianten aller Art, die nur auf semantischer Ebene zusammengeführt werden können. Verschieden geschriebene Ausdrücke, die buchstabengetreu aus der Quelle übertragen wurden, können damit nachträglich vereinheitlicht und mit einem benutzerdefinierten Bezeichner versehen werden. Diesen Vereinheitlichungsvorgang übernimmt wiederum der Interpretierende, da es in seinem Ermessen liegt, welche Ausdrücke für seine Auswertungen tatsächlich dasselbe bedeuten sollen. Die untenstehende schematische Darstellung des Postkodierungsvorgangs mit anschliessender Zuordnung von Labels zeigt das dahinterstehende Konzept (vgl. Graphik 4). Über jedes Feld können beliebig viele Kodierungslisten mit beliebig vielen Kodierungsvariablen (Kodierungen innerhalb einer Liste) und den dazugehörigen Labels definiert werden.



Graphik 4: Postkodierung – Feldinhalt und Kodierung trennen

4. Flexibles Retrieval mit ausgefeilten Suchfunktionen

Die Arbeit mit Personendaten, die mit den Regeln der quellennahen Datenverarbeitung aufgenommen worden sind, d.h. buchstabengetreu transkribiert in die Datenbank eingeflossen sind, wäre ohne Hilfe von *Namensausgleichsverfahren* nicht denkbar. Die normalerweise von den kommerziellen Anbietern mitgelieferten *Matching-Funktionen*[16] reichen aber in der Regel für die Arbeit mit historischem Datenmaterial nicht aus. Schreibungsvarianten entstehen durch phonetische und graphematische Variabilität und müssen folglich auch auf dieser Ebene wieder ausgeglichen werden. Verschiedene, mit dem Konsonantenskelett von Eigennamen arbeitende Ausgleichsalgorithmen müssen folglich angewendet werden können. Der bekannteste Algorithmus ist *Soundex*[17]. Die Wahl des geeignetsten Algorithmus für eine vorliegende Quelle hängt von der darin enthaltenen Wortstruktur des «Dialekts»[18] ab. Die Variabilität der Schreibungen muss in einem komplexen Prozess ausgeglichen werden, ohne dabei die bedeutungstragenden Elemente der Wörter zu eliminieren. Der Algorithmus zusammen mit den experimentell aufgefundenen Ausgleichsgruppen erzeugen Matching-Funktionen, die weit über die Leistungsfähigkeit von Wildcards hinausgehen.

16 Wildcards, reguläre Ausdrücke etc. in Kombination mit den logischen Operatoren UND, ODER und NICHT.

17 Neben Soundex enthält Kleio auch zwei weitere Algorithmen mit den Bezeichnungen «Skeleton» und «Guth».

18 In unserem Fall handelt es sich um den zu Zürich und Umgebung gehörenden mittelhochdeutschen Dialekt aus dem 15. Jahrhundert mit den Eigenheiten des jeweiligen Schreibers.

Die Gleichsetzungsgruppen müssen für jede «Sprache» speziell definiert werden können. Es nützt nichts, einen vorgefertigten Algorithmus zur Verfügung zu haben, der für den Ausgleich des modernen Deutsch konzipiert wurde, wenn man es mit der regional gefärbten Sprache des 15. Jahrhunderts zu tun hat. Mit benutzerdefinierbaren Ausgleichsalgorithmen bekommen die Datenbankabfragenden ein mächtiges Instrument in die Hand, um auch ohne genaue Kenntnisse der Schreibungsvarianten innerhalb einer Quelle gesuchte Zeichenketten zu finden. Konsonantengerüstbasierte Ausgleichsalgorithmen in Kombination mit Kodierungslisten für den Ausgleich auf semantischer Ebene garantieren ein fast sicheres Auffinden der gesuchten Datenbankeinträge, und dies trotz Erhalt der ursprünglichen Schreibungen.

Input-Zeichenkette:	„*Tachelshofer*“	Separatoren:	
	2 4 75¦ 3 8	a;e;i;o;u;h;y;j	(=1)
Matching:	„*Tachelshower*“	Konsonantenausgleich:	
	2 4 75¦ 3 8		
	„*Tachelshofferin*“	d;t	Code=2
	2 4 75¦ 3 8	f;v;w;p;b	Code=3
	„*Tachellßover*“	c;k;g;q;x	Code=4
	2 4 75¦ 3 8	s;ß;z	Code=5
	„*Tachselhouwer*	m;n	Code=6
	2 4 **5 7**¦ 3 8	l	Code=7
	„*Dahelhofer*“	r	Code=8
	2 4 7 3¦8		

Graphik 5: Handling von Schreibungsvarianten in Soundex

Das Beispiel (vgl. Graphik 5) eines selbstdefinierten Soundex-Ausgleichsverfahrens zeigt die Leistungsfähigkeit und gleichzeitig die Grenzen des Verfahrens. Der Algorithmus besteht aus folgenden Regeln:

- Alle Zeichen, die im Algorithmus nicht definiert sind, werden ignoriert. (z.B. ein Apostroph)
- Jede Gruppe von Zeichen, die auf ein Zeichen folgt, das denselben Code bekommt wie die Gruppe von Zeichen, wird ignoriert. (Gleiche Konsonanten in Serie werden auf einen Konsonanten reduziert.)
- Eine Auswahl von Zeichen fungiert als Separatoren. (In der Regel die Vokale, verschieden je nach «Sprache».) Diese Zeichen werden ignoriert, ausser sie stehen am Anfang der Zeichenkette. In diesem Fall wird ihnen die Codeziffer 1 zugeordnet.

– In der Regel begnügt man sich mit einem vierstelligen Code, d.h. der Algorithmus endet nach der Definition der ersten vier Code-Ziffern (Zeichen «_» in Graphik 5).
– Die Zeichenketten können vorbearbeitet werden. (z.B. können Vorsilben wie «von» oder Endungen wie «erin» vor dem Eingang der Zeichenkette in den Algorithmus abgetrennt werden.)

Das Beispiel zeigt, wie einfache phonetische und graphematische Variabilität durch Gleichsetzungsgruppen ausgeglichen werden kann, jedoch kaum eine Rotation von zwei Konsonanten wie im Beispiel «Tachselhouwer» oder das Ausfallen wichtiger Konsonanten wie im Beispiel «Dahelhofer».

Bereits mehrfach ist darauf hingewiesen worden, dass Datenaufnahmekonventionen von den Abfragebedürfnissen nicht tangiert werden sollten. Bei der Transkription der Informationselemente werden üblicherweise Trankriptionsregeln verwendet, die eine möglichst grosse Quellennähe garantieren sollen. Die Abfragenden einer Datenbank, die die Texte nicht selbst aufgenommen haben, können aber den Variantenreichtum der Textelemente (z.B. Eigennamen) kaum abschätzen, weshalb die Suche nach bestimmten Personen nur durch phonetische Ausgleichsalgorithmen bewerkstelligt werden kann. Die von uns auf dem Internet zur Verfügung gestellte prosopographische Suche in den drei Datenbanken benutzt aus diesem Grunde die Soundex-Transformation für die Personen-Nachnamen und Kodierungslisten für die Vornamen, die nur auf semantischer Basis gleichgesetzt werden können.[19]

5. Spezialanforderungen an das Feld

Das Feld einer Datenbank ist das Gefäss, in dem die zu speichernden Informationseinheiten «gelagert» werden. Alle kommerziellen Datenbankprogramme stellen eine ganze Palette von Feldtypen zur Verfügung: *Textfelder*, *Zahlenfelder*, *Datumsfelder*, *Logische Felder* etc. Als HistorikerIn wird man aber sogleich mit der Tatsache konfrontiert, dass diese Programme den speziellen Bedürfnissen der Arbeit mit historischen Quellen nicht genügen. Es gibt keinen Nummern-Feldtyp, der mit einem Tripel wie *2 £ 7 s 8 d* oder einen Datumstyp der Art *Mittwoch nach Laetare 1476* umgehen könnte. Das Datum ist in Richtung Vergangenheit in einigen Fällen

19 Das Abfrageformular hat folgende URL: https://www.unizh.ch/hist/gilomen/NFProjekt/nfpgilo/ssldir/query_form.htm. Ein Beispiel einer Personenabfrage liefert der Beitrag von Bosshard/Malamud/Sutter in diesem Heft. Da die Seite geschützt ist, muss ein Passwort am Lehrstuhl Gilomen angefordert werden.

auf den 1.1.1900 beschränkt. Kommerzielle Datenbankprogramme bieten heute zwar stets integrierte Programmierwerkzeuge an, mit deren Hilfe spezielle Feldtypen generiert werden können. Oft können die Werteberei-che eines Feldes durch eine Prozedur oder ein Skript überprüft werden. Im Allgemeinen aber muss man auf mühevollen Umwegen die nötigen Feld-typen zusammenbasteln und riskiert dabei beträchtliche Geschwindigkeits-verluste, da diese Funktionen nur über die Grundstrukturen übergestülpt sind. Ein DBMS, das für die Arbeit mit historischem Quellenmaterial kon-zipiert worden ist, muss hingegen alle diese speziellen Mechanismen in der Grundkonzeption vorsehen. Einige dieser Spezialmechanismen, die das Feld betreffen, sollen im Folgenden kurz beschrieben werden.

Der Datums-Feldtyp ist ein gutes Beispiel für den Nutzen, den man aus DBMS herausziehen kann, die speziell für die Abbildung historischen Quellenmaterials konzipiert worden sind. Daten liegen z.B. in unseren Quellen fast ausschliesslich als Wochentage in Relation zu einem Festtag vor, z.B. *Mittwoch nach Dorothe*. Die Festtage können einerseits ein fixes Jahresdatum haben, andererseits aber auch in Relation zu Ostern liegen, wobei bekanntlich das Osterdatum auch wiederum variabel ist. Idealer-weise wird nun der Datums-Feldtyp die Möglichkeit bieten, einerseits den originalen Eintrag der Art *Mittwoch nach Dorothe* (fixer Festtag) oder *Samstag nach Laetare* (beweglicher Festtag) aufzunehmen und andererseits diese zu den Festtagen relativ positionierten Daten auf elegante Weise automatisch in ein Nummerndatum umzuformen, indem zuerst mit internen Algorithmen der Wochentag des Festtages und anschliessend der Tages-abstand zwischen dem aufzulösenden Datum und dem Festtag errechnet wird. Daneben sollte das System als Ganzes die Möglichkeit bieten, mit *Termini post* und *ante quem* zu arbeiten.

Ebenfalls sehr nützlich ist die Möglichkeit, ein Feld mit Mehrfachein-trägen zu versehen (vgl. Grafik 6).[20] Bei der Arbeit mit historischen Quel-len taucht das Phänomen immer wieder auf, dass man z.B. anstatt nur einer Berufsangabe zwei davon hat oder dass man nicht entscheiden kann, ob es sich bei einem Begriff um den Namen oder den Beruf einer Person handelt. Idealerweise – weil man sich für die späteren Abfragen nichts verbauen möchte – wird man in diesem Fall beide Begriffe im Feld *Name* als Mehr-facheintrag einfügen, am besten durch die Verwendung eines Fragezei-chen-Tags. Ausserdem kann mit der Möglichkeit von Mehrfacheinträgen in

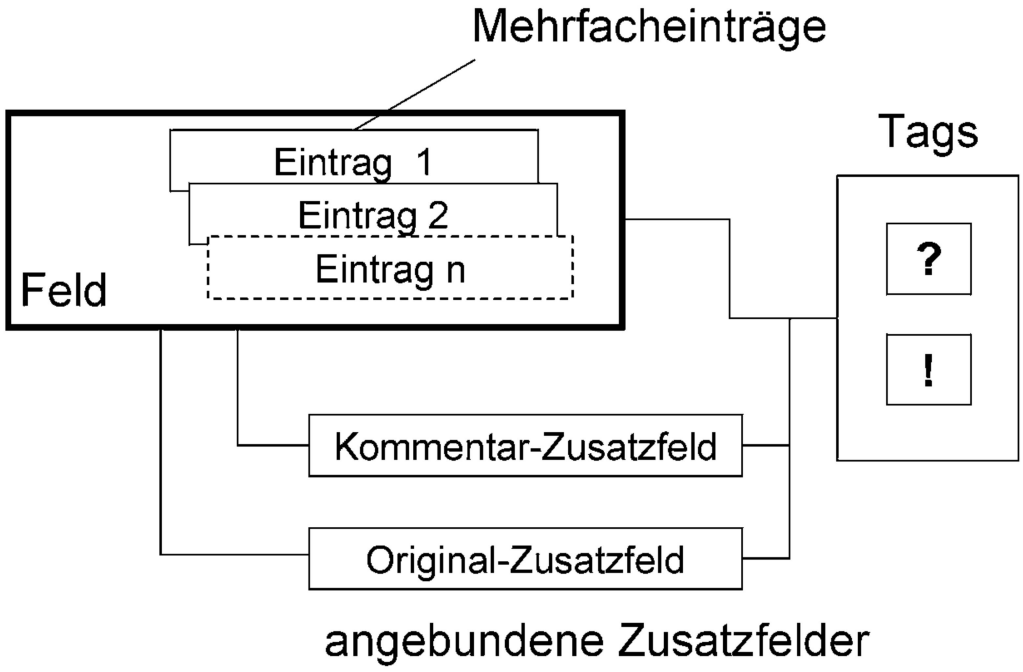
20 Im relationalen Modell werden Mehrfachfelder im Normalisierungsprozess in eine eigene Entität übersetzt; damit werden ebenfalls keine überflüssigen Felder mitgeführt, die Strukturverästelungen nehmen jedoch zu.

einem Feld elegant dem Problem begegnet werden, dass die Anzahl von Einträgen in einem Feld tatsächlich unregelmässig ist. Hat man z.B. für das Feld *Beruf* eine Verteilung von *einem* Eintrag in 80 % der Fälle, *zwei* Einträgen in 15 % der Fälle und *drei* Einträge in den restlichen 5 % der Fälle, so wird man trotzdem nur ein *Berufe*-Feld definieren und gleichzeitig nicht stets eine grosse Anzahl leerer Felder mitführen müssen. Die Möglichkeit, mehr als einen Eintrag in einem Feld zu verwalten, wird allerdings mit einer z.T. beträchtlichen Verkomplizierung der Abfragevorgänge erkauft.

Neben dem eigentlichen Feld eines bestimmten Typs sind *Original- und Kommentar-Zusatzfelder*, die an jedes normale Feld angehängt werden können, bei der Abbildung historischer Quellen äusserst nützlich. Quellennähe bedeutet auch, dass bei gewissen Feldern eine Originalschreibung mitgeführt werden kann. Kommentarfelder bieten die Möglichkeit, wichtige Zusatzinformationen, die in der Quelle nicht explizit enthalten sind oder auf interpretatorischer Ebene entstehen, anzubringen. Inhalte dieser Spezialfelder können mit speziellen Funktionen angesteuert werden.

Ein weiterer Spezialmechanismus ist die Möglichkeit, Feldeinträge sämtlicher Feldtypen mit sogenannten *Tags* zu versehen. Durch Anbringen eines dieser Feldattribute können Feldinhalte als unsicher (Fragezeichen) oder als bemerkenswert (Ausrufezeichen) markiert werden. Auch die Tags können mit Spezialfunktionen angesteuert werden.

Ein letztes, ebenfalls sehr wichtiges Merkmal ist die unbegrenzte Länge der Feldeinträge. Quellen können sowohl in Listenformat als auch in Volltextformat vorliegen, oder auch eine Kombination von beidem wie in unseren Rats- und Richtbüchern. Volltexte in einer Datenbank genau gleich zu verwalten wie Informationselemente aus Listen widerspricht aber der klassischen Datenbanktechnik und ist deshalb in den kommerziellen DBMS relationaler Art oft nicht vorgesehen. Bei den kommerziellen Systemen bedarf es immer einer Sonderanstrengung, die Volltexte in den Datensätzen mitzuführen. Die Abfragesprachen dieser Systeme (SQL, *Structured Query Language*) lassen diese speziell verwalteten Volltexteinträge teilweise beiseite. Jedes Feld muss aber geeigneterweise potentiell in der Länge unbegrenzt sein, da Volltexteinheiten bei der historischen Forschungsarbeit mitgeführt werden müssen. Für die Inhalte dieser Volltextfelder bedarf es sodann einer ganzen Palette von Spezialfunktionen, die es erlauben, ein intelligentes Volltext-Retrieval durchzuführen. Die Arbeit mit Volltexten muss im DBMS speziell berücksichtigt sein, gleichzeitig sollte aber der Text-Feldtyp in der Grundkonzeption des Datenbank-Programms voll integriert sein.



Graphik 6: Das Feld – Mehrfacheinträge, Zusatzfelder und Tags

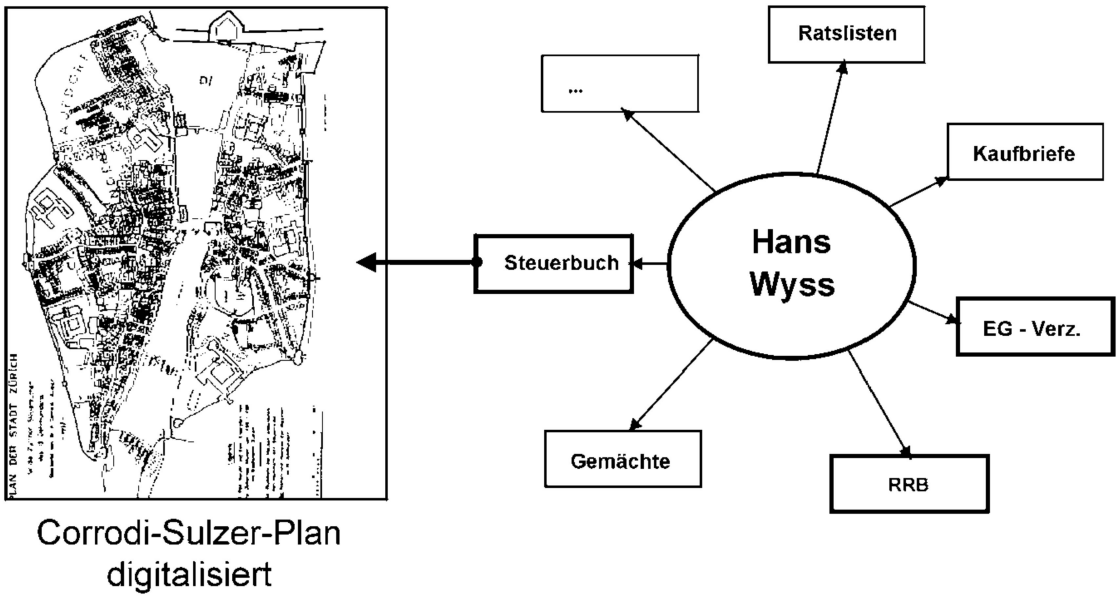
6. Visualisierung raumbezogener Daten (GIS)

Quellennähe kann auch bedeuten, dass man zur Verfügung stehende kartographische Quellen an die textuellen Quellen anbindet, um die Möglichkeit der topographischen Visualisierung auszunutzen. Die Verwendung von *geographischen Informationssystemen* (GIS) hat in den letzten Jahren, ausgehend von der Geographie, einen richtigen Boom erfahren. In den Geschichtswissenschaften macht der Einsatz von GIS überall dort Sinn, wo die topographische Dimension mit sozioökonomischen Fragestellungen korreliert wird. Wirtschafts- und SozialhistorikerInnen, deren Forschungsobjekt die Stadt Zürich im 15. und 16. Jahrhundert ist, sind dank des Murer-Plans und den Rekonstruktionen von Corrodi-Sulzer in der glücklichen Lage, auf eine bis auf weite Strecken präzise Rekonstruktion der städtischen Topographie zurückgreifen zu können. Die Verbindung von *Haus* und *Steuerbuchnummer* ist für die meisten Häuser in den Wachten bekannt, und da die Stadttopographie bereits rekonstruiert ist, liegt es nahe, topographische Objekte zu digitalisieren und mit den Einträgen im Steuerbuch zu verbinden (vgl. Graphik 7).

Kleio beinhaltet die Möglichkeit, raumbezogene Daten zu visualisieren, wobei das Handling der topographischen Objekte veraltet und damit eher schwerfällig ist. Es können jedoch beliebige Abfragen an die drei von uns

berücksichtigten Quellen gestellt und, soweit eine topographische Visualisierung überhaupt Sinn macht, angezeigt und im Post-Script-Format ausgedruckt werden.

Möchte man GIS auf dem neuesten Stand zur Verfügung haben, muss man sich die entsprechenden Programmpakete beschaffen. GIS ist ein Entwicklungsgebiet für sich, wobei eine gewaltige Diskrepanz zwischen den wirklich professionellen Anwendungen und Pseudo-GIS-Programmen mit beschränkten Möglichkeiten besteht. Die Rechenleistung der verfügbaren Maschinen ist hier ein wichtiges Kriterium. Da es sich beim Murer-Plan um eine aus professioneller Sicht bescheidene digitale Umsetzung einer Karte handelt, kann durchaus eine einfache GIS-Anwendung den Anforderungen genügen. Wichtig ist eine einfache Handhabung der topographischen Objekte, die Möglichkeit der Arbeit mit verschiedenen Ebenen und eine bequeme Anbindung an externe Daten, d.h. Abfrageresultate-Files der verwendeten Datenbank.[21]



Graphik 7: Personen in diversen Quellen – topographische Visualisierung

Die folgende thematische Karte (vgl. Abbildung 3) zeigt anhand der Zuordnung verschiedener Flächenmuster und Grautöne die Anzahl Hausbewohner pro Haus innerhalb des Corrodi-Sulzer-Planes für das Jahr 1467. Eine Kleio-Abfrage liefert die Daten in Form eines ASCII-Files, das dann

21 Ein solches Programm ist MapViewer von Golden Software. Der Corrodi-Sulzer-Plan liegt als Vektorgraphik im Format dieses Programms vor.

zu einem Excel-File konvertiert und schliesslich in MapViewer importiert wird.

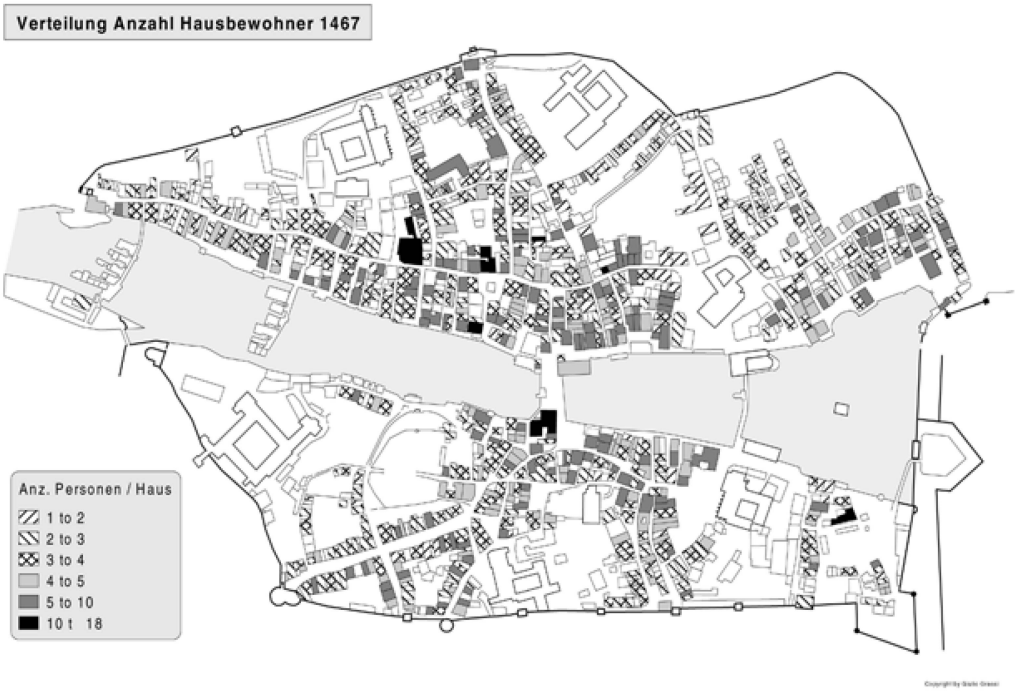


Abbildung 3: Verteilung der Anzahl Hausbewohner in der Stadt Zürich 1467

7. Zusammenfassung

Die Regeln der quellennahen Datenverarbeitung und die wichtigsten Anforderungen an ein DBMS für HistorikerInnen können in folgenden Punkten zusammengefasst werden:[22]

1. *Strukturabbildung (Modellierung) und Kategorisierung der Quelleninhalte:*
– Exakte Abbildung der logischen Dokumentenstruktur
– Berücksichtigung der Quellen-Variabilität trotz Modellierung (Datensatzsensitivität)
– Modularer Aufbau mit nachträglichem Datensatz-Linking (Nominal Record Linking)
– Sekundäre, benutzerspezifische Kategorisierung über den Feldinhalten (Postkodierung)

22 Ohne dabei Anspruch auf Vollständigkeit zu erheben.

2. *Intelligentes Retrieval*
– Logische Operatoren (UND, ODER, NICHT), Wildcards, reguläre Ausdrücke etc.
– Handling von Schreibungsvarianten durch benutzerdefinierte Ausgleichsalgorithmen (z.B. Soundex) oder Kodierungen

3. *Spezialanforderungen an das Feld*
– Mehrfacheinträge, z.B. für zwei Berufe
– Unbegrenzte Länge – kein Unterschied zwischen Kurzinfo-Feldern und Volltextfeldern
– Spezielle, z.T. benutzerdefinierte Feldtypen: Währungen, Datum u.a.m.
– Angebundene Zusatzfelder für Kommentare, Originaleinträge, Anmerkungen (gleiche Recherche)
– Tags: «?», «!» etc. zur Markierung der Feldinhalte

4. *Visualisierung raumbezogener Daten*
– Kartographische Visualisierung räumlicher Entitäten und Untersuchungsergebnisse (thematische Karten)