Patientendatenerfassung in RDBMS

Diplomarbeit

Universität Rostock, Fachbereich Informatik

vorgelegt von

Murr, Steffen
geb. am 14. Februar 1969 in Rostock

Betreuer: Prof. Dr. Andreas Heuer
Dipl.-Math. Bernd Griefahn

Abgabedatum: 15. Juni 1998

Zusammenfassung

Die Diplomarbeit ist ein Anwenderprojekt der Medizinischen Fakultät der Universität Rostock. Das Ergebnis dieser Arbeit ist eine Client/Server Datenbank-Applikation. Die Applikation wird im Archiv des Zentrums für Nervenheilkunde, dem vier Kliniken angehören, eingesetzt. In einer dafür installierten Domäne von Windows NT 4.0, kommt als Front-End MS ACCESS 97 und als Back-End, der MS-SQL Server 6.5 zum Einsatz. Die Entwicklung der Applikation umfaßte insbesondere alle Phasen des Softwarelebenszyklusses und stellt ein Archivierungssystem von stationären Patientenaufnahmen dar. Durch Hinzufügen klinikspezifischer Daten, wird es auch als Informationssystem von den Kliniken genutzt. Der Datenschutz wird durch eine entsprechende Rechtevergabe gewährleistet. Die Arbeit beschreibt wichtige Aspekte der Client/Server-Technologie und wie Front- und Back-End, durch verschiedene Methoden, auf der Grundlage von ODBC, Daten austauschen können. Die Entwicklung der Applikation wird vollständig beschrieben und an ausgewählten Beispielen veranschaulicht.

Abstract

This thesis is produced in conjunction with the medical faculty of the University of Rostock. The culmination of this project is a Client/Server database application which will be used in the archives of the neurology center which spans four hospitals. The components of the Windows NT 4.0 domain are MS ACCESS 97 as a front end and MS SQL Server 6.5 as a back end. The development of this application involved all phases of the software life cycle. The main use of this application is as an archive system for patient admission. As it includes the ability to add specific clinical data it is also usable as a clinical information server. Data security is guaranteed by controlled permission access. This thesis describes important aspects of the client/server technology, and data exchange between ODBC based front and back ends. The development of this project is described completely and illustrated throughout with examples.

CR-Klassifikation

H.2 Database Management
H.2.1 Logical Design
H.2.2 Physical Design

Keywords

Relationales DBMS, Client/Server-Architektur, Front-End, Back-End

Danksagung

Die Arbeit wäre ohne die Mithilfe vieler weiterer Personen kaum möglich gewesen, denen ich an dieser Stelle recht herzlich danken möchte. Mein besonderer Dank gilt Prof. Dr. Andreas Heuer und Dipl.-Math. Bernd Griefahn für die umfangreiche fachliche Unterstützung, sowie weiteren Mitarbeitern des Fachbereichs Informatik. Insbesondere hat Dipl.-Inf. Jürgen Schlegelmilch meine LaTeX-spezifischen Fragen geduldig ertragen und war stets ein hilfsbereiter Ansprechpartner. Ganz besonders möchte ich mich auch bei Dr. Uwe Zettl für die Medizinfachliche und die organisatorische Betreuung bedanken. Großen Anteil an dem Gelingen der Arbeit haben auch die Vertreter der Kliniken Dr. Schläfke, Dr. Häßler und Dr. Klauer, die bei der Analyse sehr entgegenkommend waren. Mit großem Engagement unterstützten mich Frau Richter und Frau Liermann bei der Analyse der Archivarbeit. Abschließend möchte ich mich bei meiner Frau Katrin für die großartige Unterstützung während dieser Zeit bedanken, in der zuweilen sehr wenig Zeit für die Familie blieb.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Die Diplomarbeit ist ein Anwenderprojekt der Medizinischen Fakultät der Universität Rostock. Das Ziel der Arbeit bestand darin, ein relationales Client/Server-Datenbanksystem für die Archivierung von Patientendaten zu entwickeln. Die Archivarbeit des Zentrums für Nervenheilkunde wurde bisher nicht durch die elektronische Datenverarbeitung unterstützt. Das Anwenderprojekt erhielt den Namen PDAZN und stellt ein Akronym für Patientendatenarchiv des Zentrums für Nervenheilkunde dar.
Die Herausforderung bei der Entwicklung des PDAZN bestand unter anderem darin, eine umfassende Analyse des Informationsflusses vorzunehmen, um daraus die Funktionen für die Applikation abzuleiten. Dazu war es notwendig zu analysieren, *welche* Informationen benötigt werden, *zwischen welchen* Einrichtungen und *wie* sie ausgetauscht werden. Die Aufgabenstellung sah für dieses Projekt zunächst eine Archivierungsapplikation vor, die sich aufgrund der Anwenderwünsche darüber hinaus, zu einem Informationssystem für die ansässigen Kliniken des Zentrums für Nervenheilkunde, kurz ZN, entwickelte. Dazu war eine Rechtevergabe auf der Grundlage des zu beachtenden Datenschutzes erforderlich.

Die Durchführung des Projektes erforderte die Analyse und die Bestellung sicherer Hard- und Software, die eigens für diese Applikation im Archiv installiert wurde. Als relationales Client/Server-Datenbankmanagementsystem kommt der MS SQL Server 6.5 und als Datenbank-Front-End MS ACCESS 97 zum Einsatz. Zur Hardware gehört ein PC-Server, ein PC-Client, eine USV und ein Laserdrucker. Die Komponenten wurden an das Subnetz der Medizinischen Fakultät angeschlossen und waren entsprechend der Aufgabenstellung auch zu administrieren. Unter dem Betriebssystem Windows NT 4.0 wurde der Server als Primärer Domänen Controller (PDC) für eine eigene Domäne (ZN) eingerichtet.

Zunächst werden in Abschnitt 1.1 einige Informationen zum zukünftigen Anwender gegeben. In Abschnitt 1.2 werden die Ziele der zu entwickelnden Applikation und im Abschnitt 1.3 der weitere Aufbau der Arbeit beschrieben.

1.1 Einordnung des Zentrums für Nervenheilkunde

Das Zentrum für Nervenheilkunde gehört zur Medizinischen Fakultät der Universität Rostock und hat seinen Standort in Rostock - Gehlsdorf.
Zum Zentrum gehören die nachfolgend genannten vier Kliniken und ein Institut:

- Klinik und Poliklinik für Kinder- und Jugendneuropsychiatrie/ Psychotherapie, kurz: Kinder- und Jugendklinik
- Klinik für Neurologie und Poliklinik, kurz: Neurologie
- Klinik und Poliklinik für Psychiatrie und Psychotherapie, kurz: Psychiatrie
- Klinik und Poliklinik für Psychosomatik und psychotherapeutische Medizin, kurz Psychosomatik
- Institut für Psychologie

Die Abbildung 1.1 zeigt die Struktur des Zentrums für Nervenheilkunde, wobei das Archiv im Mittelpunkt der Betrachtungen steht. Es werden nur die Einrichtungen abgebildet, die am Informationsaustausch beteiligt sind. Das Institut für Psychologie ist für die nachfolgenden Betrachtungen nicht relevant, da die Aufgabenstellung nicht die Eingliederung des Instituts in den Informationsverbund vorsah.
In dem Archiv des ZN werden alle stationären Patientenaufnahmen der genannten vier Kliniken archiviert.
Dem ZN gehört weiterhin ein Patientenwesen an, das vor der Einführung des PDAZN auch als Informationsquelle für das Archiv diente. Das Patientenwesen hat primär die Aufgabe, die Daten der aufgenommenen Patienten elektronisch zu erfassen, um die Kostenabrechnung zu gewährleisten. Der Datenaustausch erfolgte zwischen den Kliniken, dem Patientenwesen und dem Archiv ohne Verwendung der elektronischen Medien. Die Informationen wurden durch ausgedruckte Listen etc. oder persönlich weitergegeben.

1.2 Ziel der zu entwickelnden Applikation

Die zu entwickelnde Applikation hat die Aufgabe, die Arbeit des Archivs wesentlich zu erleichtern. Bisher wurde die gesamte Archivierungsarbeit ohne Einsatz der elektronischen Datenverarbeitung durchgeführt. Die Erfassung von stationären Patientenaufnahmen wurde an Hand von Listen vorgenommen, die das Archiv über das Patientenwesen bezog. Dazu wurden Aufnahmebücher geschrieben, die als eine Art Index zu verstehen sind, um auf die wichtigsten Informationen einer Patientenaufnahme schnell zugreifen zu können.

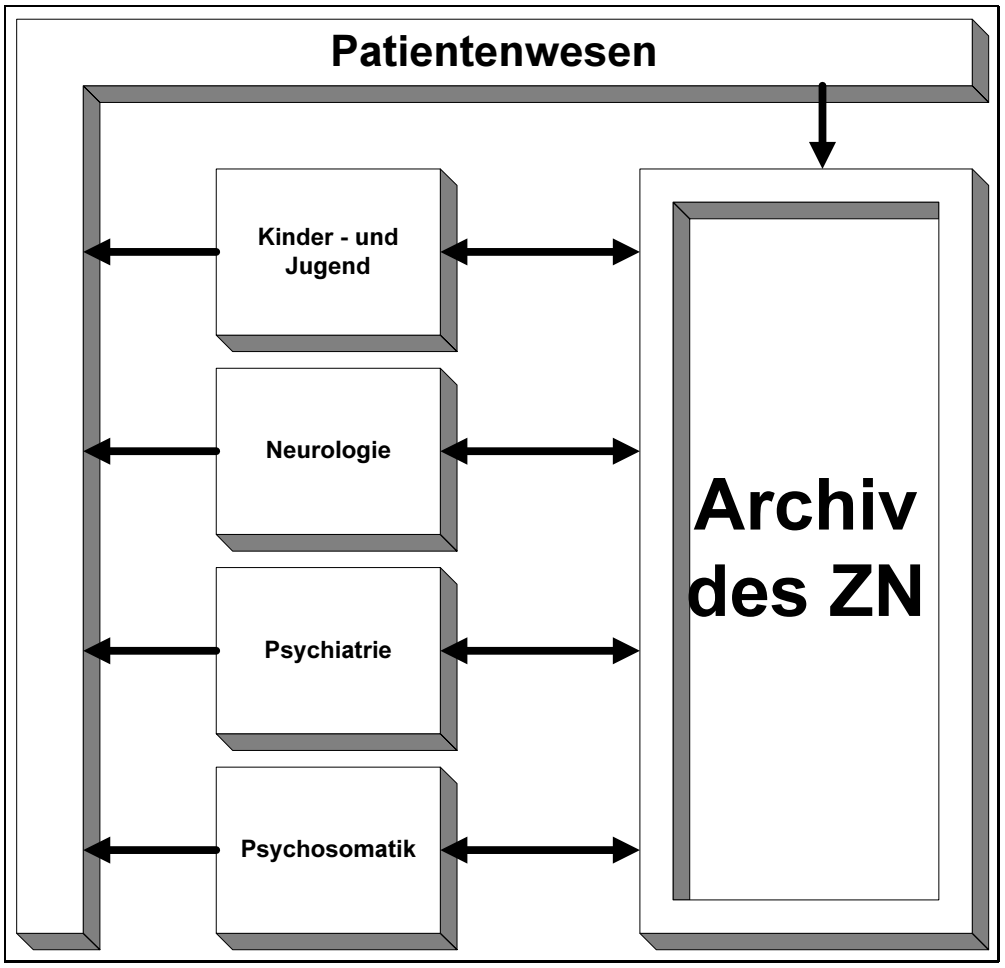


Abbildung 1.1: Informationsfluß im Zentrum für Nervenheilkunde

Das PDAZN stellt eine Patientendatenverwaltung dar, die den Zweck eines Archivierungssystems, mit allen dazu erforderlichen Funktionen, erfüllen soll. Es ist nicht Ziel der Arbeit, die Patientenakten, die in Papierform vorliegen, vollständig elektronisch zu speichern, auch wenn das ein langfristig erstrebenswertes Ziel darstellt. Es sollen nur ausgewählte Daten erfaßt werden, die dem Archiv die Informationen liefern können, die für die tägliche Archivierungsarbeit relevant sind. Dazu gehören neben Patientendaten wie Name, Geburtsdatum, Geschlecht, auch Informationen wie Aufnahmedatum, Aufnahmemodus, Station, etc.. Aufgrund dieser Daten können schnell Anfragen der Kliniken beantwortet werden. Ohne Einsatz der Applikation beanspruchen Anfragen, bei entsprechender Komplexität und einem gewissen Datenbestand, besonders viel Zeit. Repräsentativ ist die folgende Anfrage:

Geben Sie mir bitte alle Patienten im Alter von 60-70 Jahren, die in dem Zeitraum B mit der Diagnose C des Diagnosesystems D stationär aufgenommen wurden!

Besonders deutlich werden hier die Vorteile der elektronischen Datenverarbeitung. Die Applikation ermöglicht die Beantwortung dieser Anfragen innerhalb weniger Sekunden, während sie ohne Applikation mehrere Stunden dauern kann.
Die Erfassung von neuen Patientenaufnahmen wird durch den Import von Patientendaten aus dem Datenarchiv der Medizinischen Fakultät, besonders erleichtert. Das Datenarchiv wird von dem Bereich DVSE (Datenverarbeitungs- Service und Einrichtung) betreut und administriert. Die DVSE ist außerdem für die klinikumsweite Betreuung und Versorgung mit Soft- und Hardware zuständig. Vorrangiges Aufgabengebiet stellt die abrechnungstechnische Seite einer Patientenaufnahme dar. Aufgrund des Datenarchivs stehen eine Vielzahl von Daten elektronisch zur Verfügung, die durch das Patientenwesen eingegeben wurden. Für das Archiv besteht die schwierige Aufgabe darin, die Daten einer neuen Aufnahme einem eventuell schon vorhandenen Patienten in der Archiv-Datenbank zuzuordnen.

Die Applikation wurde darüber hinaus zu einem Informationssystem für die vier ansässigen Kliniken des Zentrums. Dazu wurden spezifische Daten der Kliniken analysiert und implementiert, die für die Mitarbeiter einer Klinik einzusehen und zum Teil auch zu bearbeiten sind. Die Mitarbeiter anderer Kliniken können ebenfalls, bei entsprechender Authorisierung, diese Daten einsehen, aber nicht bearbeiten.

Soweit es im Rahmen dieser Arbeit möglich war, sollte das PDAZN größtmögliche Flexibilität bieten. Falls zum Beispiel die Psychosomatik in Zukunft eine zusätzliche Station erhält, muß die Applikation mit relativ geringem Aufwand an diese neue Situation anpaßbar sein. Ebenso verhält es sich mit der Erweiterung von spezifischen Daten der Kliniken. Als Beispiel soll hier die Erweiterung um ein weiteres Verfahren bzw. einer Methode zur Ermittlung des psychopathologischen Befundes genannt werden, die bisher vielleicht noch nicht in der Praxis als einsatzfähig galten, aber nun doch zu den Befunden hinzugefügt werden sollen. Selbst der Versicherungsstatus[1] wurde nicht im System fest „verdrahtet“, sondern ist abänder- und erweiterbar.

[1] Zur Zeit wird der Versicherungsstatus durch F-Familie, M-Mitglied, R-Rentner präsentiert.

1.3 Überblick über weitere Kapitel

Im folgenden wird eine kurze Übersicht über den weiteren Aufbau der Arbeit gegeben:

- Das **Kapitel 2** beschreibt die Grundlagen der Client/Server-Technologie und wie das PDAZN diesbezüglich einzuordnen ist.
- Das **Kapitel 3** gibt einen Überblick über die eingesetzte Soft- und Hardware. Insbesondere stehen hier Applikations-spezifische Betrachtungen im Vordergrund.
- In **Kapitel 4** wird die vollständige Entwicklung des PDAZN, angefangen von der Analyse bis zum Test, beschrieben und an ausgewählten Beispielen geschildert.
- Das **Kapitel 5** gibt eine Zusammenfassung über die Arbeit und inwieweit Erweiterungen wünschenswert oder denkbar sind.
- Der **Anhang** enthält das vollständige ER-Modell und alle abgeleiteten Relationenschemata.

Kapitel 2

Client/Server Technologie

In den letzten Jahren gewann die Client/Server-Technologie immer mehr an Bedeutung, dessen Interpretation recht unterschiedlich ausfällt. Die Client/Server-Technologie ist aufgrund der Popularität, auch zu einem Verkaufsargument für die Soft- und Hardware geworden.

Die Begriffe Client und Server werden sowohl im softwaretechnischen, als auch im hardwaretechnischen Kontext verwendet. Im hardwaretechnischen Kontext stellt der Server meistens einen Rechner dar, der wesentlich bessere Ausstattungsmerkmale, als andere Rechner seiner Umgebung bietet. Dazu gehören die Prozessoren, die Hauptspeicherausstattung, hochentwickelte und damit sichere Lüftungssysteme, die Bussysteme und weitere mehr. Zum Server wird ein Rechner aber erst durch die Installation einer bestimmten Software. Sie bestimmt durch ihre Anforderungen die Rechnerausstattung.

Was ist nun aus softwaretechnischer Sicht ein Client und was ist ein Server? Dazu werden zunächst die folgenden zwei Definitionen gegeben:

Definition 2.1 (Client) *Als Client wird ein Programm bezeichnet, das bestimmte Ressourcen (Funktionen, Objekte, Dienste) eines Diensteanbieters nutzt.*

Definition 2.2 (Server) *Als Server wird ein Programm bezeichnet, das bestimmte Ressourcen (Funktionen, Objekte, Dienste) anderen Programmen zur Verfügung stellt.*

Die Definitionen besagen, daß der Client bestimmte Aufträge an den Server weiterreicht, zu deren Lösungen der Server zum Beispiel bestimmte Funktionen zur Verfügung stellt. Der Server erwartet Aufträge und sendet das Ergebnis nach erfolgter Bearbeitung an den Client zurück. Ein Datenbankserver erhält beispielsweise eine Anfrage und schickt nach deren Bearbeitung Datensätze zurück, während ein Print-Server das Ergebnis in Form von Ausgaben auf den Drucker zur Verfügung stellt, nach dem dieser einen Druckauftrag erhalten.

Eine Client/Server-Beziehung ist jedoch immer kontextabhängig. Es hängt von der Aufgabenstellung ab, ob gerade die Rolle als Client oder als Server übernommen wird. Deshalb ist es möglich, daß ein Server, der einen Auftrag von einem Client erhalten hat, selbst zum Client wird, wenn er seinerseits wieder einen Auftrag an einen anderen Server übergibt, um den Auftrag auszuführen. Die Abbildung 2.1 verdeutlicht graphisch diesen Sachverhalt. Die Rechner unterscheiden sich aufgrund ihrer zur Verfügung gestellten Dienste in ihrer

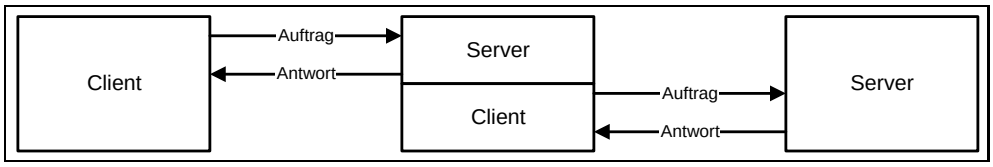


Abbildung 2.1: Verschiedene Rollen bei Auftragsausführungen

Hardwareausstattung. Durch die angebotenen Dienste werden so beispielsweise Datenbank-, Applikations-, File- und Print-Server.
Auch diese Rechner werden in dem Moment zum Client, wenn sie auf Ressourcen anderer Rechner zugreifen. Der Datenbankserver ist zum Beispiel nur aus Sicht des Datenbankzugriffs ein Server, wenn außerdem noch ein File-Server zum Einsatz kommt. Greift der Datenbank-Server auf eine Datei des File-Servers, wird er selbst zum Client.

Entscheidendes Merkmal der Client/Server-Technologie ist die Aufteilung einer Aufgabe auf mehrere Rechner, um die Gesamtaufgabe erfüllen zu können.
Die Aufgaben werden im Gegensatz zu Host-Systemen nicht zentral auf einem Rechner ausgeführt, sondern auf mehrere Rechner aufgeteilt. Die Rechner in Client/Server-Systemen verfügen alle über gewisse Leistungspotentiale, die weit über die Nutzung als Terminal hinausgehen. Das große Potential dieser Technologie besteht darin, die Aufgaben dort auszuführen, wo ihr Ergebnis benötigt wird. Die verschiedenen Architekturen der Client/Server-Technologie, die im folgenden Abschnitt verdeutlicht werden, unterstreichen diesen Aspekt.

2.1 Client/Server-Architekturen

Bevor eine Definition der Client/Server-Architektur gegeben wird, werden die in [Bers92] ausführlich beschriebenen Merkmale, die dieser Architektur zugrunde liegen, hier kurz genannt:

- Die *kooperative Informationsverarbeitung* bedeutet eine Zusammenarbeit von Prozessen auf unterschiedlichen Rechnern, um ein gemeinsames Ziel zu erreichen. Die Gesamtaufgabe wird auf Client und Server je nach Architektur aufgeteilt.
- Ein *offenes System* muß folgende Hauptkriterien erfüllen:
  - Übereinstimmung mit Industrie-Standards für Programmierung, Kommunikation, Netzwerk, Systemmanagement, Präsentation, etc.
  - Portabilität
  - Skalierbarkeit
  - Interoperabilität (Zusammenarbeit über die Systemgrenzen hinweg)

Diese beiden Merkmale bilden die Grundlage der Client/Server-Technologie. Die Nutzung von Servern durch Clients findet, dank offener Systeme, nicht nur in homogenen, sondern auch in heterogenen Umgebungen statt.

Zum einheitlichen Verständnis der Client/Server-Architektur, wird die folgende Definition nach [KaMü94] gegeben:

Definition 2.3 (Client/Server-Architektur) *Eine Client/Server-Architektur ermöglicht die verteilte, kooperative Informationsverarbeitung in offenen Systemen. Die Kooperation findet sowohl zwischen Softwarekomponenten, als auch zwischen den Hardwarekomponenten statt.*

Ein Anwendungssystem besteht aus mehreren Komponenten, die beispielhaft in der Abbildung 2.2 dargestellt sind. Die Applikation kann in die Aufgabenbereiche *Präsentations-Logik*, *Applikations-Logik* und *Daten-Management* untergliedert werden. In einem Standalone- oder Host-System befindet sich die Applikation vollständig mit ihren Aufgabenbereichen auf einem Rechner.
Durch die Client/Server-Technologie werden bestimmte Aufgabenbereiche zum Teil oder vollständig auf andere Rechner übertragen. Entsprechend des Umfangs der Verteilung entstehen unterschiedliche Architekturen.

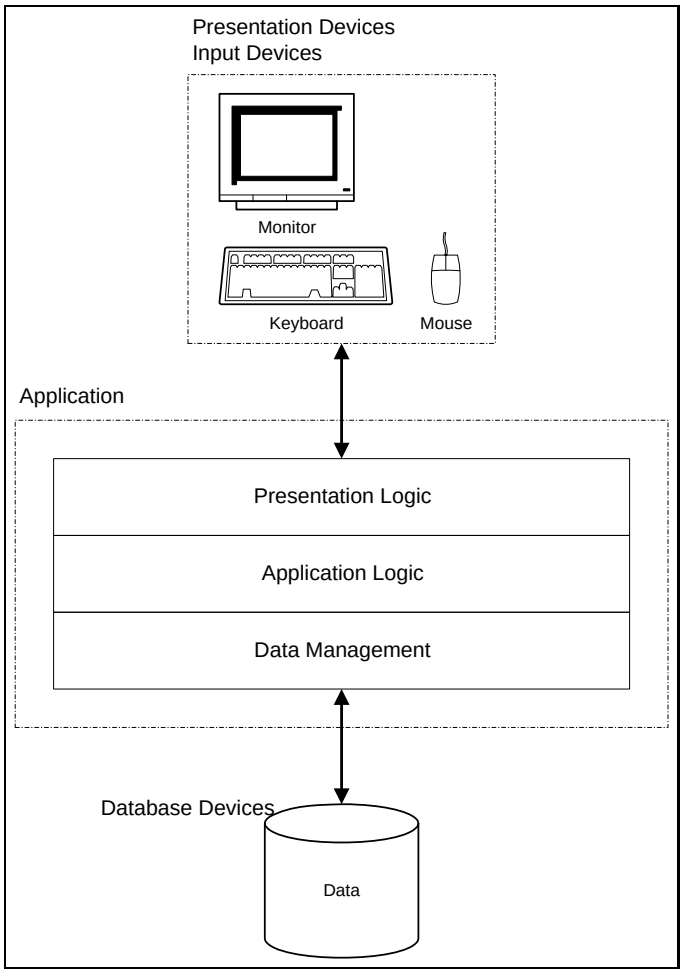


Abbildung 2.2: Die Komponenten eines Anwendungssystems und die Aufgabenbereiche einer Applikation

Die Aufgabenbereiche werden im folgenden nach [Dada96] definiert:

Definition 2.4 (Präsentations-Logik) *Der Präsentations-Logik werden alle Aufgaben zugeordnet, die mit der Eingabe von Informationen via Tastatur oder Maus, sowie mit deren Ausgabe in Zusammenhang stehen.*

Die Präsentations-Logik enthält alle Funktionen, die für die Darstellung der Daten auf dem Front-End notwendig sind. Dazu gehört zum Beispiel das Füllen von Masken mit Ausgabewerten und das Selektieren bestimmter Werte in Formularen oder Masken via Maus oder Tastatur.

Definition 2.5 (Applikations-Logik) *Die Applikations-Logik repräsentiert die gesamte anwendungsspezifische Programm- und Ablauflogik. Sie ruft für die dialogorientierte Ein- und Ausgabe Präsentationsfunktionen und für den Zugriff auf gespeicherte Daten entsprechende Datenverwaltungsfunktionen auf.*

Die Applikations-Logik enthält den spezifischen Kern einer Applikation und überwacht deren Geschäftsregeln. Dazu zählen die folgenden Aufgaben, die bei weitem nicht vollständig sind: Authorisierungsmechanismen, benutzergruppenabhängige Menüsteuerungen und Ereignisbehandlungen bei Nutzeraktionen (Interaktionssteuerung). Durch die Applikations-Logik wird beispielsweise auch festgelegt, welche Transaktionen zusammengefaßt werden müssen, um die Datenintegrität gewährleisten.

Definition 2.6 (Daten-Management) *Das Daten-Management realisiert den physischen Zugriff auf die in der Datenbank gespeicherten Daten und stellt diese ggf. an den entsprechenden Schnittstellen bereit.*

Das Daten-Management ist für den Zugriff und die Abspeicherung von Daten unter Einhaltung des Transaktionsprinzips verantwortlich, wobei auch die Mehrbenutzerkontrolle unterstützt wird. Bei konkurrierenden Zugriffen sorgen Sperrmechanismen für den ordnungsgemäßen Datenzugriff unter Einhaltung der Datenintegritat. Diese Mechanismen können durch den Datenbankadministrator beeinflußt werden, indem bestimmte Feineinstellungen bezüglich der internen Ebene vorgenommen werden können. Dazu gehört die Größe des zu sperrenden Bereiches und die Auswahl von optimistischen oder pessimistischen Sperrmechanismen. Das Datenmanagement muß darüber hinaus die Datensicherheit und den Datenschutz gewährleisten. In [HeSa95] erfolgt eine ausführliche Beschreibung der Architektur und der Aufgaben eines DBMS.

Mit der Definition der drei Aufgabenbereiche wurde die Grundlage für das Verständnis des Client/Server-Modells der Gartner-Group, vorgestellt in [KaMü94], geschaffen, das in Abbildung 2.3 dargestellt wird. Das Modell ist so konstruiert, daß in Abhängigkeit der Aufgabenverteilung auf Client und Server, die folgenden fünf Architekturvarianten entstehen:

- Distributed Presentation
- Remote Presentation
- Distributed Logic
- Remote Data Management
- Distributed Database

Client und Server sind dabei über ein Netzwerk miteinander verbunden. Es existieren auch Misch- oder Pseudoformen, die an entsprechender Stelle der folgenden Abschnitte beschrieben werden, in denen die Vorstellung der Architekturvarianten erfolgt.

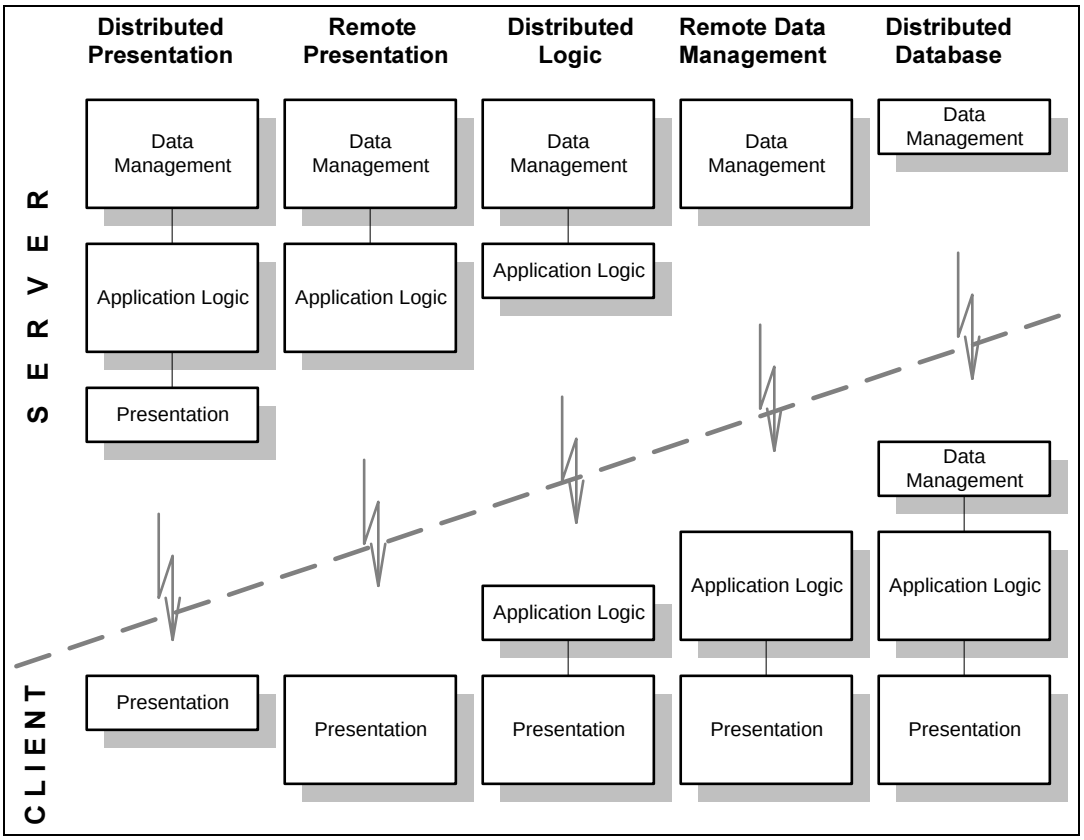


Abbildung 2.3: Client/Server-Architektur in Anlehnung der Gartner-Group

2.1.1 Distributed Presentation

Ins Deutsche mit "Verteilte Präsentation" übersetzt, wird bei dieser Architekturvariante ein Teil der Präsentations-Logik auf den Client verlagert. Der andere Teil verbleibt zusammen mit der Applikations-Logik und dem Daten-Management auf dem Server.
Ein Beispiel stellt das X-Windows-System dar, unter der Voraussetzung, daß zwei Rechner eingesetzt werden. Das X-Windows-System besteht im wesentlichen aus zwei Komponenten: einem X-(Display)-Server und einem oder mehreren X-Clients. Der X-Server wird von den X-Clients bedient, wodurch allgemein etwas begriffliche Verwirrung gestiftet wird. Der X-Server verwaltet die Fenster seiner Clients und stellt sie dar. Dagegen liegen die X-Clients, außer auf dem gleichen Rechner des X-Servers, auch auf anderen Rechnern und bedienen den Server entsprechend mit Darstellungsinformationen.
Im sonst üblichen Client/Server-Sprachgebrauch bedient grundsätzlich der Server den Client und das ist bei X-Windows genau umgekehrt. Deshalb würde der Rechner auf dem der X-Server installiert ist, auch eher als Client-Rechner und der Rechner auf dem ein X-Client installiert ist, als Server-Rechner bezeichnet werden.

Wichtig bei der Benutzung des X-Windows-Systems ist jedoch, daß auf dem Bildschirm eines Anwenders mehrere Anwendungen durch den X-Server dargestellt werden können. Die Präsentations-Logik wird dadurch teilweise auf dem Client (Rechner mit dem X-Server) ausgelagert.

2.1.2 Remote Presentation

Die "Entfernte Präsentation" wird erreicht, in dem die Präsentations-Logik vollständig auf den Client verlagert wird.
Die Fachwelt ist sich offensichtlich nicht darüber einig, wo genau der Übergang zwischen der Host- und der Client/Server-Technologie festzulegen ist. Im Kontext der Client/Server-Technologie soll hier die Festlegung getroffen werden, daß eine Darstellung der Informationen durch ein einfaches alphanumerisches Terminal nicht zu dieser Architekturvariante gehören soll. Viele Autoren [Dada96] ziehen die Grenze dann, wenn das einfache Terminal durch einen intelligenten Rechner ersetzt wird und und die Verbindung zu einem Host-System durch eine Terminal-Emulation erfolgt.
Dadurch wird manchen Unternehmen die Möglichkeit eröffnet, ihre Produkte mit in dem Zusammenhang der Client/Server-Technologie zu nennen, obwohl im Hintergrund eine Host-basierte Umgebung existiert.
Der Schwerpunkt der Anwendung liegt auch bei dieser Architektur immer noch auf dem Server und hat damit Aufgaben auszuführen, die auch ein Client übernehmen könnte.

2.1.3 Distributed Logic

Der Architekturtyp "Verteilte Applikations-Logik", verlagert einen Teil der Applikations-Logik auf den Client. Je nach Applikation kann der Anteil mehr auf Client- oder mehr auf Server-Seite verlagert werden. Je mehr Aufgaben der Client erledigen kann, desto mehr wird der Server entlastet, der dann zum Beispiel mehr Ressourcen für die Bearbeitung von Aufgaben anderer anfragender Prozesse zur Verfügung hat.
Echte Client/Server-DBMS bieten zum Beispiel Features, wie Stored Procedures an, die es ermöglichen, einen Teil der Applikations-Logik auszuführen. Der größte Teil der Applikations-Logik befindet sich zumeist auf dem Client, der zum Beispiel für die Programmsteuerung etc. zuständig ist.
In [KaMü94] wird die horizontale Verteilung, als eine Architekturvariante erwähnt, die im eigentlichen Sinne nur die Präsentations-Logik auf den Client verlagert, indem beispielsweise Applikations-Server zum Einsatz kommen, denn die Applikations-Logik befindet sich dann auf diesem Server. Deshalb wird diese horizontale Verteilung auch als Pseudo-Architektur bezeichnet. Dagegen stellt eine vertikale Verteilung eine echte Architektur dar.

2.1.4 Remote Data Management

Das "Entfernte Daten-Management" stellt eine Architektur dar, bei der die meisten Aufgaben vom Client übernommen werden und nur das Daten-Managament auf dem Server verbleibt.
Bei den bereits erläuterten Architekturtypen, befindet sich das Daten-Management ebenfalls vollständig auf dem Server. Der Anwender darf selbstverständlich nicht merken, ob er auf eine entfernte oder auf eine lokale Datenbank zugreift.
Das Potential einer Client/Server-Anwendung kann bei dieser Architektur nicht voll ausgeschöpft werden, da der Server nicht, mit seinem mitunter beträchtlichen Leistungspotential, für Erfüllung anderer Aufgaben zur Verfügung steht.

2.1.5 Distributed Database

Entsprechend der Gartner Group wird außer einem Teil des Daten-Managements auch die Präsentations- und die Applikations-Logik auf den Client verlagert Die dadurch erhaltene Architekturvariante ist die "Verteilte Datenbank". Auf dem Server befindet sich demnach nur noch ein Teil des Daten-Managements.
Beim Zugriff auf die Daten der Applikation soll der Anwender nicht bemerken, auf welchem Rechner sich die Daten befinden. Das gewährleistet die Transparenz, als eine Forderung bei verteilten Datenbanken. In [Dada96] wird ausführlich die Problematik der verteilten Datenbanken diskutiert.
Bei dieser Architektur können die Daten dort gespeichert werden, wo sie am ehesten benötigt werden. Dazu werden in den jeweiligen Einrichtungen, in denen auf bestimmte Daten besonders häufig zugegriffen wird, weitere Datenbank-Server installiert, die diese Daten als Fragmente der gesamten Datenbank speichern. Zusätzlich ist auch die Replikation möglich. Die Client-Rechner greifen dann über das verteilte DBMS auf die Daten der einzelnen Server zu.
Ein noch komplexerer Verteilungsfall entsteht, wenn auch ein Teil der Applikations-Logik auf dem Server ausgeführt wird.

2.2 Einordnung des PDAZN

Für das PDAZN wird als Front-End MS Access 97 und als Back-End der SQL-Server 6.5 eingesetzt.
Access spielt dabei eine besondere Rolle, da es ein eigenes DBMS besitzt. Dadurch können Daten der Server-Tabellen für die Clients repliziert werden. Auf diese Daten kann Access dann lokal, durch das eigene DBMS zugreifen. Dennoch ist diese Konstellation bei weitem noch keine verteilte Datenbank. Der lokale Zugriff auf Daten, die selten aktualisiert werden,

ist aber eine effiziente Methode. Im Rahmen dieser Arbeit kam diese Methode jedoch nicht zum Einsatz, da bezüglich der Zugriffsgeschwindigkeit keine Notwendigkeit bestand.
Auch wenn erst durch den Entwicklungsprozeß, beschrieben in Kapitel 4, der Architekturtyp entsteht, soll an dieser Stelle vorweg genommen werden, daß das PDAZN in den Architekturtyp "Verteilte Applikations-Logik "einzuordnen ist.
Dabei wird die Präsentations-Logik vollständig und die Applikations-Logik zum größten Teil auf den Client verlagert.
Eine kleine Besonderheit gibt es beim PDAZN bezüglich der Nutzung des Servers. Dieser wird aus Kostengründen auch als Front-End und dadurch als Client eingesetzt. Der Client soll aber im weiteren als separater Rechner betrachtet werden.
Der andere Teil der Applikations-Logik wird auf dem Server durch den Einsatz von Stored Procedures ermöglicht, die in Abschnitt 3.1.1.4 beschrieben werden. Die Stored Procedures enthalten SQL-Abfragen, die teilweise durch explizite Transaktionen eingeschachtelt werden, um die Datenkonsistenz zu sichern. Dabei ist eine Transaktion eine Folge logisch zusammengehöriger Aktionen, beispielsweise zur Abspeicherung von Daten einer Patientenaufnahme in mehreren Tabellen.
Das Daten-Management befindet sich vollständig auf dem Server. Das DBMS von Access kann nicht das DBMS des Servers umgehen, um auf die Daten zuzugreifen. Dennoch ist es eine wichtige Unterstützung, die auch in Anspruch genommen wird, um zum Beispiel auf kleine Server-Tabellen zuzugreifen, ohne dafür speziellen Programm-Code schreiben zu müssen. Dieser Zugriff wird durch die verknüpften Tabellen realisiert, die in Abschnitt 3.1.5.1 beschrieben werden.
Die Sicherung der Datenintegrität, der Datenschutz und die Datensicherung liegen allein in der Verantwortung des SQL-Servers.

Die Daten aller Patientaufnahmen und damit auch die spezifischen Daten der Kliniken, liegen zentral auf dem Server im Archiv des ZN. Für die Zukunft wird dieses Modell aufgrund der zu erwartenden Daten ausreichend sein. Bei einem hohen Datenaufkommen wäre es möglich, den komplexeren Architekturtyp "Verteilte Datenbanken" anzustreben. Dabei würde jede Klinik einen Datenbankserver erhalten, der die spezifischen Daten der jeweiligen Klinik abspeichert. Allgemeinere Daten würden weiterhin auf dem Server des Archivs abgespeichert werden.

Kapitel 3

Die Soft- und Hardware

Dieses Kapitel beschäftigt sich mit der eingesetzten Soft- und Hardware, wobei auf besonders wichtige Eigenschaften der Software bezüglich des PDAZN in Abschnitt 3.1 detaillierter eingegangen wird. Die Vorstellung der zum Einsatz kommenden Hardware erfolgt in Abschnitt 3.2.

3.1 Die eingesetzte Software

Bei der Beschaffung der Software bestand der Wunsch des Anwenders, Microsoft-Produkte einzusetzen, um Synergieeffekte bei der Nutzung des Betriebssystem und der Applikation auszunutzen, da bereits eine Vielzahl von Rechnern im ZN mit Windows 95 und dem Office-Paket von Microsoft ausgestattet sind. Dadurch betreffen die Schulungen im wesentlichen nur die neue Applikation PDAZN.
Diesbezüglich wurde MS Access 97 als Front-End gewählt. Als Back-End kommt der MS SQL-Server 6.5 zum Einsatz. Aufgrund der Installationsbedingung wurde das Betriebssystem Windows NT 4.0 zum Einsatz gebracht.

Zunächst wird im Abschnitt 3.1.1 der MS SQL-Server näher vorgestellt, da er erst in jüngster Zeit durch den breiten Einsatz von Windows NT an Bedeutung zunahm und deshalb noch nicht so sehr bekannt ist. Im Abschnitt 3.1.2 soll MS Access 97 hinsichtlich des Einsatzes als Front-End vorgestellt werden.
Als Betriebssystem wurde Windows NT installiert und wird in Abschnitt 3.1.3 kurz dargestellt.
Der Abschnitt 3.1.4 stellt eine kurze Einführung in ODBC dar, da es die Voraussetzung für den Datenaustausch zwischen MS SQL Server und MS Access 97 bildet.
Der letzte Abschnitt 3.1.5, befaßt sich mit den verschiedenen Datenaustauschmethoden zwischen der genannten Software.

3.1.1 MS SQL-SERVER 6.5

Als relationales Datenbankmanagementsystem (RDBMS) bietet Microsoft den SQL-Server in der aktuellen Version 6.5 an, der sich sehr gut für die zu entwickelnde Applikation eignet. Es ist ein Client/Server DBMS aus der Backoffice-Produktfamilie, das auch das Konzept der verteilten Datenbanken unterstützt.
Es bietet eine hohe Integration mit dem Betriebssystem Windows NT. Besonders deutlich wird das bei der Wahl der integrierten Sicherheit, auf die im Abschnitt 3.1.1.2 näher eingegangen wird.
Weiterhin erfüllt der SQL-Server hohe Sicherheitsanforderungen und bietet Features wie Trigger und Stored Procedures an, die die Sicherung der Datenintegrität, aber auch entscheidende Vorteile für das Geschwindigkeitsverhalten einer echten Client/Server-Anwendung bieten.
Für die Administration besitzt das DBMS eine gute graphische Benutzeroberfläche, den SQL Enterprise Manager, mit der die wesentlichsten Administrationsaufgaben durchgeführt werden können.
Die Leistungsmerkmale des SQL Servers, in der Version 6.5, erfüllen die American National Standards Institute (ANSI) SQL-92-Standards. Weiterhin werden die Federal Information Processing Standards (FIPS 127-2), die durch das National Institute of Standards and Technology (NIST) festgelegt wurden, unterstützt.

Zu Beginn soll in Abschnitt 3.1.1.1 ein kleiner Überblick über die Entwicklung erfolgen. Anschließend wird in Abschnitt 3.1.1.2 das Sicherheitskonzept ausführlicher erläutert, da es für das PDAZN von großer Wichtigkeit ist und eine Voraussetzung für das Kapitel 4, der Entwicklung des PDAZN, darstellt.
Ebenfalls essentiell für die Entwicklung ist die Vorstellung des Distributed Management Object Model, kurz DMO-Modell in Abschnitt 3.1.1.3, sowie besondere Features wie Stored Procedures in Abschnitt 3.1.1.4.
Der letzte Teilabschnitt 3.1.1.5 erläutert die Installation des SQL-Servers für das PDAZN und welche wichtige Einstellungen diesbezüglich vorgenommen wurden.

3.1.1.1 Historie der Entwicklung

Um den MS SQL-Server 6.5 besser einordnen zu können, soll seine Entwicklung nachfolgend skizziert werden:

- 1973-1975: Ursprung in einem Universitätsprojekt (Universität von Kalifornien in Berkeley)
- 1981: Vorstellung des ersten kommerziellen Systems durch die Firma Britton Lee (gegründet von den ehemaligen Uni-Angehörigen Dr. Robert Epstein und Mark Hoffmann)

- 1986: Dr. Epstein und Mark Hoffmann verließen die Firma Britton Lee und gründeten SYBASE Inc. mit dem Ziel, eine offene Client/Server Architektur zu entwickeln
- 1987: erstes relationales DBMS durch Sybase verfügbar, das explizit für die Verteilung von Funktionalitäten über heterogene Client/Server Umgebungen hinweg entwickelt worden war
- 1988: Microsoft erwirbt als Partner von Sybase die Lizenz von SQL Server und gibt erste Version für OS/2 frei
- Anfang 90'er Jahre: Entwicklung für Windows NT mit dem Ziel, das DBMS eng mit dem Betriebssystem zu koppeln
- 1992: Erwerb des Quellcodes von Sybase durch Microsoft
- Ende 1993: Freigabe der SQL Server Version 4.2
- April 1994: Microsoft kündigt Kooperationsvertrag mit Sybase und entwickelt den SQL-Server allein weiter

Seit 1997 ist der SQL-Server in der Version 6.5 erhältlich, die auch im PDAZN zum Einsatz kommt.

3.1.1.2 Das Sicherheitskonzept

Der SQL-Server besitzt ein umfassendes Sicherheitskonzept und bietet drei unterschiedliche Möglichkeiten für die Login-Sicherheit an:

- *Integrierte Sicherheit*
  Nur eine Anmeldung am Betriebssystem ist erforderlich.
- *Standard-Sicherheit*
  Es ist eine Anmeldung am Betriebssystem und eine Anmeldung am DBMS erforderlich.
- *Gemischte Sicherheit*
  Je nach Verbindungsprotokoll, wird entweder die integrierte oder die Standard-Sicherheit gewählt.

Die SQL-Server-Logins werden bei der integrierten Sicherheit durch die Windows NT-Login-Sicherheit überprüft. Dagegen testet der SQL-Server, bei der Standard-Sicherheit selbst die SQL-Server-Logins, da sich der Nutzer zusätzlich zum Betriebssystem, beim DBMS zu authorisieren hat.
Der Vorteil der integrierten Sicherheit gegenüber der Standard-Sicherheit, liegt in der einmaligen Authorisierung des Nutzers am Betriebssystem.

Der Nachteil besteht allerdings darin, daß die bequemere Ameldung auf Kosten, der sonst zweischichtigen Sicherheit geht. Die integrierte Sicherheit erfordert weiterhin eine vertraute Verbindung bei den Verbindungsprotokollen, die nur bei Named-Pipes und Multi-Protokoll möglich sind. Dagegen ist die Standard-Sicherheit bei allen Protokollen möglich.
Die gemischte Sicherheit erlaubt es, in Abhängigkeit des Verbindungsprotokolls, zwischen Back-End und Front-End, die integrierte oder die Standard-Sicherheit zu wählen. Implizit wird vom System die integrierte Sicherheit gewählt, wenn die Verbindungsprotokolle eine vertraute Verbindung zulassen.

Für das PDAZN wurde die integrierte Sicherheit gewählt, die wesentlich komfortabler für den Anwender ist. Das bedeutet, der Anwender des PDAZN soll sich nach der Authorisierung am Betriebssystem nicht auch noch explizit beim DBMS anmelden müssen.

An dieser Stelle sollen die Zusammenhänge zwischen Login-ID, Benutzernamen und Benutzergruppen näher erläutert werden.
Zunächst gibt es eine spezielle Login-ID "SA". Diese Login-ID repräsentiert den Systemadministrator. Wer das Paßwort von "SA " kennt, hat demzufolge alle Rechte des Systemadministrators. Diese Möglichkeit hat ein Nutzer allerdings nur, wenn die Standard-Sicherheit aktiviert ist, da nur dann nach einer Login-ID und dem Paßwort gefragt wird.
Außer der Login-ID "SA" werden noch weitere Login-ID's bei der Installation des Systems angelegt. Dies sind "probe", "repl_publisher" und "repl_subscriber". Alle weiteren Login-ID's vergibt nur der Systemadministrator. Noch haben die Nutzer keinen Zugriff auf bestehende Datenbanken. Dazu muß der Systemadministrator erst den Login-ID's entsprechende Benutzernamen für jede Datenbank zuordnen. Die Login-ID's tragen im Regelfall den gleichen Namen wie die Benutzernamen.
Für "SA" gibt es keinen Benutzernamen, da der SQL-Server keine Berechtigungsüberprüfung für dieses Login durchführt.
In jeder Datenbank können Gruppen angelegt werden. Es gibt beim SQL-Server allerdings folgende Einschränkung:

Ein Benutzername darf nur einer Gruppe einer Datenbank zugeordnet werden.

Aufgrund dieser Einschränkung wurde eine spezielle Syntax für die Gruppenbezeichnung entwickelt, um einem Gruppennamen eine bestimmte Semantik zuzuordnen. Das Datensicherheitskonzept, in dem die Gruppennamen eine wichtige Bedeutung haben, wird in Abschnitt 4.2.2.2 erläutert.
Den Gruppen und Benutzern können bestimmte Rechte vergeben oder entzogen werden. Dabei werden zwei Arten von Berechtigungen unterschieden:

- *Objektberechtigungen*
  Diese Art der Berechtigung wird für Objekte vergeben. Als Objekte einer Datenbank werden beim SQL-Server Tabellen, gespeicherte Prozeduren und Sichten bezeichnet.

- *Anweisungsberechtigungen*
  Die Anweisungsberechtigungen sind nicht objektgebunden. Die Berechtigungen können nur durch den Systemadministrator oder den Datenbankeigentümer erteilt werden. Zu den Anweisungen gehören zum Beispiel *Create Procedure*, *Create Table*, *Dump Database*.

3.1.1.3 Verwalten des SQL-Servers mit DMO

Für den SQL-Server gibt es ein sogenanntes *Distributed Management Object Model*, kurz DMO-Modell, das in Abbildung 3.1 dargestellt wird. Durch Verwendung von Objektinstanzen dieses Modells kann ein SQL-Server *remote* administriert werden. Durch die Methoden und Eigenschaften der Objekte ist es zum Beispiel möglich, den Status des SQL-Servers zu setzen und den Namen des aktuellen Datenbankbenutzers festzustellen. Der Enterprise Manager, ein Tool zur Administrierung des SQL-Servers, wurde unter Zuhilfenahme des DMO-Modells implementiert. Weitere Informationen sind in der Online-Dokumentation und in den Handbüchern des SQL-Servers zu finden.

Im PDAZN wird mittels DMO der SQL-Server-Status und der Name des aktuellen Datenbankbenutzers ermittelt. Die Nutzung des DMO-Modells wird in Abschnitt 4.3.2.1 implementationstechnisch gezeigt.

3.1.1.4 Stored Procedures

Das Konzept der Stored Procedures ist bedeutend für die Performance und für die Sicherung der Datenintegrität. Darüberhinaus ermöglichen sie die Überprüfung von Geschäftsregeln (business rules) bzw. können einen Teil der Applikations-Logik umsetzen. Auf dieser Grundlage können Client/Server-Architekturen, wie die im Abschnitt 2.1.3 beschriebene "Verteilte Applikations-Logik"realisiert werden.
Stored Procedures sind vorkompilierte und auf dem Server gespeicherte Prozeduren, die in Transact-SQL implementiert sind. Transact-SQL ist ein SQL-Dialekt, der um prozedurale Sprachkomponenten erweitert wurde, zu denen beispielsweise der BEGIN-Block, die IF- und die WHILE-Anweisung gehören. Die Prozeduren können eine oder mehrere SQL-Anweisungen enthalten und nehmen bei Bedarf Parameter entgegen. Grundsätzlich werden Stored Procedures in System- und Benutzerdefinierte Prozeduren unterschieden.
Der Einsatz der Stored Procedures ist aus folgenden Gründen sehr vorteilhaft:

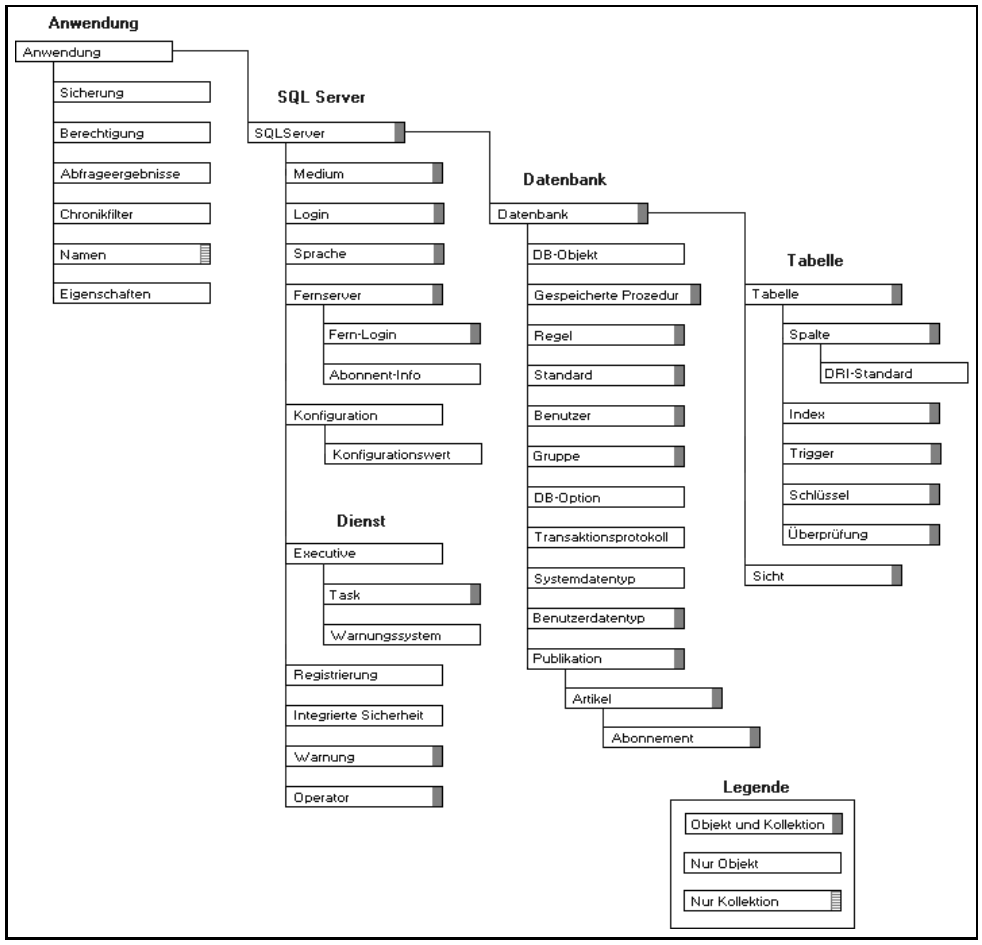


Abbildung 3.1: DMO-Modell des SQL-Servers

- Die Prozeduren werden vor ihrer Abspeicherung kompiliert. Bei weiteren Aufrufen ist somit keine Kompilierung notwendig, wodurch die Ausführung enorm beschleunigt wird.
  Die SQL-Anweisungen, die direkt an den SQL-Server gesendet werden, sind zeitaufwendiger in ihrer Bearbeitung, da sie zunächst kompiliert werden müssen. Aber auch die SQL-Anweisung selbst kann durch ihren Umfang eine zusätzliche Netzbelastung darstellen. Es erfolgt beim Aufruf von Stored Procedures nur eine geringe Netzbelastung, da nur der Prozedurname sowie die Parameter vom Client an den Server geschickt werden und der Server nur die Ergebnismenge an den Client zurückgibt.

- Die Ausführung der Stored Procedures erfolgt auf dem Server, der zumeist wesentlich schneller aufgrund seiner Hardwareeigenschaften ist. Dazu zählen der Prozessor, Bus, Hauptspeicher und weitere Komponenten.

Bis auf wenige Ausnahmen, sind fast alle Abfragen des PDAZN durch Stored Procedures implementiert, die durch das Front-End aufgerufen werden.

3.1.1.5 Installation

3.1.1.5.1 Hard- und Softwarevoraussetzungen

Die Installation gestaltete sich in Zusammenarbeit mit dem Verantwortlichen des DVSE sehr gut und unproblematisch.
Die SQL-Server Software gibt es in zwei Konfigurationsversionen: SQL-Server und SQL-Workstation. Die Workstations-Version ist mehr als eine Entwicklungsplattform zu betrachten. Der Hauptunterschied zur Server-Version liegt im wesentlichen in der Lizensierung und in der Anforderung an das Betriebssystem. Die Workstation-Version des SQL-Servers kann auch unter Windows NT Workstation installiert werden.
Wichtige Hard- und Softwaretechnische Voraussetzungen sind:

CPU:	Intels-basierte 80486 oder Pentium-Computer
	Alpha- oder MIPS-basierte Computer
	PowerPC
RAM:	mindestens 16MB (besser 32MB)
Betriebssystem:	Windows NT Server 3.51 für Server Version
	Windows NT Workstation 3.51 für Workstation Version

Weitere Daten können der Online-Dokumentation des SQL-Servers entnommen werden.

Um bestimmte Rechner als Client einzurichten, müssen bestimmte Client-Dienstprogramme des SQL-Servers auf diesen Rechnern installiert werden. Dazu gehört zum Beispiel der SQL Enterprise Manager. Für Client-Rechner werden neben Windows NT, auch die Betriebssysteme DOS und Windows 95 unterstützt. Aus Gründen der Sicherheit wird im PDZN nur Windows NT 4.0 Workstation als Client-Betriebssystem zugelassen. Es gab in der Zusammenarbeit von MS Access 97 und Windows 95 den Fehler der Art *Exception Violation*, der unter Windows NT 4.0 Workstation nicht auftrat. Vielleicht kann dieses Manko durch ein Servicepack von Access oder Windows 95 behoben werden. MS DOS fällt aus den Betrachtungen heraus, da auf jedem Client MS Access 97 als Frontend installiert werden muß.

Aus finanziellen Gründen wird der Datenbankserver im PDAZN auch als Front-End genutzt. Das ist bezüglich der Performance und Sicherheit des Servers nicht die optimale Lösung. Auf dem Datenbank-Server wurde zusätzlich zur Server-Software, noch Access 97 installiert. Auf dem Client wurden Windows NT Workstation, die SQL-Server-Dienstprogramme und Access 97 installiert.

3.1.1.5.2 Installationsoptionen

Bei der Installation sind wichtige Einstellungen bei den Installationsoptionen vorzunehmen, die sich notiert werden sollten, da sie zum Teil essentiell für das Recovery sind.
Neben dem Lizensiermodus und dem Installationspfad, der so belassen werden sollte, kann die Größe und der Speicherort des Mastermediums festgelegt werden.
Das Mastermedium speichert in verschiedenen Systemtabellen Informationen über die zur Verfügung stehenden Datenbank- und Sicherungsmedien, die auf dem Server gespeicherten Datenbanken, die Benutzerkonten, gespeicherten Systemprozeduren und weitere Informationen. Der Name des Mediums MASTER.DAT wurde nicht geändert. Die Größe des Mediums wurde zunächst mit 60 MB angegeben und kann später bei Bedarf problemlos über den SQL Enterprise Manager erweitert werden. Bei einem erforderlichen Recovery ist die Größe des Mediums einzutragen. Bei Angaben unter diesem Wert, ist kein erfolgreiches Recovery durchführbar.
Für das Recovery sind die Optionen Zeichensatz und Sortierreihenfolge ebenfalls wichtig. Beim Zeichensatz wurde "ISO 8859-1 (Latein 1 oder ANSI)"und bei der Sortierreihenfolge "mit Berücksichtigung von Groß- und Kleinschreibung"eingestellt. Diese Optionen können nur bei der Neuerstellung oder der Wiederherstellung des Mastermediums vorgenommen werden.

Eine andere wichtige Option ist die Auswahl der Netzwerkunterstützung. Hier sind eine oder mehrere Net-Libraries selektierbar, wobei für die integrierte Sicherheit nur *Named Pipes* oder *Multi-Protokoll* möglich sind. Beide wurden als Installationsangaben ausgewählt.

3.1.2 MS ACCESS 97

Als Front-End wird MS Access eingesetzt und soll deshalb unter diesem Aspekt in Abschnitt 3.1.2.1 vorgestellt werden. Das für Access zur Verfügung stehende Distributed Access Objects Model, kurz DAO-Modell, bildet die Grundlage für den Datenzugriff und wird in Abschnitt 3.1.2.2 beschrieben. Im letzten Abschnitt 3.1.2.3 werden Vor- und Nachteile diskutiert, die beim möglichen Upsizing von Access zum SQL-Server entstehen.

3.1.2.1 Einsatz als Frontend

Im PDAZN kommt MS Access 97 als Front-End zum Einsatz und bildet dadurch die Schnittstelle zum Nutzer.
MS Access 97 aus dem Office Professional Paket 97 bietet, wie auch der Vorgänger, eine vorbildliche Bedienoberfläche. Die Version 97 ist zum Beispiel bei den Controls, das sind Steuerelemente zur Interaktion mit dem Nutzer, um sogenannte Register erweitert worden,

wie sie in vielen anderen Optionseinstellungen vorkommen. Die Literatur gibt in [Bros97] einen umfassenden Überblick über die Unterschiede zwischen den Versionen 2.0, 95 (7.0) und Access 97.
Durch die Darstellungs- und Menüführungsmöglichkeiten von Access kann die Anwendung so entwickelt werden, daß der Nutzer auch nur die Daten erhält, die er wirklich anfordert. Das ist ein entscheidendes Kriterium für das Geschwindigkeitsverhalten der Anwendung in einer Client/Server-Umgebung.

Zwar wird Access auch als relationales DBMS bezeichnet, so erfüllt es aber dennoch nicht alle Anforderungen für ein vollständiges DBMS und wird daher auch als Pseudo-DBMS bezeichnet. In [HeSa95] sind weitere Unzulänglichkeiten beschrieben.
Eine Applikation mit besonders sensiblen Daten, wie das PDAZN, verlangt hohe Anforderungen an die Sicherheit. Bei vorangegangenen Access-Versionen, insbesondere Access 2.0, ist es Hackern mehrmals gelungen, das Sicherheitssystem von Access zu umgehen. Die Architektur des Access-Sicherheitssystems und deren Mängel sind in [ChGu+96] und in [Bros97] nachzulesen.
Zudem wird bei Aktivierung der Access-Sicherheit, die durch Vergabe eines Paßwortes für den Administrator erfolgt, nicht nur die Applikation für Nutzer, durch die Angabe von Login und Paßwort authorisierungspflichtig, sondern auch zugleich für andere Applikationen, die mit Access entwickelt wurden. Bei einer Run-Time-Version von Access stellt sich dieses Problem allerdings nicht dar. Diese ist in dem *Microsoft Access Developer's Toolkit* (ADT) enthalten, das allerdings separat zu erwerben ist und leider nicht für das PDAZN zur Verfügung stand. Weitere wichtige Tools sind ein *Setup-Wizard*, der die Applikation als Installationspaket vorbereiten kann und der *Help Workshop*, der die Erstellung der Applikations-Hilfe unterstützt.
Eine andere Möglichkeit des Datenschutzes besteht in der Vergabe eines einzigen Paßwortes für eine einzige Datenbank. Dies kann aber von jedem Nutzer geändert werden, wenn die Access-Sicherheit nicht aktiviert ist, da dann jeder Nutzer die Rechte eines Administrators hat. Außerdem entspricht die Vergabe eines Paßwortes an alle Anwender nicht den Sicherheitsanforderungen des PDAZN.

Die Authorisierung von Benutzern läßt sich unter Access nicht mit dem Betriebssystem Windows NT integrieren, um sich nur einmal am Betriebssystem authorisieren zu müssen.
Ebenfalls ist keine Integration der MS Access- mit der MS SQL-Server-Sicherheit möglich. Hier verspricht Microsoft in Zukunft Besserung.
Außer der notwendigen Sicherheit, fehlen Access außerdem wichtige Konzepte wie Stored Procedures, Trigger und eine umfassende Backup- und Recovery-Unterstützung.
MS Access ist primär eine Desktop-Anwendung mit hervorragenden Darstellungsmöglichkeiten und kommt aus deshalb im PDAZN als Front-End.

3.1.2.2 VBA und DAO

Alle Möglichkeiten von Access können nur durch Programmierung in Visual Basic for Application, kurz VBA, ausgeschöpft werden. Diese höhere Programmiersprache stellt einen Basic-Dialekt dar und erlaubt die Implementierung der Applikations-Logik bzw. der Geschäftsregeln, wodurch Client/Server-Architekturen des Typs "distributed logic", "data management" und "remote database" ermöglicht werden.
Mit VBA können zum Beispiel Ereignisse bearbeitet werden, die aufgrund von Benutzeraktivitäten eingetreten sind. Der Datenzugriff kann programmiertechnisch durch VBA unter Nutzung des Modells *Data Access Objects*, kurz DAO-Modell, erfolgen. Die Objektbibliothek DAO bildet mit den zur Verfügung gestellten Methoden und Eigenschaften die Schnittstelle zur eigenen Datenbank aber auch zu Datenbanken des SQL-Servers.
Die Jet-Engine ist das oberste Objekt dieser Hierarchie. Eine Ebene tiefer sind die Ob-

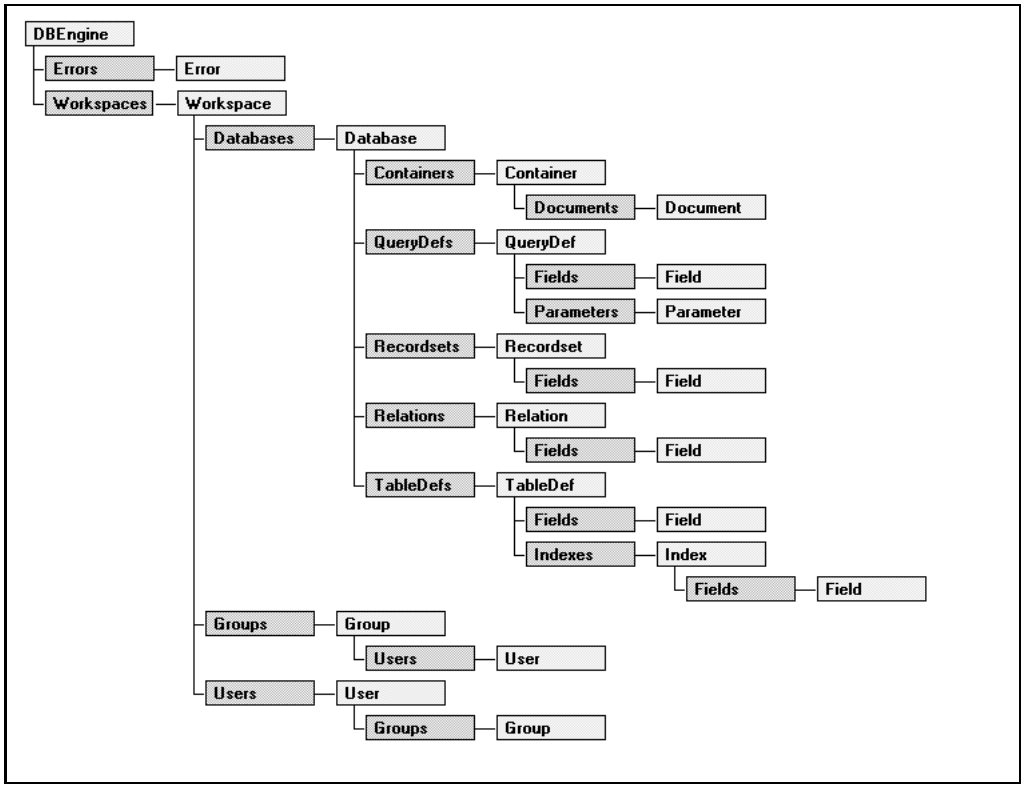


Abbildung 3.2: Data Access Objects von Access

jekte Error und Workspace angeordnet. Die Instanz eines Workspace-Objektes kann unter der Angabe spezieller Parameter erzeugt werden, die bestimmen, von welchem Typ diese Instanz ist. An dieser Stelle soll die Unterstützung von zwei verschiedenen Typen des Workspace-Objektes erwähnt werden. Das ist zum einen der Jet-Workspace und zum anderen der ODBCDirect-Workspace. Die Unterschiede bestehen in den Eigenschaften und den

zur Verfügung gestellten Methoden.
Der Jet-Workspace ermöglicht den Zugriff auf Daten in Microsoft Jet-Datenbanken, mit Microsoft Jet verbundene ODBC-Datenbanken und auf installierbare ISAM-Datenquellen in anderen Formaten, wie Paradox oder Lotus 1-2-3.
Dabei stellt ISAM ein Akronym für Indexed Sequentiell Access Method dar. Installierbare ISAM sind Treiber und werden geladen, wenn in der Anwendung darauf verwiesen wird.
Verbundene ODBC-Datenbanken stellen sich in MS Access durch "Verknüpfte Tabellen"des Back-End-Servers, wie dem MS SQL-Server dar. Die Jet-Engine, kann dadurch mit Daten aus den Tabellen des Back-Ends operieren. Dieser Remote-Zugriff ist aber nur bei kleinen Tabellen zu empfehlen, die max. 40-50 Einträge haben. Das entspricht nicht unbedingt einer strikten Client/Server-Architektur, ist aber in der Handhabung durchaus vertretbar.
Aufgrund der Jet-Engine können auch Replikate der Servertabellen auf Client-Rechner ausgelagert werden, die dadurch von der Jet-Engine selbst bearbeitet werden und eine enorme Beschleunigung des Zugriffs darstellen. Diese Möglichkeit ist bei großen Tabellen sehr effektiv, die relativ selten aktualisiert werden. Als Beispiel soll hier eine Tabelle mit Postleitzahlen genannt werden. Dazu ist ein Konzept zu entwickeln, zu welchen Zeitpunkten ein Update zwischen Server- und Client-Tabellen notwendig ist. Das kann ereignisgesteuert, zeitgesteuert oder aber auch per Befehl sofort erfolgen. Auf lokale Tabellen, also den Replikaten des Servers, ist ein Zugriff durch die Jet-Engine deshalb so vorteilhaft, da kein Zeitverlust durch den Datentransport über das Netzwerk auftritt. Die Client-Rechner sind zu dem auch recht leistungsfähige PC's, die solche Architekturen durchaus ermöglichen.

Ein ODBCDirect-Workspace ermöglicht den Zugriff auf einen Datenbank-Server mit Hilfe von ODBC, ohne das die Jet-Engine zur Kompilierung der Anfrage geladen wird. Dabei steht ODBC für Open Database Connectivity.
Beide Varianten werden für das PDAZN genutzt und sollten an dieser Stelle nur einen Ausblick darstellen, welches Potential dem Entwickler geboten wird.
Andere Objekte des DAO-Modells wie Querydef und Recordset werden an Beipielen in Abschnitt 4.3.2.1 erläutert.

3.1.2.3 Upsizing

Um Access-Datenbestände einer bestehenden Applikation auf einen SQL-Server zu transferieren, bietet Microsoft einen *Upsize Wizard* an, der kostenlos erhältlich ist.
Der Upsize Wizard kann aber auf keinen Fall aus einer Desktop-Anwendung eine Client/Server-Anwendung generieren, auch wenn das auf den ersten Blick so erscheinen mag.
Dazu kann der Wizard nicht in der Lage sein, da ihm keine Informationen über das Anwendungsumfeld, etc. zur Verfügung stehen.
Nach erfolgreichem Upsizing sind die ehemaligen Access-Tabellen als verknüpfte Tabellen sichtbar. Alle Abfragen, die in der Applikation auftreten, werden von der Jet-Engine kompiliert und entsprechend ihrer Optimierung auf dem Server, im positiven Fall, oder auf dem

Client, im negativen Fall ausgeführt. Alle anderen weiter oben beschriebenen Methoden des Zugriffs, zum Beispiel Pass-Through-Abfragen unter Nutzung der Stored Procedures, können beim Upsizing nicht angewandt werden.
Das Upsizing ist in [AlNi96] und [ChGu+96] ausführlich beschrieben. Abschließend wird darauf hingewiesen, daß ein Upsizing nur bei einer bereits bestehenden Applikation zu empfehlen ist, die in eine Client/Server-Applikation umgewandelt werden soll. Anschließend muß die Applikation an die Belange der Client/Server-Technik angepaßt werden, die mit sehr großem Aufwand verbunden ist.
Bei Neuentwicklungen von Applikationen in der Client/Server-Technik, sollte daher auf den Einsatz des Upsizing-Tools verzichtet werden.

3.1.2.4 Installation

MS Access 97 wurde zusammen mit dem Office-Professional-Paket 97 installiert und wird an dieser Stelle nicht weiter erläutert.
Besondere Einstellungen müssen für das PDAZN nicht vorgenommen werden. MS Access 97 kommt aus finanziellen Gründen, nicht nur als Front-End auf dem Client, sondern auch als Front-End auf dem Server zum Einsatz. Auf das Sicherheits-Risiko wird im Abschnitt 3.1.3.1 und auf den Performance-Verlust des Servers in Abschnitt 3.1.3.2 hingewiesen.

3.1.3 Windows NT 4.0

Aufgrund des eingesetzten DBMS MS SQL-Server 6.5, bestand die Forderung nach dem Betriebssystem Windows NT, da es eine Installationsvoraussetzung für den SQL-Server darstellt.

Das Betriebssystem Windows NT 4.0 ist ein sehr sicheres und zuverlässiges Produkt, das die gleiche Bedienoberfläche, wie Windows 95 bietet und deshalb für die Anwender keine großen Umstellungsprobleme bereitet.

Zwingend ist die Installation von Windows NT Server auf dem Rechner, auf dem auch der SQL-Server installiert wird. Als Client für den SQL-Server würde auch Windows 95 genügen, aber verschiedene Beispiele zeigten Schwächen in der Zusammenarbeit mit MS ACCESS 97. Daher wurde für das PDAZN die Installation von Windows NT Workstation auf den Client-Rechnern festgelegt.

Nachfolgend wird in Abschnitt 3.1.3.1, Windows NT unter dem Aspekt Sicherheit betrachtet. Die Domäne ZN, die für das PDAZN eingerichtet wurde, wird anschließend in Abschnitt 3.1.3.2 erläutert. Wichtige Bemerkungen zur Installation des Betriebssystem erfolgen in Ab-

schnitt 3.1.3.3, in dem insbesondere die sicherheitstechnischen Aspekte im Vordergrund stehen.

3.1.3.1 Die NT-Sicherheit

Der Sicherheit von Windows NT wurde bei der Entwicklung besondere Aufmerksamkeit geschenkt. Um in den USA überhaupt in kritischen Umgebungen als Betriebssystem eingesetzt werden zu dürfen, muß als Voraussetzung eine Zertifizierung durch das *National Computer Security Center* , kurz NCSC, erfolgen. Das NCSC ist wiederum eine Abteilung der *National Security Agency* (NSA).
Die Bewertung erfolgt nach dem sogenannten "Orange-Book", das eine Synonym für den Kriterienkatalog *Trusted Computer System Evaluation Criteria* (TCSEC) des Verteidigungsministeriums darstellt. In diesem Katalog existieren sieben Sicherheitsstufen: D, C1, C2, B1, B2, B3 und A. Dabei steht A für die höchste Sicherheitsstufe.
Der NT-Version 3.5 wurde unter Nutzung des Service Pack 3 und des NTFS-Dateisystems, das C2-Zertifikat ausgestellt, das ein recht hohes Sicherheitszeugnis darstellt.
Wesentliche C2-Kriterien sind folgende:

- Jeder Benutzer muß sich beim Login mit einem eindeutigen Benutzernamen und einem Paßwort identifizieren.
- Es muß die Möglichkeit der Überwachung aller sicherheitsrelevanten Aktionen bestehen, wobei Nutzer zur Rechenschaft gezogen werden können.
- Nur der Besitzer einer Ressource hat die vollständige Zugriffskontrolle.
- Auf gelöschte Objekte kann in keiner Art und Weise zugegriffen werden.
- Die Administratoren haben keine unbeschränkten Dateizugriffsrechte, können allerdings Besitzrechte übernehmen, die dann aber durch den ursprünglichen Besitzer erkennbar sind.

Die Hardware muß bei einer so hohen Zertifizierung selbstverständlich vor physikalischem Zugriff geschützt werden.

Im PDAZN ist aufgrund der hohen Datensensibilität, ein hohes Sicherheitsniveau des Betriebssystems unabdingbar, das durch Einsatz von Windows NT 4.0 erreicht wird. Dazu wurde das Service Pack 3 eingespielt und bei der Installation das NTFS-Dateisystem eingestellt.
Sicherheit bedeutet auch Datenschutz. Dafür bietet Windows NT die Möglichkeit an, Festplatten zu spiegeln. Fällt eine Festplatte aus, wird von der andern gebootet und das ganze System (inklusive Betriebssystem, DBMS) steht sofort wieder zur Verfügung. Da der Server

des PDAZN mit zwei Festplatten ausgestattet ist, wird diese Technik ebenfalls im PDAZN eingesetzt.
Der physikalische Schutz des Servers ist im PDAZN *nicht gewährleistet* . Auf diese Problematik wurde bei der Standortauswahl hingewiesen. Der Server müßte in einem verschlossenen Raum stehen, an dem kein Mitarbeiter mit normalen Rechten arbeiten darf. Die Gefahr durch unberechtigte Personen einen Schaden am Datenbestand zuzufügen oder einen unberechtigten Zugriff auf Patientendaten zu erlangen, ist sonst ungleich höher. Es besteht ein höheres Risiko bezüglich des Virenschutzes, da selbst Disketten ohne weiteres in den Server gelegt werden können. Damit besteht auch die Möglichkeit ein Betriebssystem einfach, durch ein anderes zu ersetzen, um dann an die Datenbestände des PDAZN zu gelangen.
Das Archiv ist im ZN ein Bereich, der nur unter Aufsicht einer zugangsberechtigten Person betreten werden darf. Diesbezüglich kommt die Problematik, des gewollten unberechtigten Zugriffs auf Patientendaten, nicht zum Tragen.

3.1.3.2 Die NT-Domäne ZN

Im PDAZN wurde die NT-Domäne "ZN" eingerichtet. Dazu wurde auf einem Rechner Windows NT Server installiert und der Name "ZNARCHIV" vergeben. Dieser Rechner übernimmt die Funktion des Primären Domain Controllers (PDC), des Datenbank-Servers und stellt zugleich auch ein Front-End dar.
Das ist für die Performance eine sehr ungünstige Konstellation. Sofern wenige Benutzer mit dem PDAZN arbeiten, wird diese Architektur nicht so schnell deutlich werden. Bei einer großen Domäne ist der PDC in seiner Funktion durch das Front-End und den Datenbank-Server eingeschränkt.
Bezüglich der Funktion als Datenbank-Server hat wiederum das Front-End und die Funktion als PDC negativen Einfluß auf die Performance.
Da der Rechner auch als Front-End benutzt wird, ist auch wiederum diese Nutzung, durch die anderen beiden Funktionen beeinträchtigt.
Günstiger ist die Installation des Datenbank-Servers auf einen zweiten Windows NT Server-Rechner innerhalb der Domäne "ZN". Der PDC und der Datenbank-Server dürfen in einer optimalen Architektur auch keine Front-Ends sein. Aus finanziellen Gründen bestand nicht die Möglichkeit einer anderen, besseren Architektur, da dafür mindestens ein weiterer Rechner notwendig ist.
Alle weiteren Rechner, die in Zukunft für die Nutzung des PDAZN installiert werden, sollten der Domäne "ZN" angehören. Die Nutzung von Rechnern anderer Domänen ist durch Einstellung der vertrauten Beziehung möglich.
Bezüglich der theoretischen Grundlagen des NT-Domänen-Modells, wird auf [Dapp+96] verwiesen.
Der Laserdrucker ist ebenfalls am PDC angeschlossen, wodurch der PDC nun auch die Aufgabe des Print-Servers hat. Sollte in Zukunft die Zahl, der für das PDAZN installierten Rechner, stark anwachsen und ist eine entsprechende Print-Nachfrage vorhanden, sollte ein anderer

Rechner, zum Beispiel ein Client im Archiv, die Aufgabe des Print-Servers übernehmen. Dadurch werden die Aufgaben besser verteilt und es stehen dem Datenbank-Server dadurch mehr Ressourcen zur Verfügung. Die Abbildung 3.3 zeigt die Komponenten der Domäne und welche Software momentan auf ihnen installiert wurde. Der zweite Client-Rechner existiert momentan nicht und dient nur zur Demonstration der Ausbaufähigkeit.

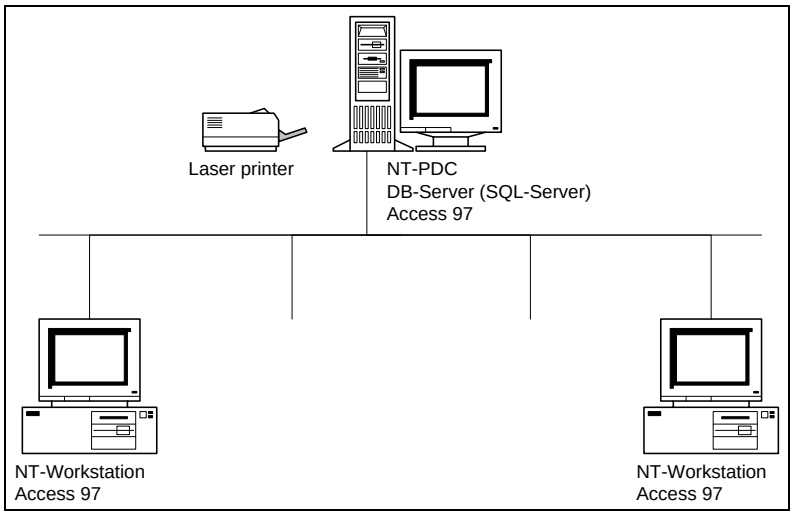


Abbildung 3.3: Architektur der NT-Domäne ZN

3.1.3.3 Die Installation

Für die Installation gibt es eine umfangreiche Literatur, wobei besonders [Dapp+96] hervorgehoben werden soll.
Die Diplomarbeit soll keine Installationsanleitung für Windows NT beinhalten, sondern es sollen nur wesentliche Aspekte, die auch aus Gründen der Sicherheit notwendig sind, erwähnt werden. Zunächst sind, wenn nicht schon vorhanden, drei Setupdisketten zu erstellen. Außerdem muß genügend freier Speicher auf der Festplatte vorhanden sein, um ein lokales Quellverzeichnis anzulegen. Die Setup-Disketten entfallen bei Windows NT 4.0, wenn der Rechner von einem CD-ROM-Laufwerk booten kann.
Es gibt zunächst einen textbasierten Teil der Installation, bei dem ein Erkennungsprozeß gestartet wird. Dabei geht es um Graphik- und Netzadapter , sowie Festplatten etc.. Beim textbasierten Teil ist die Auswahl des NTFS-Dateisystems aus weiter oben beschriebenen Gründen unverzichtbar. Im nachfolgenden graphischen Setup-Teil ist bei der Server-Version der Server-Typ anzugeben. Dabei erfolgt die Auswahl als PDC.
Alle weiteren Installationsschritte sollen im Rahmen dieser Arbeit nicht weiter diskutiert werden. Zudem stellt die Installation immer eine Aufgabe des System-Administrators dar.

Für das PDAZN wurde die Windows NT 4.0 Server- und die Workstation-Software in Zusammenarbeit mit dem Zuständigen des DVSE installiert.

3.1.4 Open Database Connectivity (ODBC)

Damit Applikationen wie Access oder Excel einen Zugriff auf SQL-Datenbanken, wie dem MS SQL-Server erhalten können, besteht eine Variante durch das sogenannte *call level interface*, kurz CLI. Dadurch existiert für den Zugriff auf unterschiedliche Datenbanken eine einheitliche Schnittstelle. Microsoft bietet dafür die ODBC-Schnittstelle an.
Microsoft ODBC basiert auf den Definitionen der SQL Access Group/X/Open und ist integraler Bestandteil der Microsoft *Windows Open Service Architectur* (WOSA) - Systemarchitektur. Wenn eine Applikation auf eine SQL-Datenbank zugreift, bestehen grundsätzlich zwei Probleme:

- Wird die SQL-Datenbank aus den verschiedensten Gründen ausgetauscht, muß die Applikation mit erheblichen Aufwand angepaßt werden.
- Noch schwieriger ist der Aufwand beim Zugriff einer Applikation auf unterschiedliche SQL-Datenbanken. DBMS.

Es lag nahe eine Schnittstelle zwischen Applikation und SQL-Datenbank (DBMS) zu entwickeln, die es erlaubt einheitliche Anfragen an die Schnittstelle zu übergeben, die aber selbst dafür sorgt, die Spezifik des DBMS zu berücksichtigen. Die Spezifik eines DBMS drückt sich vor allen Dingen in den unterschiedlichen SQL-Dialekten aus.
Das Ziel besteht darin, die Applikation nur über ODBC auf ein DBMS zugreifen zu lassen. Aus diesem Grund muß es Komponenten geben, die der Spezifik des DBMS Rechnung tragen. Das wird durch den speziellen *ODBC-Treiber* für ein DBMS realisiert, der ein Teil von ODBC darstellt. Der andere Teil ist der *Allgemeine ODBC Treiber Manager*, der die Schnittstelle für alle Applikationen darstellt. Wird beispielsweise in Access die bekannte Technik der verknüpften Tabellen verwendet, die in Abschnitt 3.1.5.1 erläutert werden, so liegt ODBC transparent dahinter.

3.1.4.1 Einrichten der Datenquelle

Damit Tabellen eingebunden werden können ist es notwendig, die entsprechende ODBC-Datenquelle einzurichten. Das geschieht außerhalb von Access unter dem Betriebssystem. Es ist für jeden Rechner, ob Client oder Server, auf dem das PDAZN installiert wird notwendig, eine ODBC-Datenquelle einzurichten.
Dazu wird unter Windows NT der *ODBC Data Source Administrator* gestartet, dessen Icon nach dem Öffnen der *Control Panels* erscheint.

Für das PDAZN wurde die System-Datenquelle, "systempdazn.dsn "eingerichet, indem der spezielle Treiber des MS SQL-Servers ausgewählt wurde. Eine System-DSN gilt für jeden Benutzer und ist nur für den konkreten Rechner gültig. Der ODBC Treiber *SQL Server*, wird durch die Installation des SQL-Servers zur Verfügung gestellt. Die Einrichtung der System-DSN ermöglicht weiterhin den Remote-Zugriff auf SQL-Server-Tabellen, ohne jedesmal die notwendige Verbindungszeichenfolge explizit anzugeben, da diese Informationen in der System-DSN abgespeichert sind. Dazu wird zum Beispiel der Server-Name, die Datenbank und die Sprache angegeben. Im Beispiel des PDAZN, im Archiv des ZN, ist das in der gleichen Reihenfolge: *znarchiv*, *pdazn*,*deutsch*. Auf anderen Plattformen können sich natürlich die Server-Namen ändern. Der Remote-Zugriff erfolgt zum Beispiel in dem Moment, indem eine verknüpfte Tabelle in MS Access geöffnet wird. Auch bei speziellen Abfragen, wie den SQL-Pass-Through-Abfragen, ist die Einrichtung der Datenquelle erforderlich, da sonst jedesmal eine erneute explizite Anmeldung am SQL-Server notwendig wäre.

3.1.5 Datenaustausch zwischen MS Access und MS SQL-Server

Nach einer kurzen Einführung von ODBC folgt eine Vorstellung mehrerer Methoden, die einen Zugriff von Access aus, auf die Daten anderer DBMS, wie dem MS SQL-Server ermöglichen. Unterschiede gibt es dabei in der Performance und im Aufwand der Entwicklung. Es gibt im wesentlichen fünf verschiedene Methoden, um Daten zu übermitteln:

- Verknüpfte Tabellen
- SQL-Pass-Through-Abfragen
- (OLE)-Automatisierung
- Open Database Connectivity Application Programming Interface (ODBC API)
- Remote Data Objects (RDO)-Abfragen

Die drei erstgenannten Methoden kommen auch im PDAZN zum Einsatz. Die beiden anderen Methoden wurden aus Gründen der Verfügbarkeit (RDO) und der Notwendigkeit (ODBC API) nicht eingesetzt.
Grundsätzlich erfolgt der Datenaustausch immer über ODBC. Die einzige Ausnahme bildet die OLE-Automatisierung. Bei den Methoden unterscheidet sich sehr stark, inwieweit einzelne Komponenten an der Übersetzung der Anweisungen oder dem Verbindungsaufbau beteiligt sind. Die Abbildung 3.4 basiert auf einer Darstellung von [ChGu+96], die schematisch darstellt, welche Komponenten am Verbindungsaufbau bzw. an der Übermittlung Daten beteiligt sind. Bei der Pass-Through-Abfrage, wurden die Verbindungen zur Jet-Engine, dem ODBC-Manager und dem ODBC-Treiber nur durch schwache Linien gekennzeichnet, da sie

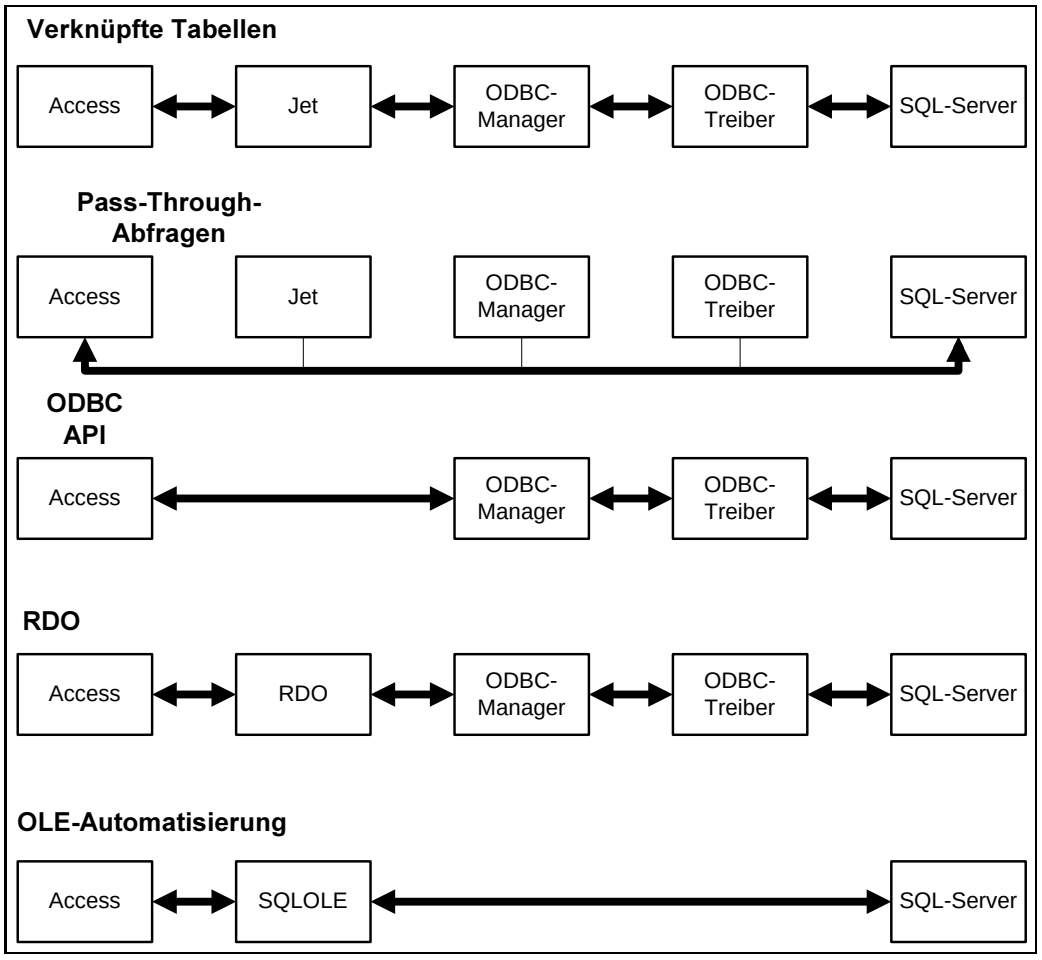


Abbildung 3.4: Methoden für den Datenaustausch zwischen Access und SQL-Server

zwar am Verbindungsaufbau beteiligt, aber nicht an der Umformung der Abfrage beteiligt sind.
Manche Komponenten werden für den Verbindungsaufbau nicht benötigt und werden deshalb bei der entsprechenden Methode auch nicht dargestellt, auch wenn diese Komponenten dennoch existieren, nur nicht in diesem Kontext. Ein Beispiel ist hierfür die Jet-Engine bei RDO.
Die genannten Methoden werden in den folgenden Abschnitten kurz vorgestellt.

3.1.5.1 Verknüpfte Tabellen

Access bietet die Möglichkeit auf Tabellen anderer DBMS zuzugreifen, in dem Server-Tabellen mit Access verknüpft werden. Die Jet-Engine operiert nach außen hin mit den Daten der Tabellen so, als wären es lokale Tabellen. Dennoch überwacht der SQL-Server die Integrität

der Daten und lehnt bei deren Verletzung die versuchten Änderungen der Jet-Engine mit einer Fehlermeldung ab.
Die Änderung der Tabellenschemata ist durch Access nicht möglich. So können keine Validierungsregeln, keine Fehlermeldung bei deren Verletzung, keine Standardwerte und weitere mehr angegeben werden. Diese Änderungen unterliegen der vollständigen Kontrolle durch den SQL-Server.
Die Jet-Engine kompiliert bei dieser Methode die SQL-Anweisung, wobei Jet-Engine-Anfrage-Optimierer versucht, den größten Teil der Abfrage auf dem SQL-Server auszuführen. Probleme bereiten hier nutzer- und Access-spezifische Funktionen, die der SQL-Server nicht kennen kann. In diesem Fall müssen die Daten über das Netz zum Client transportiert werden, der dann die Auswertung übernimmt. Diese ungünstige Situation tritt bei der Verwendung von heterogenen Verknüpfungen auf, die dann entstehen, wenn zum Beispiel eine lokale Tabelle mit einer Servertabelle verknüpft wird.
Der Entwickler hat bei verknüpften Tabellen sehr streng auf diese Thematik zu achten, da es sonst zu einem erheblichen Performance-Einbruch kommen kann. Dieser wird aber leider erst bei einem bestimmten Datenbestand deutlich, worauf in der Testphase geachtet werden muß.
Bezüglich der Geschwindigkeit entsteht bei der Verwendung von verknüpften Tabellen ein Overhead. Verursacht wird dieser, durch den Aufruf der Jet-Engine mit ihrem Abfrageoptimierer. Unter bestimmten Umständen gleicht Access diesen Overhead durch gute Optimierung wieder aus.
ODBC wird bei der Datenermittlung vom SQL-Server implizit verwendet. Dazu übersetzt die Jet-Engine das spezielle Access-SQL in den allgemeinen ODBC SQL-Dialekt und sendet es dann an den Treiber-Manager. Die Verbindung wird durch den Treibermanager hergestellt und das allgemeinere SQL, an den speziellen ODBC-Treiber übergeben. Dieser übersetzt es schließlich in den korrekten SQL-Dialekt des SQL-Servers.

Im PDAZN kommt diese Methode bei sehr kleinen Tabellen zum Einsatz, die nicht mehr als 50 Datensätze beinhalten. Selbst wenn diese Tabellen bei Abfragen durch den Abfrageoptimierer der Jet-Engine vom Server zum Client transportiert werden, ist eine ausreichende Geschwindigkeit gegeben. Bei großen Tabellen wäre diese Methode nicht vertretbar.

3.1.5.2 SQL-Pass-Through-Anfragen

Diese Methode eröffnet dem Entwickler die Möglichkeit, die Features des SQL-Servers zu nutzen. Durch SQL-Pass-Through-Anfragen ist es möglich Stored Procedures aufzurufen, die ein mächtiges Mittel zur Steigerung der Zugriffsgeschwindigkeit darstellen.
Die Jet-Engine kompiliert die Abfrage nicht, sondern schickt sie direkt zum SQL-Server. Sie wird somit immer auf dem SQL-Server ausgeführt. Demnach können auch fehlerhafte Anweisungen zum SQL-Server geschickt werden, die er dann mit der entsprechenden Fehler-

meldung quittiert. Bei der Entwicklung von Anweisungen dieser Methode bietet es sich an, die SQL-Anweisungen zunächst direkt auf dem Server, unter Nutzung seiner Abfrage-Tools zu testen.
Der Overhead ist im Vergleich zu den verknüpften Tabellen gering, da die Jet-Engine nur für die ODBC-Verbindung verantwortlich ist. Die ODBC-Komponenten reichen eine Abfrage weiter, ohne sie zu weiter zu beeinflussen.

Die Verwendung von Stored Procedures auf der Grundlage von Pass-Through-Abfragen spielt im PDAZN eine sehr große Rolle, auf deren Grundlage alle wesentlichen Abfragen erfolgen. Die Vorteile wurden in dem Abschnitt 3.1.1.4 bereits behandelt.

3.1.5.3 (OLE)-Automatisierung

In diesem Abschnitt steht die Automatisierung im Mittelpunkt. Früher wurde sie vollständig als OLE-Automatisierung bezeichnet. Dabei steht OLE für Object Linking and Embedding. Diese Automatisierung ist ein Funktionsmerkmal des COM (Component Object Model), einer als Industriestandard anerkannten Technologie, mit dem Anwendungen ihre Objekte für Entwicklungsprogramme, Makrosprachen und andere Anwendungen, die Automatisierung unterstützen, offenlegen.
MS Access wird in diesem Kontext als sogenannte ActiveX-Komponente bezeichnet, da es die Automatisierung unterstützt. Früher wurde die ActiveX Komponente, je nach Funktion, als OLE-Automatisierungs-Server oder als OLE-Automatisierungs-Controller bezeichnet. Access ist der Automatisierungs-Server, wenn es Objekte offenlegt und Automatisierungs-Controller bei der Nutzung von Objekten anderer Anwendungen.
Beim SQL-Server wurde das hierarchische DMO-Modell erwähnt. Dessen Objekte werden auf diese Art und Weise durch Access oder anderen Applikationen, die Automatisierung unterstützen, nutzbar. Access ist in diesem Fall der OLE-Automatisierungs-Controller. Zu diesem Zweck muß die SQLOLE-Bibliothek zur Verfügung stehen. Es wird in der Online-Dokumentation ausdrücklich darauf hingewiesen, DMO primär zur Administration und weniger für die reine Datenermittlung zu verwenden.
Die OLE-Automatisierung wird im PDAZN für das Datensicherheitskonzept verwendet, um den Anwender des PDAZN festzustellen.

3.1.5.4 ODBC API

Eine besonders schnelle, aber leider auch aufwendige Methode stellt die Möglichkeit dar, direkt die Schnittstelle zu ODBC zu nutzen.
Dafür steht das *ODBC Application Programming Interface*, kurz ODBC API, zur Verfügung, das eine Vielzahl von Funktionen bereithält.

Die Verwendung dieser Methode hat den geringsten Overhead, da die Anfrage nicht durch die Jet-Engine umgeformt wird und sie auch nicht die Verbindung zum Treiber-Manager herstellen muß. Die Verbindungsinformationen und die Abfragen müssen bei ODBC API direkt implementiert werden, um sie direkt zum Treiber-Manager zu senden.
Die Aufgabe des Verbindungsaufbaus wird für verknüpfte Tabellen und Pass-Through-Abfragen von der Jet-Engine übernommen.

Das PDAZN kommt in der ersten Version ohne ODBC API aus, da die Geschwindigkeit zufriedenstellend ist. Sollte die Applikation später um bestimmte Abfragen erweitert werden, deren Datenermittlung mit den anderen Methoden zu viel Zeit in Anspruch nimmt, kann ODBC API eingesetzt werden.
Ein ausführlicher Einstieg in ODBC API wird durch [ChGu+96] geboten.

3.1.5.5 RDO

Eine andere Methode auf Daten des SQL-Servers zuzugreifen, besteht in der Verwendung der *Remote Data Objects*, kurz RDO. Diese Methode kann allerdings nur mit der Enterprise Edition von Visual Basic genutzt werden. Dadurch stand diese Methode nicht für das PDAZN zur Verfügung.

Für RDO existiert eine Objekthierarchie, wobei die Eigenschaften und Methoden dieser Objekte dazu verwendet werden, eine Abfrage zu erstellen. Diese Abfrage wird in den Code für ODBC API übersetzt und anschließend zum Treiber-Manager gesendet.
Die Verwendung von RDO ist schneller als die Verwendung der Jet-Engine, aber durch die Übersetzung in ODBC API etwas langsamer als die direkte Verwendung von ODBC API.

Sollte sich bei einer Anwendung der Datenzugriff über andere Methoden, außer ODBC API, als zu langsam erweisen und die Enterprise Edition von Visual Basic steht zur Verfügung, sollte RDO bevorzugt gegenüber ODBC API eingesetzt werden, da der Programmieraufwand für RDO immer noch geringer ist, als direkt den Code für ODBC API zu schreiben.
Der Datenzugriff über RDO, wird in [ChGu+96] erläutert.

3.2 Die installierte Hardware

In Absprache mit der Medizinischen Fakultät und der Prüfung durch das DVSE, wurden folgende Komponenten für das PDAZN zur Verfügung gestellt:

- ein Rechner mit der hardwaretechnischen Ausstattung eines Servers (Pentium Pro 200, 64 MB RAM, 2 HDD's, USV, DAT-Streamer usw.)

- ein Rechner mit einer Ausstattung als Client (Pentium MMX 166, 32 MB, usw.)
- Laserdrucker

Selbstverständlich war ein gewisser finanzieller Rahmen zu beachten. Mit einem größeren Budget hätte die Ausstattung des Client-Rechners verbessert werden können, um ihn in der Domäne auch mit der Server-Software von Windows NT auszustatten. Dann wäre dieser Rechner als Datenbankserver zum Einsatz gekommen. Die Gründe wurden weiter oben bereits in Abschnitt 3.1.3.2 beschrieben.

Die Rechner wurden an das Subnetz der Medizinischen Fakultät angeschlossen, das auf der *Twisted-Pair* Technologie basiert. Die Rechner sind dadurch technisch durch HUB's untereinander verbunden.

Der Laserdrucker wurde an den Server angeschlossen, der dadurch zugleich auch der Print-Server ist, um anderen Rechnern den Zugang zu dieser Ressource zu ermöglichen. Der Client hätte aus hardwaretechnischer Sicht keinerlei Probleme, um ebenfalls als Print-Server zu fungieren.

Der Standort der Hardware ist das Archiv des ZN. Der Archivverantwortliche wurde auf die Problematik des Server-Standortes hingewiesen, denn der Server darf aus sicherheitstechnischen Gründen nicht in diesem Archiv stehen und schon gar nicht durch Mitarbeiter genutzt werden.
Auf speziellem Wunsch der Medizinischen Fakultät wurde er dennoch dort installiert, da er auch als Front-End genutzt werden soll. Auf die Nachteile dieses Einsatzes wurde bereits in den Abschnitten 3.1.3.1 und 3.1.3.2 hingewiesen.

Kapitel 4

Entwicklung des PDAZN

Der Inhalt dieses Kapitels stellt die Entwicklung des PDAZN im Rahmen des Softwarelebenszyklusses dar und ist deshalb ähnlich diesem Zyklus strukturiert. Wichtig dabei ist nicht nur das Ergebnis, sondern auch, wie der Prozeß selbst verlaufen ist. Die Analyse der Hard- und Softwareanforderungen wird nicht in diesem Abschnitt diskutiert, da diese Aufgabe bereits in Kapitel 3 behandelt wurde. Dieser Abschnitt befaßt sich mit der Entwicklung der Applikation PDAZN.
Zu Beginn erfolgt in Abschnitt 4.1 eine Beschreibung der Analyse. In Abschnitt 4.2 wird eine Beschreibung des Entwurfs vorgenommen, der sich in den Datenbankentwurf und den Programmentwurf untergliedert. In Abschnitt 2.2 wurde bereits die Architektur des PDAZN kurz beschrieben, aber nicht erläutert, wie diese Architektur mit den zur Verfügung stehenden Mitteln und Methoden entworfen wurde. Anschließend wird in Abschnitt 4.3 an Beispielen gezeigt, wie die entworfenen Programmstrukturen umgesetzt wurden. Danach erfolgen Angaben zum Test. Schließlich stellt der letzte Abschnitt 4.5, die wichtigsten Aspekte bezüglich der Wartung des PDAZN dar.

4.1 Analyse

Die Analyse ist als ein Prozeß zu betrachten, der sich nicht geschlossen realisieren ließ. Das hängt vor allen Dingen mit der Komplexität der Aufgabenstellung zusammen. Insbesondere konnte die funktionale Analyse erst durch die Darstellung von Prototypen vorgenommen werden. Dazu wurden anhand der Prototypen in mehreren Präsentationen, die Möglichkeiten mit den Anwendern diskutiert. Dadurch erführ der Anwender, welche Mittel die zukünftige Applikation zur Verfügung stellen kann. Dazu zählen spezifische Darstellungsmöglichkeiten des Front-Ends, wie Listenfelder, Kombinationsfelder, Optionsfelder, Kontrollfelder, Umschaltflächen, etc..
Bevor die für die Archivierung benötigten Daten näher erläutert werden, wird zunächst definiert, was im Sinne des Archivs und damit des PDAZN, unter einer Patientenaufnahme zu

verstehen ist.

Definition 4.1 (Patientenaufnahme) *Eine Patientenaufnahme ist ein Vorgang, bei dem ein Patient stationär in eine Klinik aufgenommen wird und innerhalb der Klinik auf andere Stationen verlegt werden kann. Mit der Entlassung aus dieser Klinik endet auch die Patientenaufnahme.*

Die Notwendigkeit dieser Festlegung ergibt sich deshalb, da eine Patientenaufnahme im Sinne des DVSE nicht abgeschlossen wird, wenn der Patient auf eine andere Klinik innerhalb des Klinikums verlegt wird. Diese kostenorientierte Sichtweise ist für das PDAZN jedoch unbrauchbar, denn aus Archivsicht müssen die Patientaufnahmen klinikabhängig gespeichert werden.
Ein kleines Beispiel soll deshalb vorab zeigen, welche Folgen eine Mißachtung dieser Sichtweisen haben kann: Wird ein Patient von einer Klinik des Klinikums in eine andere verlegt, wird das in der Terminologie des DVSE als Patientenverlegung bezeichnet. Im Sinne des Archivs bedeutet es eine Patientenentlassung der einen und eine Patientenaufnahme der anderen Klinik.
Diese unterschiedlichen Bedeutungen kommen besonders im Abschnitt 4.3 zum Tragen, wo die Analyse des Datenimports vom DVSE beschrieben wird.
Im Abschnitt 4.1.1 steht die archivspezifische Analyse im Vordergrund, während in Abschnitt 4.1.2 die klinikspezifische Analyse erläutert wird.

4.1.1 Archivspezifische Analyse

Die erste Aufgabe bestand darin, den Informationsaustausch des Archivs mit den anderen Einrichtungen zu analysieren, um die notwendigen Daten und Aufgaben abzuleiten. Insbesondere mußte eine Einarbeitung in das Vokabular des Anwenders erfolgen. Sofern es möglich war, wurde in dieser Arbeit auf spezielles medizinisches Vokabular verzichtet, um die Verständlichkeit nicht zu sehr durch Branchenwissen zu beeinträchtigen. Falls dennoch die Notwendigkeit bestand, erfolgt eine Erläuterung an entsprechender Stelle.
Nach der Datenanalyse in Abschnitt 4.1.1.1 erfolgt die Datenklassifikation in Abschnitt 4.1.1.2. Im letzten Abschnitt 4.1.1.3 werden die Funktionen erläutert, die aus der Aufgabenanalyse abgeleitet wurden.

4.1.1.1 Aufgaben des Archivs und der Informationsaustausch

Der analysierte Informationsfluß im ZN wird durch die Abbildung 4.1 dargestellt. Der einzige weiße Pfeil in der Abbildung stellt den Informationsfluß über das lokale Netz der Medizinischen Fakultät dar. Die grau unterlegten Pfeile repräsentieren den Informationsaustausch über konventionelle Listen etc..

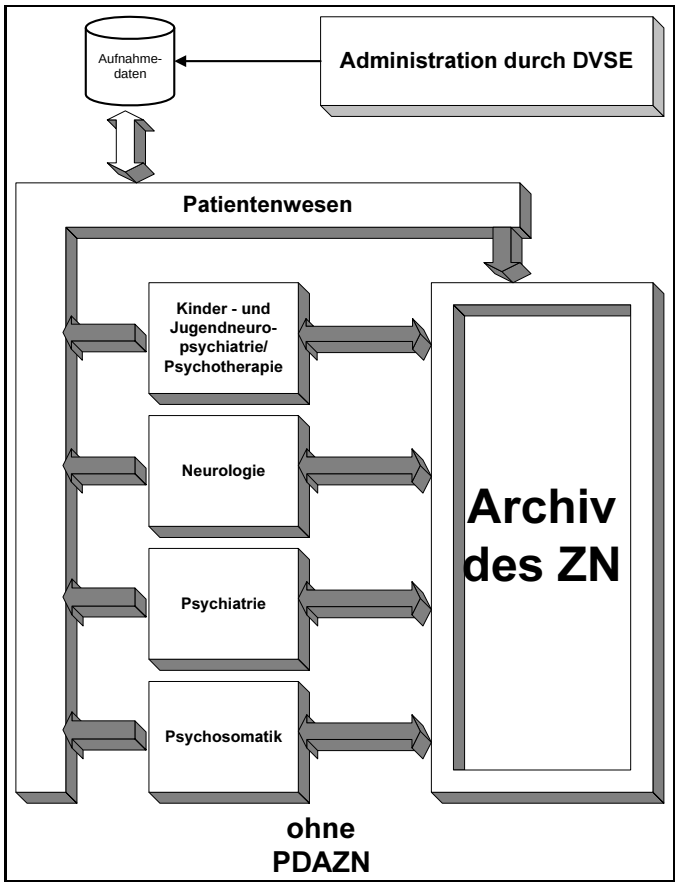


Abbildung 4.1: Informationsfluß mit dem Archiv vor Einführung des PDAZN

Das Patientenwesen umspannt aus informationstechnischer Sicht das ZN, da es die Daten der Patientenaufnahmen entgegennimmt und gleichzeitig als Informationsquelle für das Archiv dient.
Um den Informationsfluß zu verstehen, soll zunächst ausführlich erläutert werden, wie aus informationstechnischer Sicht eine stationäre Patientenaufnahme durchgeführt wird. Der abrechnungstechnische Aspekt soll dabei nicht im Vordergrund stehen, sondern vielmehr die Versorgung der Einrichtungen mit Informationen, die für die Behandlungen der Patienten notwendig sind.
Bei einer Patientenaufnahme, stellt sich der Patient selbst beim Patientenwesen vor oder falls dies nicht möglich ist, werden die Daten durch Klinikangestellte übermittelt. Die Informationen werden demzufolge nicht auf elektronischem Wege von den Kliniken weitergeleitet. Das Archiv erhält anschließend Informationen über neue Patientenaufnahmen in Form von gedruckten Aufnahmelisten. Diese Listen enthalten Informationen wie Name, Vorname, Geburtsdatum, Aufnahmedatum, Aufnahmeuhrzeit, Stationsname, etc.. Das Archiv trägt diese Informationen in Aufnahmebücher ein, die auch als Index verstanden werden können, um

später schnell Auskünfte über Patientenaufnahmen geben zu können.
Damit kann das Archiv bereits Auskünfte über Patientenaufnahmen geben, denen noch kein Entlassungsdatum zugewiesen wurde.
Die Kliniken fordern gleichzeitig bei der Aufnahme eines Patienten eine Akte vom Archiv an, die sie bei entsprechender Existenz erhalten. Findet das Archiv keine Akte, geht die Meldung zurück zur Klinik und diese fordert eine neue leere Akte vom Patientenwesen an.
Solange der Klinik die Patientenakte nicht vorliegt, hat sie auch keine Informationen über eventuell schon vorhandene Krankheiten, die unter bestimmten Umständen sehr wichtig sein können. Es gibt dabei Ausnahmefälle, daß bestimmte Kliniken, wie die Psychosomatik, schon über eigene Informationssysteme verfügen und damit auch lokale Daten über den Patienten vorliegen, sofern er schon einmal durch die Klinik aufgenommen wurde. Die lokalen Informationssysteme können selbstverständlich keine Informationen über Patientenaufnahmen anderer Kliniken des ZN besitzen.
Nach der Entlassung des Patienten erhält das Patientenwesen von den Kliniken die Entlassungsinformationen und gibt an das Archiv wieder eine Liste von Informationen, die dann zu den bereits bestehenden hinzugefügt werden können. Dazu gehören das Entlassungsdatum, der Kostenträger etc..
Nach der Entlassung des Patienten wird von der Klinik der sogenannte Arztbrief angefertigt. Durch Fertigstellung des Arztbriefes, ist die Epikrise[1] als durchgeführt zu betrachten.
Bekommt das Archiv die Patientenakte zurück, wird ein Auszug (*Abstract*) des Arztbriefes, in die Aufnahmebücher eingetragen. Er enthält detailliertere Informationen zu den gestellten Diagnosen und erhöht dadurch den Informationsgehalt der Bücher deutlich. Damit ist informationstechnisch die Aufnahme abgeschlossen.

Im folgenden werden die zwei wesentlichen Aufgabengebiete noch einmal zusammengefaßt:

- Archivierung von stationären Patientaufnahmen aller Kliniken des ZN
- Schnelle Bearbeitung von Klinikanfragen

Die zu entwickelnde Applikation muß deshalb alle notwendigen Informationen speichern, die eine Archivierung der Akten in dem Sinne unterstützt, daß alle bisher geführten Aufnahmebücher ersetzt und solche Informationen aus der Akte hinzugenommen werden, die eine schnelle Informationsauskunft über die wichtigsten Daten ermöglichen. Weiterhin muß das PDAZN die häufigsten Anfragen durch geeignete Mittel unterstützen.

In vielen Analysegesprächen mit den Mitarbeiterinnen des Archivs, wurden Daten spezifiziert, die zur Erfüllung der genannten Aufgaben notwendig sind.
Das Archiv pflegt auch ein Sterbebuch, wobei der Wunsch seitens des Archivs bestand, dieses Buch mit in das PDAZN aufzunehmen.

[1] Epikrise - abschließende ärztliche Beurteilung

Aufgrund der Semantik der analysierten Informationen, wurden diese in Informationsregister eingeteilt, die aus einer Menge von Attributen bestehen.
Für die Archivarbeit ergaben sich folgende drei Informationsregister, wobei an dieser Stelle nicht alle Attribute aufgezählt werden sollen, sondern nur charakteristische Beispiele:

- *Allgemeine Daten*
  Dazu gehören persönliche Daten wie Name, Vorname, Geburtsdatum, Geburtsname, Geschlecht, Adresse, Familienstand, Religion, Nationalität etc.. Weiterhin werden alle Stationsverlegungen dokumentiert, die innerhalb dieser Patientenaufnahme stattfanden. Dazu werden neben dem Aufnahmedatum, der Aufnahmeuhrzeit, Entlassungsdatum, Entlassungsuhrzeit und dem Stationsnamen, auch Aufnahmemodus und Entlassungsmodus gespeichert.
  Andere Informationen sind die Anzahl der Aufnahmen eines Patienten für jede Klinik, die nicht elektronisch gespeichert wurden, sowie spezielle Patientennummern anderer klinikinterner Informationssysteme, die in der Psychosomatik und in der Kinder- und Jugendklinik zum Einsatz kommen.
  Zum Register Allgemeine Daten gehört auch die Aufnahmenummer. Es ist aber keine abstrakte Identität einer Patientenaufnahme des PDAZN, sondern die Aufnahmenummer, die durch eine andere Applikation der Medizinischen Fakultät vergeben wird, nachdem das Patientenwesen die neue Aufnahme elektronisch erfaßt hat. Diese Vergabe erfolgt kostenorientiert. Wie bereits zu Beginn des Abschnittes Analyse beschrieben, wird eine Patientenaufnahme im Sinne des DVSE nicht beendet, wenn es zu einer Verlegung von einer Klinik in eine andere Klinik innerhalb des Klinikums kommt. Die Konsequenz ist, daß sich dadurch auch die Aufnahmenummer nicht ändert und deshalb für das PDAZN als Aufnahmenummer nicht verwendet werden kann.
  Eine weitere Information, die für die Archivarbeit von großem Nutzen ist, stellt die Information über die Durchführung der Epikrise dar.

- *Diagnosen*
  Die Diagnoseinformationen werden allen Kliniken durch ein einheitliches Schema präsentiert.
  Dazu diente als Vorlage ein Erfassungsformular der Psychosomatik, dem sich die Vertreter aller Kliniken anschließen konnten. Wichtige Angaben einer Diagnose sind der Rang, der Diagnoseschlüssel, das Diagnosesystem[2], Diagnosezeitpunkt und mögliche Zusätze (z.Z. erscheinungsfrei etc.).
  Weitere Angaben sind das bereits beschriebene Abstract zur Diagnose. Andere Angaben sind klinikspezifisch, wie zum Beispiel das Attribut "Diagnoseergänzender Symptomkatalog" bei der Kinder- und Jugendklinik.

[2]Berücksichigte Diagnosesysteme sind ICD-9, ICD-10, DSM-IIIR, DSM-V

- *Sterberegister*
  In dem Sterbebuch sind alle Patienten, die verstorben sind, mit dem Sterbedatum und dem Eintrag intern oder extern vermerkt, wobei "intern" für innerhalb des ZN und "extern" für außerhalb des ZN verstorben, steht.

Nach Feststellung der Aufgaben und der dazu notwendigen Daten, erfolgt im nächsten Abschnitt eine Klassifikation und die Spezifikation der Daten.

4.1.1.2 Klassifizierung und Spezifikation der Informationen

Bisher wurde untersucht, welche Aufgaben das Archiv hat und welche Informationen es zu deren Erfüllung benötigt. Noch wurden die Abhängigkeiten der Daten nicht von den möglicherweise mehrfach auftretenden Aufnahmen eines Patienten berücksichtigt.
Dazu mußte in Analysegesprächen geklärt werden, inwieweit die Geschichte des Patienten aus den Aufnahmedaten reproduzierbar sein soll. Die Abbildung 4.2 zeigt ein Klassifizie-

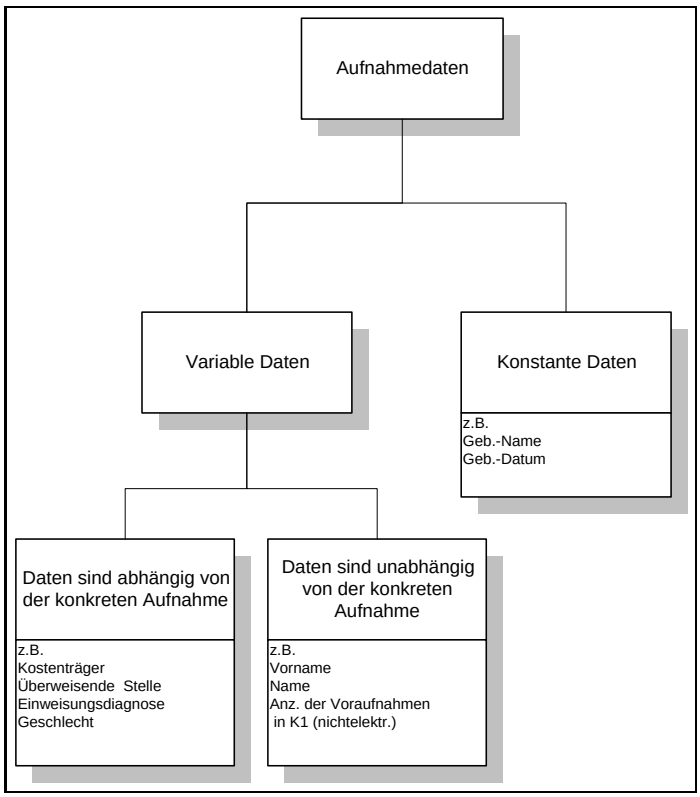


Abbildung 4.2: Klassifizierung der Archivdaten bezüglich ihrer Änderbarkeit

rungsschema, in das die Analysedaten eingeordnet wurden, um in der nächsten Phase des

Softwarelebenszyklusses die temporalen Abhängikeiten berücksichtigen zu können. Das Beispiel des Geschlechts eines Patienten verdeutlicht diesen Aspekt. Es wäre sogar falsch anzunehmen, daß sich das Geschlecht eines Patienten nicht ändern könne. Demzufolge muß die Art des Geschlechts in Abhängigkeit von der konkreten Aufnahme eingetragen werden. Dies betrifft zum Beispiel auch Einträge über den Familienstand, Anzahl der Kinder, Religion etc..
Andere Attributwerte wie Geburtsname oder Geburtsdatum können sich niemals ändern und sind deshalb auch nicht temporal von der Aufnahme abhängig.
Bei einer anderen Gruppe von Attributen wurde durch Analysegepräche festgelegt, daß selbst wenn sie sich ändern, nicht nachvollziehbar sein muß, zu welchem Zeitpunkt sie sich geändert haben. Als Beispiel sollen hier der Name und Vorname genannt werden.

4.1.1.3 Funktionale Analyse

Dieser Prozeß zog sich teilweise bis in die Implementierungsphase, um den Anwendern durch Prototypen die Möglichkeiten der elektronischen Datenverarbeitung zu veranschaulichen.
In den folgenden Unterabschnitten werden die hauptsächlichen Funktionen spezifiziert, die das PDAZN bieten soll. Dazu zählen die Patientendatenverwaltung, eine Allgemeine Datenpflege, der Import von Patientenaufnahmen, sowie ein kleiner Statistikteil.

4.1.1.3.1 Patientendatenverwaltung

In Abstimmung mit den Archivmitarbeitern enthält die Patientendatenverwaltung folgende Funktionalität:

- *Anzeigen*
  Die Informationsregister können angezeigt werden. Eine Manipulation der Daten ist dabei nicht möglich.
- *Bearbeiten*
  Es kann eine Bearbeitung der Informationsregister erfolgen.
- *Löschen*
  Eine Patientenaufnahme kann unwiderruflich gelöscht werden.
- *Neuaufnahme*
  Es besteht die Möglichkeit, eine Patientenaufnahme manuell einzutragen. Dabei soll auch eine Neuaufnahme auf der Grundlage bereits vorhandener Patientendaten möglich sein.

- *Suchen*
  Die Suche von Patientenaufnahmen soll durch die Angabe bestimmter Attribute erfolgen die nachfolgend aufgeführt sind:
  - Name
  - Vorname
  - Geburtsname
  - Geburtsdatum
  - Geburtsort
  - Aufnahme-Nummer(DVSE)
  - Epikrise (ja, nein, nicht beachten)
  - Klinik (Auswahl möglich)

4.1.1.3.2 Allgemeine Datenpflege

Die Werte der Attribute sollten möglichst problemlos um weitere erweitert oder geändert werden können.
Der Vorteil liegt in der Einheitlichkeit des verwendeten Vokabulars und der Einfachheit der Verwendung. Beispielsweise werden bei der Bearbeitung von Aufnahmedaten, für die folgenden Informationen Wertebereiche angeboten: "Familienstand", "Kostenträger", "Versicherungsstatus", "Höchster Schulabschluß", "Diagnosetyp", "Diagnosezeitpunkt"etc..

4.1.1.3.3 Import von Patientendaten

Eine weitere Funktion, die das PDAZN erfüllen muß, ist die Unterstützung des Patientdaten-Imports.
Dazu wurde zunächst analysiert, welche Informationen genutzt werden können, die durch das Patientenwesen bereits eingegeben wurden und sich auf einer Datenbank des DVSE befinden. Das Ergebnis war, daß ein Großteil der Informationen bereits in elektronischer Form vorliegt, die zum Register "Allgemeine Daten" gehören.

Ein Ergebnis der Analyse war auch, daß der Datenimport auf keinen Fall vollständig automatisiert werden kann.
Die Identifizierung eines zu importierenden Patienten in der PDAZN-Datenbank, kann nur persönlich durch einen Archivmitarbeiter vorgenommen werden. Kein Attribut und auch keine Kombination von Attributen kann die Identität eines bereits vorhandenen und eines zu importierenden Patienten zweifelsfrei feststellen, so daß an dieser Stelle ein Automatismus möglich wäre. Das wäre nur denkbar, wenn alle Patienten aus allen Ländern einen Ausweis

hätten, in dem zum Beispiel eine Ausweisnummer einen Patienten zuverlässig identifizieren würde.

Leider kann auch die Patientennummer, die in den Anwendungen des DVSE vergeben wird, nicht genutzt werden, da sie aufgrund ihrer Syntax in bestimmten Fällen neu vergeben wird. Die Bildungsvorschrift dieser Patienten-ID erzwingt bereits ein Informationsgehalt, der sich im Laufe des Lebens ändern kann. Beispielsweise besteht die erste Stelle der Patienten-ID aus dem Vornamen und die dritte Stelle aus einem Geschlechtskürzel. Das sind aber Patientendaten, die sich durchaus ändern können. Mit der nächsten Aufnahme führt diese Bildungsvorschift zwangsläufig zur Redundanz von Patientendaten, da dieser Patient eine neue Patientennummer bekommen muß.
Bei dem Patientendaten-Import, muß es demzufolge zwei Möglichkeiten geben, den Patienten in die Datenbank des PDAZN zu importieren:

- *Identifizierung positiv*
  Der Patient konnte zweifelsfrei identifiziert werden. Das bedeutet, daß er schon einmal aufgenommen wurde. Bei sich ändernden Werten eines Patienten, wurde die Aktualisierung der alten durch die neuen Daten festgelegt. Eine Aktualisierung der alten Patientendaten durch die neuen Daten erfolgt nur bei den Daten, die variabel und aufnahmeunabhängig sind (siehe Abschnitt 4.1.1.2). Dies betrifft ausschließlich einen Teil des Registers Allgemeine Daten. Dazu gehören Attribute wie Vorname, Name, Geburtsort etc.. Dagegen werden aufnahmeabhängige Daten natürlich nicht aktualisiert. Als Beispiele sollen hier Geschlecht, Familienstand, Religion genügen.

- *Identifizierung negativ*
  Kann der Patient nicht zweifelsfrei identifiziert werden oder wird eindeutig festgestellt, daß dieser Patient noch nie aufgenommen wurde, muß er als neuer Patient in die Datenbank des PDAZN aufgenommen werden.
  Dadurch besteht Gefahr der Datenredundanz, die aber einer fehlerhaften Zuordnung vorzuziehen ist. Im Zweifelsfall ist ein neuer Patient anzulegen.
  Aufgrund des begrenzten Umfangs der Arbeit, war es nicht möglich, redundante Patientendaten vereinen zu können. Das sollte für eine Folgeversion unbedingt vorgesehen werden.
  Das Auftrennen einer fehlerhafter Zuordnung kann allerdings auch in Zukunft nicht möglich sein.

Der Import der Daten soll soweit unterstützt werden, daß die Patientenaufnahmen temporär im PDAZN abgelegt werden. Diese müssen anschließend den Archivmitarbeitern als neue Aufnahmen präsentiert werden.
Die temporären Aufnahmen sollen nicht automatisch nach ihrem Import gelöscht werden, sondern nur bei Bestätigung.

Die Suchfunktion, die im Abschnitt 4.1.1.3.1 beschrieben wurde, soll den Datenimport, sowohl bei der Suche in temporären Aufnahmen, als auch in den bereits gespeicherten Aufnahmen ermöglichen.

4.1.1.3.4 Statistik

Da eine wesentliche Aufgabe des Archivs darin besteht, Anfragen zu bearbeiten, sollten die am häufigsten gestellten, durch entsprechende Formulare unterstützt werden, die eine Reihe von Attributen enthalten. Ein Großteil der Anfragen, wird durch die Suche nach folgenden Attributen abgedeckt:

- *Diagnose*
  Soll die Diagnose berücksichtigt werden, ist die Angabe des Diagnosesystems und des Diagnoseschlüssels erforderlich.
- *Aufnahmedatum*
  Muß das Aufnahmedatum beachtet werden, ist auch ein Zeitraum zu spezifizieren.
- *Alter des Patienten*
  Benötigt die Anfrage das Alter des Patienten, erfolgt die Angabe eines Geburtszeitraumes.
- *Klinik*
  Hier muß eine Auswahl der Klinik erfolgen können.
- *Kostenträger*
  Falls die Anfrage den Kostenträger einer Aufnahme berücksichtigen muß, ist eine Auswahl erforderlich.

Das Ergebnis dieser Anfragen soll eine Liste von Patientenaufnahmen sein, die auch ausdruckbar sein soll.
Durch die genannten Attribute läßt sich zum Beispiel folgende Anfrage schnell und problemlos bearbeiten:

Geben Sie mir bitte alle Patienten im Alter von 30-40 Jahren, die in dem Zeitraum B mit der Diagnose C des Diagnosesystems D stationär aufgenommen wurden!

4.1.2 Klinikspezifische Analysen

Da das PDAZN auch ein Informationssystem für die Kliniken des ZN ist, mußten spezifische Klinikdaten analysiert werden. Diese Informationen wurden wiederum, entsprechend ihrer Semantik, in verschiedene Register eingeteilt.
Die folgenden Unterabschnitte beschreiben die Analyseergebnisse, die durch mehrfache Gespräche und Präsentationen von Prototypen erzielt wurden.

4.1.2.1 Kinder- und Jugendklinik

In der Kinder- und Jugendklinik bestanden bereits sehr konkrete Vorstellungen darüber, welche Daten das Informationssystem speichern können muß. Zu den Attributen existierten auch eine eine Menge von Werten, die sehr hilfreich bei der Festelegung der Wertebereiche waren.
Durch diese Werte wurde allerdings auch deutlich, daß es für die Zukunft sehr unpraktisch wäre, diese Werte fest „verdrahtet“ in das System einzubinden. Darum wurde in den Gesprächen darauf verwiesen, daß eine Erweiter- und Abänderbarkeit dieser Werte vorteilhafter wäre. Aufgrund der hohen Akzeptanz wurde dieser wichtige Punkt berücksichtigt. Die Veränderung dieser Werte ist nur mit Archiv- oder Administratorrechten durchführbar.
Viele gewünschte Informationen allgemeiner Art, wie Name, Vorname, etc. wurden bereits durch die analysierten Archivdaten abgedeckt. Deshalb werden diese Daten im folgenden vernachlässigt und primär die Analyse der klinikspezifischen Daten diskutiert.
Aufgrund der spezifischen Daten ergab sich eine Einteilung der Attribute in folgende Register:

- *Beruf/Bildung/Soziales*
  Dieses Register enthält Daten des Patienten bezüglich der elterlichen Berufsausbildung, der eigenenen Bildung, Informationen über den ökonomischen Status der Familie, bis hin zu Angaben über den Verlust von Elternteilen und weiteren Daten.

- *Anamnese*
  Hier werden Daten abgespeichert, die zur Anamnese einer Patientenaufnahme gehören. Dazu zählen Attribute wie "Krampfanfälle in der Anamnese", "psychiatrisch/psychische Belastungen in der Familie" und "Angaben zur Menses".

- *Befunde*
  Die Befunde wurden weiter unterteilt in die Bereiche "Somatische Befunde", "Laborbefunde" und "Sonstige Befunde".

- *Behandlungen*
  Zu einer Behandlung zählen auch die Medikationsdaten. Dadurch kann sich die Klinik

schnell darüber informieren, mit welchen Medikamenten therapiert wurde. Dazu muß nicht immer die Angabe des speziellen Medikamentes notwendig sein. Erlaubt sind explizite Angaben zur verwendeten Medikamentgruppe. Weiterhin kann auch nur die Untergruppe angegeben werden, wobei das System die Zuordnung zur entsprechenden Gruppe kennt. Ebenso verhält es sich bei der Generic-Angabe. Hier wird auch durch das System erkannt, zu welcher Gruppe und/oder Untergruppe dieses Generic gehört. Schließlich ist auch die Angabe von Handelsnamen möglich, die auch wieder eine Zuordnung zu Generic, Gruppe und/oder Untergruppe besitzen.
Das Attribut "Andere Behandlungsverfahren/Methoden" ermöglicht weitere Angaben.

4.1.2.2 Neurologie

Die Analyse für die Neurologie ergab ebenfalls eine Unterteilung der Attribute in mehrere Register.
Dabei wurde darauf geachtet, inwieweit neue Informationen ähnlich zu bereits analysierten aus der Kinder- und Jugendklinik sind. Das Ziel bestand darin, für gleiche Sachverhalte auch gleiche Namen zu verwenden, um die Informationsstruktur nicht durch Synonyme unverständlicher zu gestalten.
Die erforderlichen Attribute wurden in folgende Register eingeteilt:

- *Anamnese*
  Dieses Register enthält Informationen über bestehende Zweiterkrankungen und einer weiteren Information zur Klinikaufnahme.

- *Befunde*
  Die Befunde wurden in "Somatische Befunde", "Laborbefunde", "Psychologischer Befund" und "Sonstige Befunde" eingeteilt.

- *Behandlungen*
  Die medikamentöse Therapie wurde bereits für die Kinder- und Jugendklinik in Abschnitt 4.1.2.1 beschrieben. Zusätzlich ist die Angabe weiterer Behandlungen möglich.

4.1.2.3 Psychiatrie

Bei der Psychiatrie bestand ebenfalls der Wunsch, umfangreiche Informationen abspeichern zu können.
Durch die Semantik der Informationen wurden die Attribute in folgende Register untergliedert:

- *Beruf/Bildung/Soziales*
  Dazu gehören Informationen zur eigenen Bildung, zum Berufabschluß und Angaben zu Formen der Betreuung.

- *Anamnese*
  Dieses Register enthält lediglich die Attribute "psychiatrisch/psychische Belastungen in der Familie" und "Angaben zur Suizidalität".

- *Befunde*
  Die Befunde werden in "Somatische Befunde", "Psychopathologische Befunde", "Laborchemie" und "Laborbefunde" unterteilt.

- *Behandlungen*
  Zur Behandlung gehört ebenfalls die Medikation.
  Weitere Angaben betreffen die "Psychotherapeutische Behandlung", "Weitere Behandlungen". Unter einer weiteren Einteilung "Sonstige Befunde" können auch "Konsile" und die "Art der Therapiebeendigung" angegeben werden.

4.1.2.4 Psychosomatik

Aufgrund des vorhandenen EDV-Systems in dieser Klinik wurde nur ein Register hinzugefügt:

- *Beruf/Bildung/Soziales*
  Das Register enthält nur die Attribute "Höchster Schulabschluß", "Letzte Berufstätigkeit" und "Feste Partnerschaft".

In den Analysegesprächen wurde darauf hingewiesen, daß der Informationsgehalt aufgrund der geringen Anzahl der Attribute sehr gering ist. Es bestand ausdrücklich der Wunsch, nicht mehr Daten in das PDAZN aufzunehmen.
Die Psychosomatik hat Zugriff auf die Informationen des Patienten durch das eigene EDV-System, aber andere Kliniken, die sich durch das PDAZN über den Patienten informieren möchten, bekommen nur sehr wenig Informationen bezüglich der Aufnahmen in der Psychosomatik. Eine Erweiterbarkeit des PDAZN für die Psychosomatik, unter den genannten Gesichtspunkten, wäre sehr wünschenswert.

4.1.3 Veränderung des Informationsflusses

Nachdem in den vorangegangenen Abschnitten Informationen und Aufgaben spezifiert wurden, soll an dieser Stelle diskutiert werden, wie sich der Informationsfluß zwischen den betei-

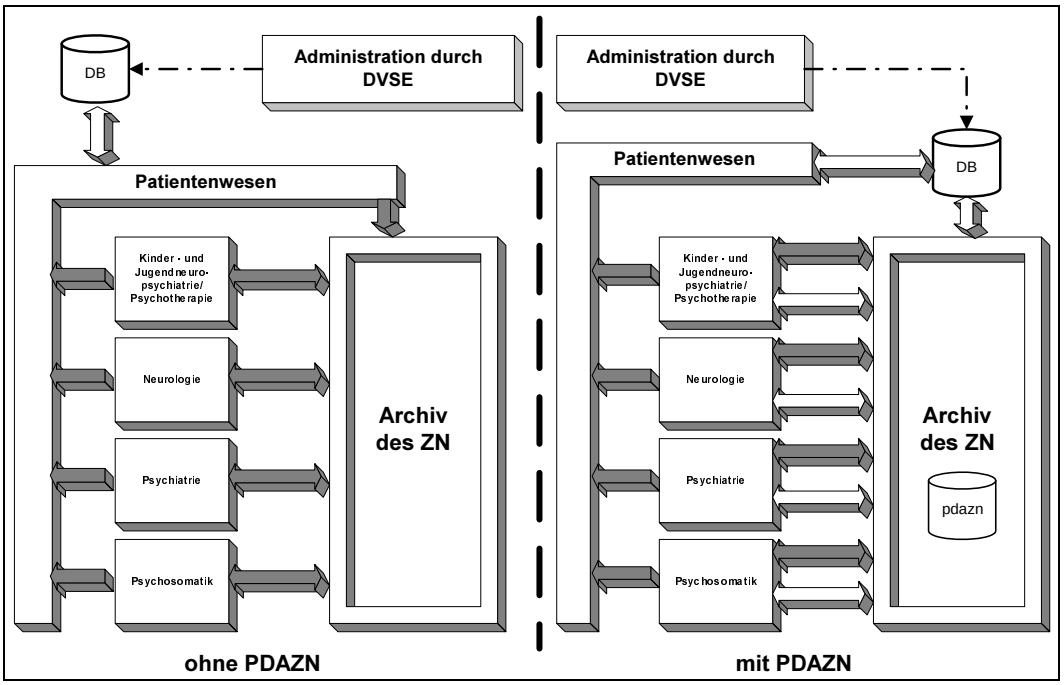


Abbildung 4.3: Die Veränderung des Informationsflusses beim Einsatz des PDAZN

ligten Einrichtungen durch Einsatz des PDAZN verändern wird. Die linke Seite der Abbildung 4.3 stellt zunächst den Informationsfluß ohne PDAZN dar, während die rechte Seite die Veränderung mit Einsatz des PDAZN darstellt. Die grau unterlegten Pfeile demonstrieren den konventionellen Informationsaustausch, also mit Weitergabe von Listen, persönlichen Anfragen etc.. Die hellen Pfeile deuten auf den Transport von Informationen über das Subnetz der Medizinischen Fakultät. Dazu wird die Nutzung von Informationen per Computer vorausgesetzt, auf dem sich die PDAZN-Applikation befindet.

Auf der linken Seite existiert kein elektronischer Datenaustausch zwischen dem Archiv und den Kliniken und das Archiv erhält außerdem die Daten auf konventionellem Wege vom Patientenwesen.

Auf der rechten Seite der Abbildung 4.3 wird deutlich, daß die Kliniken bei Einsatz des PDAZN auch elektronisch auf Archivdaten zugreifen können. Der grau unterlegte Pfeil konnte allerdings nicht vollständig ausgetauscht werden, da die Kliniken nicht uneingeschränkten Zugriff haben, sondern bestimmte Anfragen nur über das Archiv stellen dürfen.
Der Informationsfluß vom Patientenwesen zum Archiv entfällt, da nun die Daten direkt von der Datenbank des DVSE in das PDAZN eingespielt werden.

4.1.4 Forderungen des Datenschutzes

Bisher wurden die ermittelten Informationsregister beschrieben und einige Beispiele vorgestellt.
Bei sehr sensiblen medizinischen Daten, wie den Aufnahmedaten, ist eine dedizierte Rechtevergabe erforderlich.

Aus den Gesprächen mit dem Datenschutzverantwortlichen der Medizinischen Fakultät, dem Archivverantwortlichen, sowie aus den Gesprächen mit den Vertretern der einzelnen Kliniken des ZN, gingen folgende Restriktionen hervor:

- Die Daten einer Klinik dürfen nur von bestimmten Personen anderer Kliniken eingesehen werden.
- Das Informationsregister Sterberegister darf nur von Administratoren oder Archivangestellten angezeigt und bearbeitet werden.
- Die Informationsregister "Allgemeine Daten" und "Diagnosen" einer Klinik dürfen von Klinikangehörigen und bei entsprechender Berechtigung auch von Angehörigen anderer Kliniken eingesehen werden. Eine Bearbeitung dieser Register erfolgt grundsätzlich nur durch die Administratoren oder die Archivmitarbeiter.
- Der Import von Patientenaufnahmen, Nutzung des Statistikteils und die Pflege der Stammdaten ist nur durch Administratoren oder Archivmitarbeiter möglich. Die Funktion der "Allgemeinen Datenpflege" ist auch bei den Archivmitarbeitern nur im begrenzten Umfang möglich. Die Pflege von Stammdaten wie Aufnahmemodi, Entlassungsmodi, Stationsnamen, Nationalitäten, Religionen dürfen beispielweise nur Administratoren vornehmen. Die Ursache liegt darin, daß bestimmte Informationen sowohl im PDAZN als auch in Applikationen des DVSE die gleiche Bezeichnung tragen müssen, um den Import von Patientenaufnahmen zu gewährleisten. Dabei stellt das Vokabular der DVSE-Applikationen oft nur eine Teilmenge der Informationen dar, die im PDAZN verwendet werden.

4.2 Der Entwurf

Entsprechend dem allgemeinen Softwareentwurf und dem Datenbankentwurf wurde bisher die funktionale und die Informations-Analyse dargestellt. Die sich nun anschließende Entwurfsphase unterteilt sich in den Entwurfsteil für die Datenbank und dem Programmentwurf. Der Abschnitt 4.2.1 wird sich zunächst mit dem Datenbankentwurf und Abschnitt 4.2.2 mit dem Entwurf des Programmes beschäftigen.

4.2.1 Datenbank-Entwurf

Die Datenbankentwicklung unterteilt sich analog dem allgemeinen Softwareentwurf in mehrere Phasen. Nach der erste Phase, der Informationsanalyse, wird in diesem Abschnitt der konzeptionelle und der logische Entwurf diskutiert und an einigen Beispielen verdeutlicht.

4.2.1.1 Konzeptioneller Entwurf

Im konzeptionellen Entwurf wird die Darstellung der Beziehungen zwischen den Informationsobjekten, die Zuordnung von konkreten Datentypen zu den analysierten Wertebereichen und die Festlegung der Integritätsbedingungen vorgenommen.
Im nachfolgenden Abschnitt wird zunächst die Entwicklung des ER-Modells diskutiert.
Der sich anschließende Abschnitt beschreibt das Vorgehen bei der Zuordnung konkreter Datentypen des SQL-Servers zu den analysierten Wertebereichen.
Der letzte Abschnitt enthält eine Erläuterung, wie weitere Intergritätsbedingungen spezifiziert wurden.

4.2.1.1.1 ER-Modell

Als Modell zur Darstellung des konzeptionellen Entwurfs wurde das Entity-Relationenship-Model, kurz ER-Modell, gewählt. Das ER-Modell wird in [HeSa95] ausführlich beschrieben und soll im folgenden als bekannt vorausgesetzt werden. Zusätzlich müssen für diese Arbeit noch einige Festlegungen getroffen werden.
Die Abbildung 4.4 stellt eine Beziehung R zwischen zwei Entity-Typen A und B dar. Dabei gilt die folgende Interpretation:

Ein Entity des Typs A kann maximal mit einem Entity des Typs B, aber ein Entity des Typs B mit mehreren Entities vom Typ A in Beziehung treten.

Diese Interpretation entspricht dem sogenannten *Standard-ER-Modell mit Kardinalitäten.*

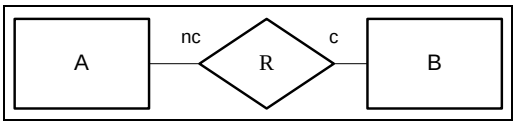


Abbildung 4.4: ER-Modell zur Interpretation

Die Angabe *c* bedeutet dabei, daß es sich um eine optionale Beziehung handelt. Weiterhin werden nur zweistellige Beziehungen verwendet.
Um eine einheitliche Betrachtungsweise zu gewährleisten, soll auch auf die häufig anzutreffende Intervallnotation eingegangen werden, bei der die Angabe von (min,max) Auskunft gibt, wie oft ein Entity an der Beziehung teilnimmt. Es soll an dieser Stelle noch einmal hervorgehoben werden, daß die Intervallnotation die Teilnahmeanzahl der Entities an einer Beziehung ausdrückt. In dieser Arbeit wird aber betrachtet, mit wievielen Entities ein konkretes Entity in Beziehung steht. Dadurch ist diese Kardinalitätsangabe genau umgekehrt zur Intervallnotation.

Die Schlüsselattribute der Entity-Typen werden im ER-Modell wie üblich unterstrichen dargestellt. Dagegen wird beim logischen Entwurf deutlich, welche Schlüssel zum Primärschlüssel des Relationship-Typs werden. Die Festlegung der Schlüsselattribute für die Entity-Typen geht mit der Entwicklung eines ER-Modells einher, da es immer zwingend erforderlich ist, ein Entity genau zu identifizieren. Mitunter ist es auch notwendig, Attribute hinzuzufügen, wenn vorhandene Attribute ein Entity nicht eindeutig identifizieren können. Dazu wird weiter unten in diesem Abschnitt ein Beispiel erfolgen.
Die weiteren Integritätsbedingungen, sind die Typintegrität und die referentielle Integrität. Zur Typintegrität gehören die Festlegung der Datentypen zu Attributen in Abschnitt 4.2.1.1.2 und die Angabe ihrer Wertebereiche. Weiterhin können default-Werte vergeben werden und Angaben darüber, ob der Wert auch ein NULL-Wert sein darf oder nicht.
Die referentielle Integrität stellt eine weitere wichtige Integritätsbedingung dar, die in Abschnitt 4.2.1.2 im Zusammenhang mit dem logischen Entwurf erläutert wird. Dort erfolgt ein exemplarisches Beispiel anhand der entstandenen Relationenschemata mit der zusätzlichen Angabe von default- und NULL-Werten.

Für das PDAZN gibt es sowohl ein Gesamt-ER-Modell, als auch ein spezifisches ER-Modell für jede Klinik. Bestimmten Relationenship-Typen oder Attributen werden in diesen Modellen Metainformationen hinzugefügt, wenn es sich um nützliche Zusatzinformationen handelt, die der besseren Verständlichkeit dienen.
Die Abbildung 4.5 zeigt einen Ausschnitt aus dem ER-Modell, bei dem ein Relationenship-Typ mit zwei Typen von Metadaten gekennzeichnet wurde. Dabei werden die Metainformationen durch die öffnenden oder schließenden Klammern dargestellt. Die Darstellung der Metadaten mit öffnenden oder schließenden Klammern hängt von der Position zum Symbol ab. Diese Klammern können mittels durchgehender oder gestrichelter Linie dargestellt werden.

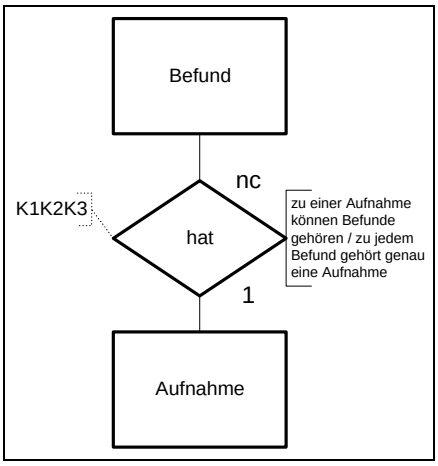


Abbildung 4.5: Metainformationen im ER-Modell

Die durchgehende Klammer enthält beschreibenden Text und wird sowohl für das Gesamt- als auch für das spezifische ER-Modell eingesetzt. Die gestrichelt dargestellte Klammer beinhaltet eine Aufzählung von Synonymen der Kliniknamen und wird nur im Gesamt-ER-Modell angegeben. Das ist erforderlich, um zu kennzeichnen, welche Attribute und Beziehungen von den jeweiligen Kliniken benötigt werden. Für die Kliniken gelten dabei folgende Synonyme:

Kinder- und Jugendklinik $\Longrightarrow$ K1
Neurologie $\Longrightarrow$ K2
Psychiatrie $\Longrightarrow$ K3
Psychosomatik $\Longrightarrow$ K4

Diese Synonyme werden aus Gründen der Übersichtlichkeit, nicht nur im ER-Modell, sondern auch später in der Implementation verwendet. Sie eignen sich für die Darstellung wesentlich besser, als lange bezeichnende Namen. In der Abbildung 4.5 gilt beispielsweise die Beziehung nur für die angegebenen Kliniken K1, K2 und K3. Für K4 gibt es diese Beziehung nicht.
Diese Metainformationen werden nur an Attributen des Entity-Typs Aufnahme und an Relationenship-Typen dargestellt. Die Entity-Typen müssen nicht gekennzeichnet werden, da zweistellige Beziehungen vorausgesetzt werden. Wenn die Beziehung gilt, müssen die Entities zwangsläufig dazugehören. Wie eben erwähnt, besitzen auch manche Attribute diese Metainformationen, um zu verdeutlichen, daß diese Attribute nur für die genannten Kliniken erforderlich sind.

Ein sehr wichtiger Aspekt bei dem Entwurf bestand darin, zu entscheiden, ob die Informationen als Attribute oder als Entity-Typ modelliert werden. Dazu folgen zunächst einige Ausführungen:
Werden bestimmte Informationen als Attribute modelliert, deren Werte sich oft wiederholen,

besteht die Möglichkeit dieses Attribut als ein Entity zu modellieren. Der Vorteil liegt in den folgenden drei Punkten begründet:

- *Einsparung von Speicherplatz*
  Die Redundanz einer Information erfordert zwangsläufig mehr Speicherplatz.
  Die betreffenden Attribute eines Entity-Typs werden dann jeweils als eigene Entities modelliert, die dann mit dem Entity-Typ, aus dem sie hervorgegangen sind, in Beziehung stehen.

- *Einheitliches Vokabular*
  Der Vorteil liegt in einer einheitlichen Bezeichnung eines Sachverhaltes, der am Beispiel der Nationalität eines Patienten verdeutlicht werden soll. Sie ist zunächst als Attribut zu betrachten und ist im Sinne dieser Arbeit nicht von der Aufnahme abhängig, da sie sich nicht ändern kann. Die Nationalität ist nicht mit der Staatsangehörigkeit gleichzusetzen. Folgende Angaben gehören beispielsweise zur Nationalität: "deutsch", "polnisch", "kroatisch", "serbisch" usw.. Als Attribut modelliert, hat der Anwender zu große Freiräume hinsichtlich der Bezeichnungsvielfalt. Damit wäre für "deutsch" auch möglich: "Deutsch", "Deutscher", "Deutsche", im schlimmsten Fall sogar "BRD" oder "Deutschland". Als Entity modelliert, soll es für den Anwender nur die Möglichkeit geben, "deutsch" zu wählen. Bei dem genannten Beispiel hätte es keine inhaltlichen Probleme bei der Verwendung der vielfältigen Bezeichnungen gegeben. Dennoch werden Suchfunktionen kein vollständiges Ergebnis liefern können, wenn nach "deutsch" gesucht wird, aber auch der Eintrag "Deutscher" verwendet wurde und die Suchfunktion die Groß- und Kleinschreibung berücksichtigt.
  Es besteht aber auch die Gefahr, daß für bestimmte Attribute Abkürzungen verwendet werden, die nur durch den Anwender richtig interpretiert werden, von dem sie eingegeben wurden.
  Die Verwendung eines einheitlichen Vokabulars erzwingt eine vorherige Absprache mit allen Anwendern, die selbst diese Werte nicht ändern können, sondern nur von bestimmten authorisierten Personen.

- *Vereinfachung der Eingabe*
  Die Eingabe von Informationen kann dadurch erleichtert werden, daß der Anwender eine Auswahl der einzufügenden Werte erhält und nur noch aus diesen auswählen muß.

Es gibt aber auch den Nachteil, der Komplexitätserhöhung des ER-Modells. Unter bestimmten Umständen kann das auch zu einem Performance-Einbruch, der später zu implementierenden Anwendung, führen, da das DBMS wesentlich mehr Verwaltungsaufwand betreiben muß. Sei es nun, je nach DBMS, die Verwaltung von persistenten Objekten oder die Verwaltung von Tabellen.
Aufgrund der Vorteile wurde die Modellierung zum Entity-Typ bevorzugt, auch wenn dadurch die Komplexität des ER-Modells und deshalb auch der Entwicklungsaufwand gesamten

Anwendung enorm anstieg.
In der Analysephase wurden die Daten nach ihrer Änderbarkeit klassifiziert (siehe Abbildung 4.2). Um zu verdeutlichen, wie diese Klassifikation beachtet wird, ist ein ausgewähltes Beispiel in der Abbildung 4.6 dargestellt. Diese Abbildung enthält nur einen sehr kleinen Auszug vom Gesamt-ER-Modell, um zu erläutertern, wie die Klassifikation der Analysedaten umgesetzt wurde:

Im Mittelpunkt des ER-Modells steht die Patientenaufnahme, kurz Aufnahme, mit der die meisten anderen Entity-Typen in Beziehung stehen. Der Grund liegt in der Analyse der Informationen, die natürlicherweise zum größten Teil aufnahmeabhängig sind.
Das Attribut AufnNr wurde als neues Attribut zum Entity-Typ Aufnahme, mit der Funktion als Primärschlüsselattribut zugeordnet, da es die einzige Möglichkeit ist, eine Aufnahme eindeutig zu identifizieren. Dieses Attribut wird durch einen abstrakten numerischen Wert dargestellt, der auf keinen Fall mit der Aufnahme-Nummer des DVSE gleichgesetzt werden darf, denn für mehrere Aufnahmen im Sinne des PDAZN, kann es eine einzige Aufnahmenummer des DVSE geben. Die Gründe dafür, wurden in der Analyse ausführlich diskutiert. Die Aufnahme des DVSE wurde als Attribut AufnNrDVSE spezifiziert und dem Entity Aufnahme zugeordnet.
Die variablen aufnahmeunabhängigen Attribute, wie Vorname, Name und konstante Daten, wie Geburtsname, Geburtsdatum wurden als Attribute des jeweiligen Entity-Typs modelliert. Sie sind demzufolge für jede Aufnahme gleich und deshalb dem Entity-Typ Patienten zugeordnet.
Auch die Nationalität ist aufnahmeunabhängig und hat deshalb eine Beziehung zum Patienten und nicht zur Aufnahme.
Dagegen sind aufnahmeabhängige Attribute, wie beispielsweise das Geschlecht, als Attribute des Entity-Typs Aufnahme modelliert.
Die aufnahmeabhängigen Daten, die in Form von Entity-Typen modelliert wurden, müssen aus Gründen der Abängigkeit zur Aufnahme, eine Beziehung zu dieser haben. Die Abbildung 4.7 stellt einen weiteren Auszug aus dem ER-Modell dar, um zu verdeutlichen, wie die Abhängigkeit der Entity-Typen von der Aufnahme modelliert wurde. Der Entity-Typ Kostenträger kann beispielsweise für jede Aufnahme unterschiedlich sein und hat deshalb eine direkte Beziehung zum Entity-Typ Aufnahme. Andere Beispiele sind Versicherungsstatus, Familienstand und Religion.
An diesem Auszug wird auch deutlich, daß Attribute, die berechnet werden können, durch gestrichelte Linien dargestellt werden. Dabei erfolgt auch die Angabe, wie dieses Attribut berechnet wird. Das Attribut Verweildauer wird durch Subtraktion des Aufnahmedatums vom Entlassungsdatum berechnet. Diese Beispiele sollen ausreichen, um zu vermitteln, wie das ER-Modell entstanden ist und wie es interpretiert werden muß.
Das vollständige Gesamt-ER-Modell und die spezifischen ER-Modelle der Kliniken, sind im Anhang zu finden.

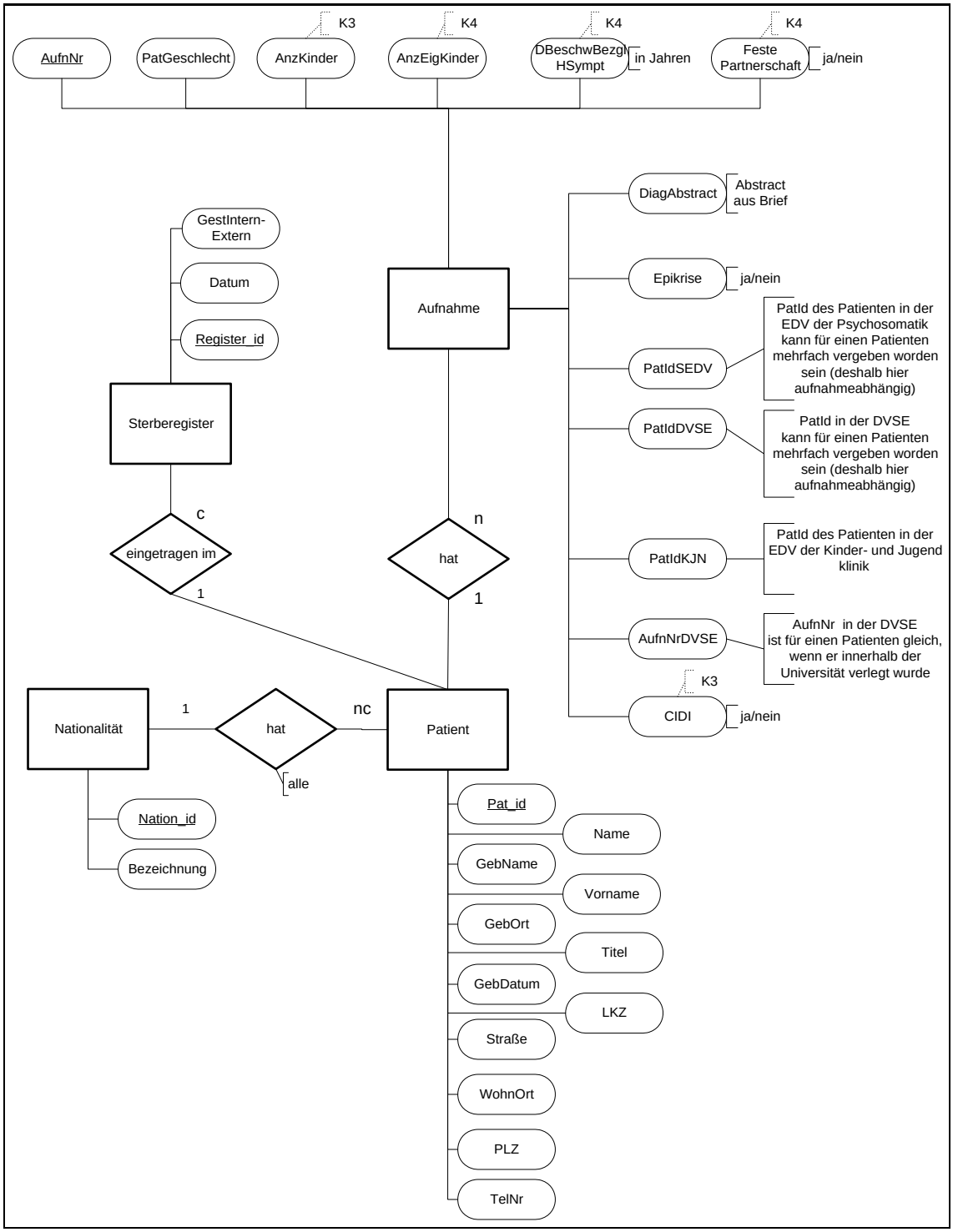


Abbildung 4.6: ER-Modell-Auszug 1

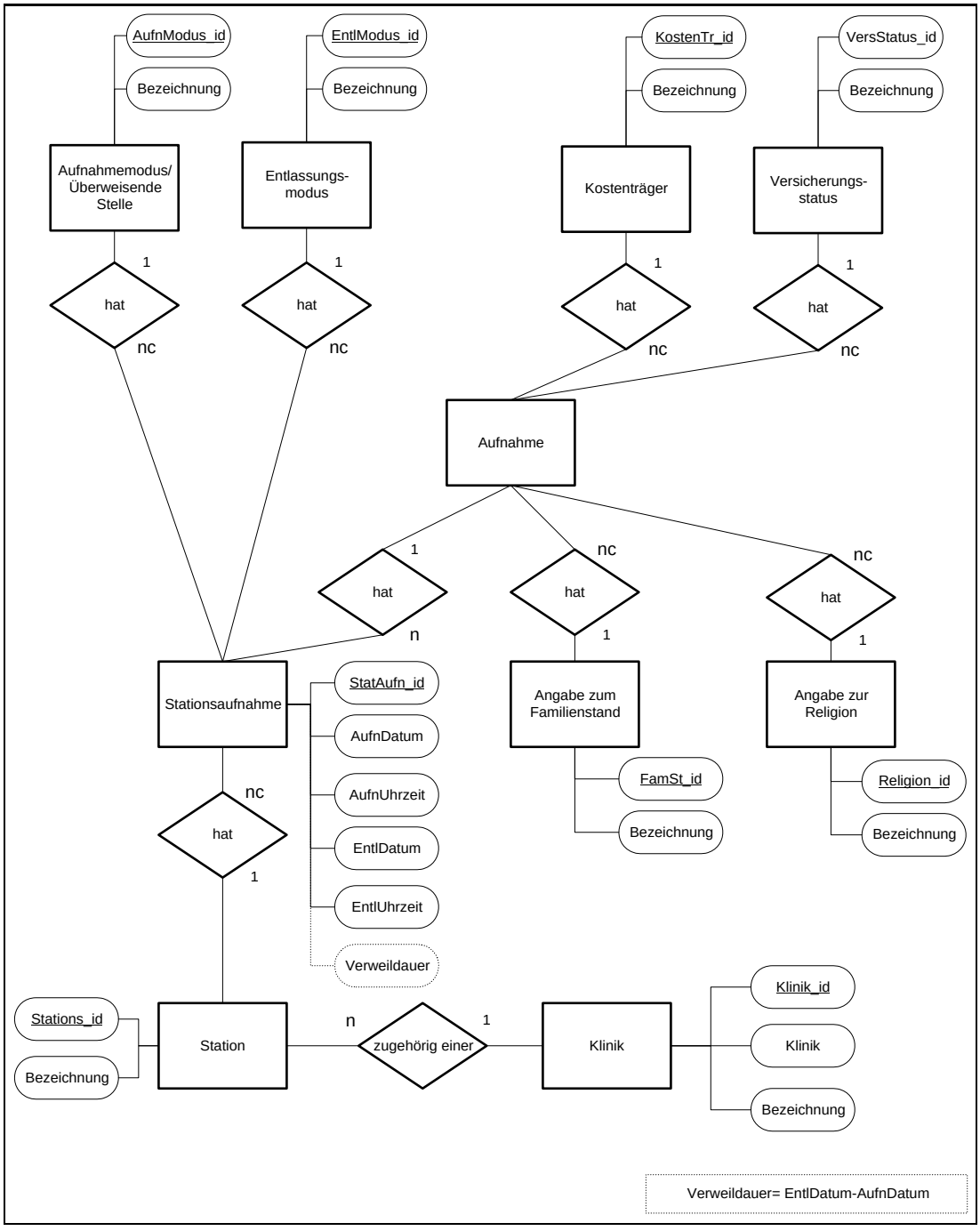


Abbildung 4.7: ER-Modell-Auszug 2

Entity Aufnahme		
attribute	*type-specification*	*type-MS SQL-Server*
AufnNr	int/4 byte	integer
PatGeschlecht	m/w	bit
DBeschwBezglHSympt	int/1 byte	tinyint
FestePartnerschaft	ja/nein	bit
AnzEigKinder	int/1 byte	tinyint
AnzKinder	int/1 byte	tinyint
CIDI	ja/nein	bit
PatIdSEDV	int/4 byte	integer
PatIdDVSE	char/10	varchar(10)
PatIdKJN	char/10	varchar(10)
AufnNrDVSE	char/15	varchar(15)
DiagAbstract	char/2000	text
Epikrise	ja/nein	bit

Tabelle 4.1: Zuordnung von Datentypen zu Attributen aufgrund ermittelter Wertebereiche

4.2.1.1.2 Zuordnung von Datentypen zu Attributen

Eine weiterer Aufgabe des konzeptionellen Entwurfs ist die Bestimmung der Datentypen. In der Analyse wurden Beispieldaten erfaßt und Informationen darüber, wie sich die Werte eines Attributs entwickeln können.
Am Beispiel des Entity-Typs Aufnahme soll erläutert werden, wie den analysierten Wertebereichen konkrete Datentypen des MS SQL-Servers zugeordnet wurden. Der MS SQL-Server hat spezifische Datentypen wie integer, smallint, tinyint, bit, text, image, usw. die den Wertebereichen zugeordnet werden müssen. An dieser Stelle soll keine vollständige Beschreibung der Datentypen erfolgen, sondern nur anhand eines Beispiels die Zuordnung gezeigt werden.

Die Tabelle 4.1 stellt die Attribute des Typs Aufnahme mit den vorher analysierten Informationen und der Zuordnung zu konkreten Datentypen dar. In der Spalte type-specification befinden sich die Angaben über Typ und Größe der Werte des konkreten Attributes. Beispielsweise wurde beim Attribut DiagAbstract eine notwendige Zeichenkettenlänge von 2000 festgestellt. Dafür bietet der SQL-Server den alphanumerischen Datentyp "text" an, der Zeichenketten bis zu 2 Gbyte darstellen kann. Der Datentyp "text" gehört, wie "image", zu den BLOBS (Binary Large Objects), auf die nur durch BLOB-Funktionen zugegriffen werden kann.
Durch den genannten image-Datentyp ist es in Zukunft möglich, große Bilder abzuspeichern. In der Analyse des PDAZN bestand dieser Wunsch zunächst nicht. Dies kann allerdings für die Zukunft von Interesse sein, wenn Befunde wie Röntgenbilder in digitalisierter Form vor-

liegen und dadurch elektronisch abgespeichert werden können.

Andere Attribute besitzen nur zwei Werte, wie etwa FestePartnerschaft mit ja/nein oder PatGeschlecht mit m/w. Diese Wertebereiche können durch den Datentyp "bit" abgedeckt werden.
Bei numerischen Attributen, wie AufnNr, AnzKinder wurden aufgrund der unterschiedlichen Anforderungen an die Wertebereiche entsprechende Datentypen zugeordnet. Der Datentyp "integer", der durch vier Byte dargestellt wird, läßt über vier Milliarden Aufnahmen zu ist deshalb ausreichend. Der nächst kleinere Datentyp smallint mit zwei Byte, wäre mit etwa 65000 Aufnahmen nicht ausreichend. Für das Attribut AnzKinder, der Anzahl der von Kindern, ist eine maximalen Anzahl von 255 ausreichend. Aus diesem Grund wurde hier der Datentyp "tinyint" zugeordnet, der durch ein Byte dargestellt wird.
Der Anhang enthält die tabellerische Darstellung aller Attribute mit den zugeordneten Datentypen.

4.2.1.2 Logischer Entwurf

Der logische Entwurf schließt sich dem konzeptionellen Entwurf an und hat für das PDAZN eine Umsetzung in Relationenschemata zum Ziel, da der SQL-Server ein relationales DBMS ist.
Dazu wird in Abschnitt 4.1.1.2.1 die Transformation des ER-Modells in das Relationenmodell beschrieben.
Die Entity- und Relationenship-Typen werden bei diesem Prozeß auf Relationenschemata abgebildet. Anschließend erfolgt der Test auf Normalformen in Abschnitt 4.2.1.2.2.

4.2.1.2.1 Transformation des ER-Modells in das Relationenmodell

Die Transformation des ER-Modells in das Relationenmodell erfolgt nach bestimmten Regeln, deren theoretische Grundlagen ausführlich in [HeSa95] erklärt sind.
Zunächst werden die in der Abbildung 4.6 dargestellten Entity-Typen beispielhaft in Relationenschemata transformiert:

Der Entity-Typ Aufnahme bekommt als neuen Namen zum Beispiel tblAufnahmen, wobei die Attribute des Entity-Typs Aufnahme übernommen werden. Bei der Notation steht in eckigen Klammern der Tupelkonstruktor, hier ohne explizite Angabe des Datentyps. Die geschweiften Klammern sollen ausdrücken, daß es sich bei einer Relationsausprägung um mehrere Tupel handelt.

Entity-Typ Aufnahme: $\Longrightarrow$ **tblAufnahmen**:

{[AufnNr, PatGeschlecht, DBeschwBezglHSympt, FestePartnerschaft, AnzEigKinder, AnzKinder, CIDI, PatIdSEDV, PatIdDVSE, PatIdKJN, AufnNrDVSE, DiagAbstract, Epikrise]}

Entity-Typ Patient: $\Longrightarrow$ **tblPatienten**:

{[Pat_id, Name, Vorname, Titel, GebOrt, GebDat, Wohnort, LKZ, PLZ, Straße, TelNr]}

Entity-Typ tblRegister: $\Longrightarrow$ **tblRegister**:

{[Register_id, Datum, GestInternExtern]}

Entity-Typ Nationalität: $\Longrightarrow$ **tblNationalitäten**:

{[Nation_id, Bezeichnung,]}

Nun müssen noch die Relationship-Typen in Relationenschemata transformiert werden. Dazu werden die Schlüssel der beteiligten Entities und falls vorhanden, die Attribute des Relationenship-Typs übernommen.
Der Primärschlüssel des Relationenschema wird dabei auf folgende Weise bestimmt:

- 1:1 Beziehungen: Wahl eines Primärschlüssels aus den beteiligten Entity-Typen.
- 1:n Beziehungen: Wahl des Primärschlüssels der n-Seite.
- n:m Beziehungen: Beide Primärschlüssel zusammen bilden den neuen Primärschlüssel.

Da für die Relationenship-Typen zumeist Verben verwendet wurden, müssen die beteiligten Entity-Typ-Namen bei der Beschreibung hinzugenommen werden, um eine eindeutige Zuordnung zu ermöglichen. Der gebildete Namen hat folgende Struktur:

Entity_Relationship_Entity

Dadurch ergeben sich folgende Relationenschemata:

Relationenship (n:1) Aufnahme_hat_Patient $\Longrightarrow$ **tblPatsAufnahmen**:

{[AufnNr, Pat_id]}

Relationenship (nc:1) Patient_hat_Nationalität $\Longrightarrow$ **tblPatsNatio**:

{[Pat_id, Nation_id,]}

Relationenship (1:c) Patient_eingetragen_im_Sterberegister $\Longrightarrow$ **tblVerstorbenePats**:
Dabei wurde Pat_id als Primärschlüssel festgelegt.

{[Pat_id, Register_id]}

Das Ergebnis dieser Transformation wird auch als Initial-Entwurf bezeichnet, da sich nun eventuell noch Schemaverfeinerungen vornehmen lassen. Die gesamte Transformation wird dahingehend untersucht, ob sich bestimmte Relationenschemata zusammenfassen lassen. Nach dem Zusammenfassen ergeben sich folgende Schemata:

tblRegister $\cup$ tblVerstorbenePats $\Longrightarrow$ tblRegister

{[Register_id, Datum, GestInternExtern, Pat_id]}

tblAufnahmen $\cup$ tblPatsAufnahmen $\Longrightarrow$ tblAufnahmen

{[AufnNr, PatGeschlecht, DBeschwBezglHSympt, FestePartnerschaft, AnzEigKinder, AnzKinder, CIDI, PatIdSEDV, PatIdDVSE, PatIdKJN, AufnNrDVSE, DiagAbstract, Epikrise, Pat_id]}

tblPatienten $\cup$ tblPatsNatio $\Longrightarrow$ tblPatienten

{[Pat_id, Name, Vorname, Titel, GebOrt, GebDat, Wohnort, LKZ, PLZ, Straße, TelNr, Nation_id]}

Zum Schluß dieses Abschnittes folgt ein vollständiges Beispiel für ein Relationenschema in Tabelle 4.2. Als Beispiel dient das Relationenschema tblAufnahmen, das noch mit weiteren verschmolzen wurde. Dabei erfolgen auch Angaben über die festgelegten Datentypen, Default-Werte, ob NULL-Werte gestattet sind, ob das jeweilige Attribut ein Teil des Primärschlüssels ist und in welchem Relationenschema dieses Attribut Primärschlüssel und dadurch in diesem Schema Fremdschlüssel ist.
Diese Informationen werden im Anhang für alle Relationenschemata dargestellt. Sie bilden die Voraussetzung für die Datendefinion, dem nächsten Entwicklungsschritt.

tblAufnahmen					
attribute	*type*	*default*	*NULL*	*primary-key*	*foreign-key*
AufnNr	integer			X	
PatGeschlecht	bit				
DBeschwBezglHSympt	tinyint	0			
FestePartnerschaft	bit	0			
AnzEigKinder	tinyintint	0			
AnzKinder	tinyint	0			
CIDI	bit	0			
PatIdSEDV	integer		X		
PatIdKJN	varchar(15)		X		
PatIdDVSE	varchar(15)		X		
AufnNrDVSE	varchar(15)		X		
Pat_id	integer				tblPatienten
KostenTr_id	tinyint				tblKostenTr
VersStatus_id	tinyint				tblVersStatus
FamSt_id	tinyint				tblFamstaende
Religion_id	tinyint				tblReligionen
Klinik_id	integer				tblKliniken
DiagAbstract	text		X		
Epikrise	bit		X		

Tabelle 4.2: Relationenschema tblAufnahmen - Notwendige Daten nach der Schemaverfeinerung für den nächsten Entwicklungsschritt, der Datendefinion

4.2.1.2.2 Normalformen

Das Entwurfsziel eines Datenbankschemas sollte in aller Regel die dritte Normalform sein, die die erste und zweite Normalform impliziert.
Das Datenbankschema des PDAZN entspricht zwar der zweiten, aber nicht der dritten Normalform. Die zweite Normalform verlangt, daß es keine funktionalen Abhängigkeiten von Nichtschlüsselattributen zu Teilen des Primärschlüssels gibt. Es dürfen also keine partiellen Abhängigkeiten auftreten. Diese Form der Abhängigkeit war im Datenbankschema nicht festzustellen, dagegen aber transitive Abhängigkeiten. Beispielsweise ist beim Relationenschema tblPatienten das Attribut PLZ vom Primärschlüssel Pat_Nr und das Attribut Wohnort von PLZ funktional abhängig. Die dritte Normalform wäre zu erreichen, wenn das Attribut PLZ zusätzlich in einer neuen Tabelle als Primärschlüssel fungiert und Wohnort in diese Tabelle verlagert wird, also aus der Tabelle tblPatienten entfernt würde.
Zugunsten einer höheren Zugriffsgeschwindigkeit wurde auf die Erreichung der dritten Normalform verzichtet. Alle weiteren Relationenschemata entsprechen zwar der dritten Normal-

form, dies ist aber nicht ausreichend, damit sich das gesamte Datenbankschema des PDAZN in dritter Normalform befindet.

4.2.2 Programm-Entwurf

Bisher wurde aufgrund der Informationsanalyse der Teil für den Entwurf der Datenbank diskutiert.
Die Aufgabe dieser Arbeit bestand aber auch darin, ein Front-End zu entwickeln, das dem Anwender, die Daten so benutzerfreundlich wie möglich präsentieren soll. Das Front-End stellt die Schnittstelle zwischen dem Anwender und dem Datenbanksystem dar. Deshalb hängt die Akzeptanz des Gesamtsystems im wesentlichen Maße auch vom Front-End ab.
Zu Beginn wird im Abschnitt 4.2.2.1 beschrieben, wie die Funktionalitätsanforderungen im PDAZN umgesetzt wurden. Es werden auch elementare Strukturierungen der Formulare vorgestellt, die die Schnittstelle zum Anwender darstellen.
Der sich anschließende Abschnitt 4.2.2.2 befaßt sich mit der Entwicklung des Datensicherheitskonzeptes, das aufgrund der analysierten Sicherheitsanforderungen entwickelt wurde.

4.2.2.1 Funktionalität des PDAZN

Aufgrund der funktionalen Analyse wurde ermittelt, welche Funktionen das PDAZN zur Verfügung stellen muß. Dazu wurden in Abschnitt 4.1.1.3 die notwendigen Aufgaben beschrieben. Aufgrund dieser Aufgabenstellungen wurde die Menüsteuerung entwickelt, um diesen Anforderungen gerecht zu werden. Die Beachtung der Zugriffsrechte bei unterschiedlichen Anwenderebenen, wird in einem gesonderten Abschnitt 4.2.2.2.1 beschrieben.
In diesem Abschnitt soll nicht der komplette Feinentwurf beschrieben werden, sondern nur ein Überblick über die Herangehensweise vermittelt werden. Aufgrund der notwendigen Aufgaben wurde zunächst untersucht, wie dem Anwender durch das Front-End diese Funktionen zur Verfügung gestellt werden kann. Der Abschnitt wird sich nachfolgend entsprechend den Hauptfunktionen in Patientendatenverwaltung, Allgemeine Datenpflege, Import von Patientendaten und Statistik gliedern.

4.2.2.1.1 Patientendatenverwaltung

Die Patientendatenverwaltung ermöglicht den Anwendern den Zugriff auf die Daten von Patientenaufnahmen. Je nach erlaubtem Zugriff können Aufnahmedaten gelesen, bearbeitet oder gelöscht werden. Die Abbildung 4.8 stellt die grobe Programmstruktur für die Patientendatenverwaltung dar, die aufgrund der Anforderungen entwickelt wurde. Die Kanten stellen die Navigationsmöglichkeiten zwischen den Komponenten dar. Nachfolgend werden die wesentliche Zusammenhänge erläutert.

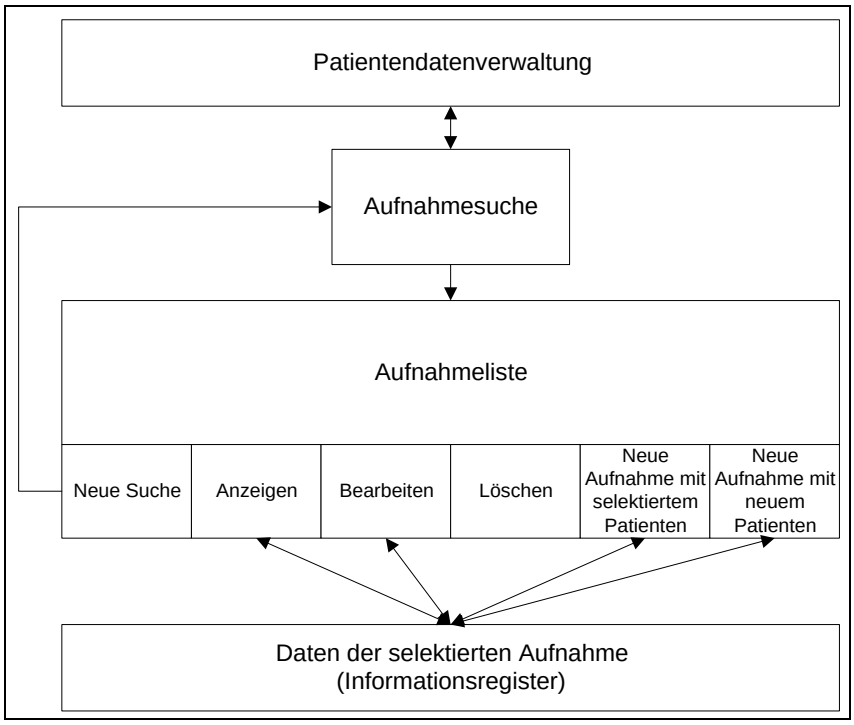


Abbildung 4.8: Patientendatenverwaltung - Ein Überblick über die Programmstruktur

Bei einer Client/Server-Anwendung ist es besonders wichtig darauf zu achten, daß dem Anwender nur die Daten angeboten werden, die ihn auch interessieren. Das ist deshalb wichtig, da im Gegensatz zu einer Desktop-Anwendung die Daten über das Netzwerk vom Server zum Client geschickt werden müssen. Deshalb ist die angeforderte Datenmenge so gering wie möglich zu halten. Aufgrund dessen, wird dem Anwender nicht gleich eine komplette Liste der Patientenaufnahmen angeboten, sondern er muß über ein Formular Suchkriterien eingeben. Die Kriterien nach denen gesucht werden sollen, sind in Abschnitt 4.1.1.3.1 dargestellt. In den meisten Fällen ist dabei die Angabe von Wildcards möglich, um die Suche komfortabel zu gestalten.
Die gefundenen Patientenenaufnahmen werden in Form von Listen präsentiert, aus denen eine konkrete Aufnahme zum Bearbeiten, Ansehen und Löschen selektiert werden kann. In der Liste stehen nur die Daten einer Aufnahme, die in Absprache mit dem Archiv ermittelt wurden und für die Selektion erforderlich sind. Zusätzlich wird auch die interne Patienten-Id mit in die Listendarstellung aufgenommen, um zweifelsfrei festzustellen, ob es sich bei zwei Aufnahmen, um die Aufnahmen ein und desselben Patienten handelt.
Die Informationen einer selektierte Aufnahme werden durch Informationsregister dargestellt. Diese wurden in Abschnitt 4.1.2 und 4.1.1.1 beschrieben. Die Abbildung 4.9 zeigt, wie die Informationsregister präsentiert werden. Die einzelnen Daten der Informationsregister sollen an dieser Stelle nicht erläutert werden. Dazu werden im Abschnitt 4.3.2 exemplarische

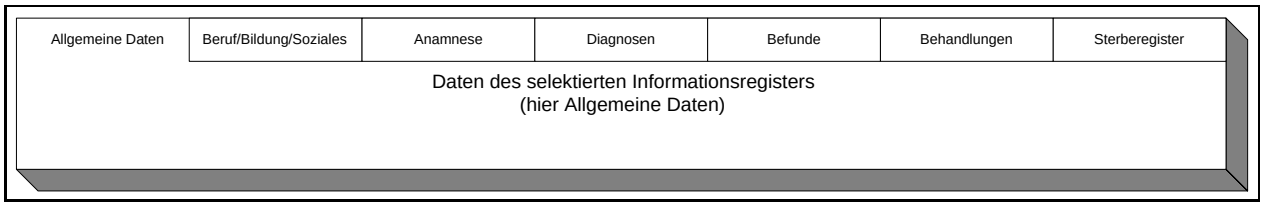


Abbildung 4.9: Informationsregister - Auswahl der Daten entsprechend ihrer inhaltlichen Zuordnung zu bestimmten Registern

Beispiele gegeben. Es ist bei der Präsentation sehr wichtig, nur die Daten vom SQL-Server anzufordern, die zum aktuellen Register gehören. Dadurch können die Ressourcen für andere Aufgaben freigehalten werden. Die Daten anderer Register werden immer erst dann angefordert, wenn das jeweilige Register selektiert wird.
Es muß weiterhin entsprechend der Analyse die Möglichkeit bestehen, Patientenaufnahmen manuell in der Datenbank anzulegen. Dazu können Patientendaten der neuen Aufnahme zum Suchen eingegeben werden, um festzustellen, ob dieser Patient schon einmal innerhalb des Zentrums für Nervenheilkunde aufgenommen wurde. Bei eindeutiger Identifizierung des Patienten wurde vereinbart, daß die neuen aufnahmeunabhängigen Patientendaten, die alten Daten in der Datenbank überschreiben. Die Identifizierung eines neuen Patienten in der Datenbank stellt sich als ein sehr schwieriges Problem mit weitreichenden Folgen dar. Das Identifizierungsproblem kann auch nicht in Abschnitt 4.2.2.1.3 beim Import von Patientendaten gelöst werden. Bestehen Zweifel bei der Identität oder ist der Patient ganz sicher noch nie aufgenommen worden, wird er als neuer Patient in der Datenbank gespeichert.

4.2.2.1.2 Allgemeine Datenpflege

Die Abbildung 4.10 stellt die Programmstruktur bezüglich der Allgemeinen Datenpflege dar. Wie im Datenbankentwurf in Abschnitt 4.2.1.1 dargestellt, wurden viele Informationen nicht durch die Attribute anderer Informationsobjekten dargestellt, sondern wurden als eigenständige Informationsobjekte modelliert, die dadurch anderen Objekten ebenfalls zur Verfügung stehen. Zum Beispiel wurde die Nationalität als Entity-Typ modelliert, dessen Entities durch andere Entities beliebig oft wiederverwendet werden können. Dadurch stieg zwar die Komplexität des Entwurfs deutlich, aber die Vorteile wie Redundanzvermeidung und dem daraus resultierenden einheitlichen Vokabular waren wichtiger. Ein weiterer Vorteil besteht darin, daß dem Anwender die Daten durch Listen angeboten werden können, wobei er dann nur noch per Mausklick den richtigen Wert auswählen muß.
Aufgrund dessen, daß das PDAZN sehr flexibel in Bezug auf die Erweiterbarkeit der Attribut-Wertebereiche sein soll, mußte die Möglichkeit geschaffen werde, die Inhalte solcher Tabellen zu pflegen. Die Allgemeine Datenpflege soll dabei diese Tabellen in einfacher Form zur Be-

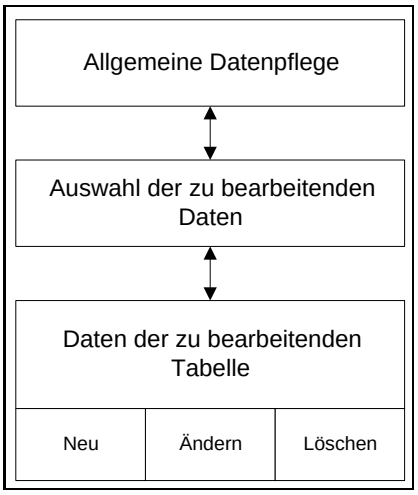


Abbildung 4.10: Allgemeine Datenpflege - Darstellung der Programmstruktur

arbeitung anbieten können.
Der Zugriff auf diese Daten wird direkt durch die Jet-Engine, also über die Methode der verknüpften Tabellen vorgenommen. Diese Methode, auf Daten des SQL-Servers zuzugreifen, wurde bereits in Abschnitt 3.1.5.1 erläutert.

4.2.2.1.3 Import von Patientenaufnahmen

In Abschnitt 4.1.1.3.3 wurde gezeigt, warum der Patientendaten-Import nicht vollständig automatisiert werden kann. Der Grund liegt Identifizierung der vorhandenen Patienten. Da diese Identifizierung durch die Archivmitarbeiter manuell durchgeführt werden muß, besteht die Notwendigkeit, diesen Prozeß entsprechend zu unterstützen. Die Abbildung 4.1.1 stellt die Programmstruktur für den Import von Patientenaufnahmen graphisch dar.
Die Patientenaufnahmen werden zunächst temporär in der Datenbank abgespeichert, wobei noch keine Zuordnung zu bereits gespeicherten Patienten des PDAZN erfolgt. Die Liste dieser temporären Aufnahmen wird dem Archivmitarbeiter angeboten, der jetzt in der Datenbank nach den Patienten sucht, die zu den temporären Aufnahmen gehören könnten. Findet der Archivmitarbeiter keine Hinweise darauf, daß der Patient schon einmal innerhalb des Zentrums für Nervenheilkunde aufgenommen wurde, kann er ihn als neuen Patienten importieren und auch alle weiteren Daten dieser Aufnahme bearbeiten. Werden dagegen mehrere Patienten gefunden, die zum neuen Patienten passen, muß der richtige Patient aufgrund des Vergleichs weiterer Daten ermittelt werden. Im Zweifelsfall soll ein Patient als neuer Patient betrachtet werden, auch wenn dadurch vielleicht eine redundante Abspeicherung erfolgt.

Nachfolgend wird kurz erläutert, wie die Patientenaufnahmen temporär in der Datenbank

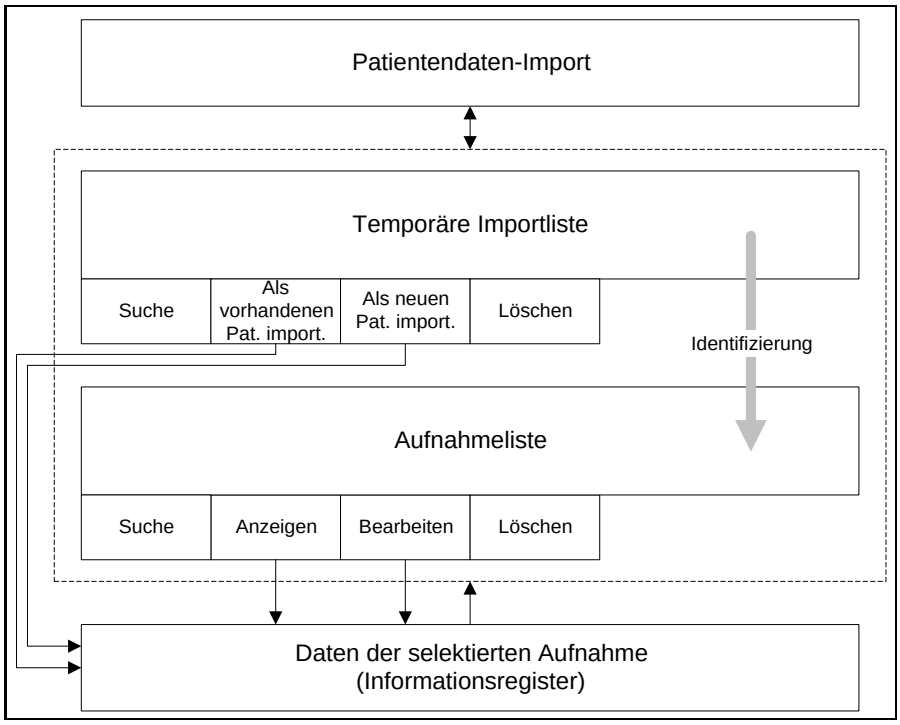


Abbildung 4.11: Patientendaten-Import - Entwickelte Programmstruktur für den Import von Patientendaten

abgelegt werden:
Dazu wurde mit dem DVSE vereinbart, daß bei bestimmten Ereignissen, wie Aufnahme, Entlassung und Verlegung eines Patienten, von einer Station auf eine andere innerhalb einer Klinik, die Daten in Form von Dateien auf dem Datenbankserver abgespeichert werden. Ein Teilprogramm sorgt dafür, daß die für das PDAZN notwendigen Daten aus den Informationen des DVSE entnommen werden und zur Abspeicherung in temporären Tabellen aufbereitet werden. Dabei mußte inbesondere die unterschiedliche Interpretation der Patientenaufnahme berücksichtigt werden (siehe Abschnitt 4.1).

4.2.2.1.4 Statistik

Da die Archivarbeit auch zu einem wesentlichen Teil aus der Beantwortung von Anfragen der Kliniken besteht, soll diese Aufgabe ebenfalls durch das PDAZN unterstützt werden.
Die Abbildung 4.12 stellt die Programmstruktur für den Statistikteil graphisch dar. In der Analysephase wurde zunächst spezifiziert, wie die Anfragen gestellt werden und welche dabei besonders häufig auftreten. Die dabei identifizierten Attribute wurden in Abschnitt 4.1.1.3.4 bereits vorgestellt.
Das Ergebnis dieser Anfrage besteht nicht nur aus Aufnahmelisten, die auch ausgedruckt

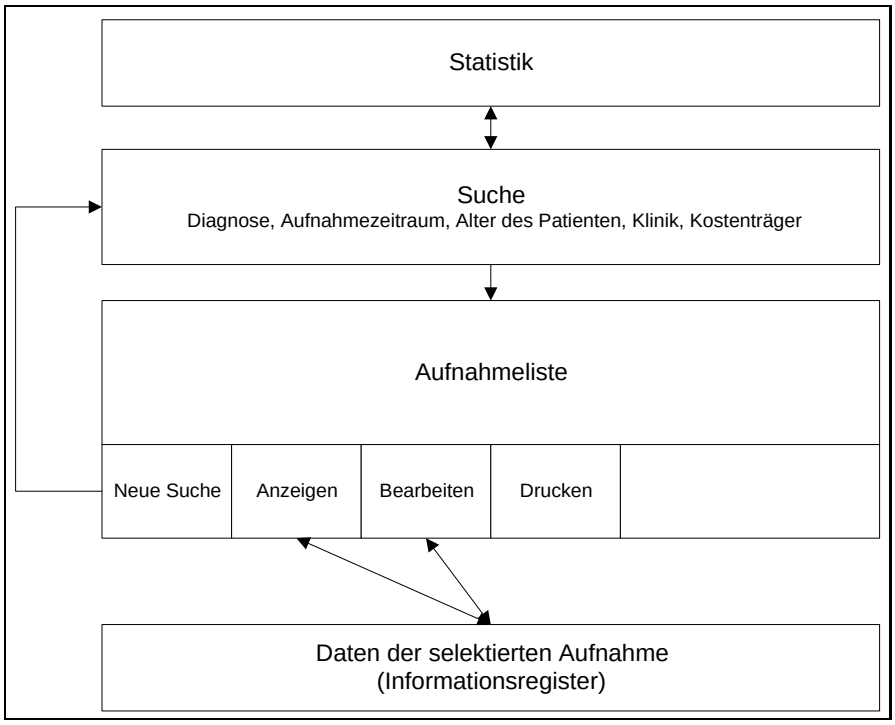


Abbildung 4.12: Statistik - Darstellung der Programmstruktur

werden können, sondern es können auch wiederum Informationen eines selektierten Patienten gesichtet und bearbeitet werden.
Es besteht der Wunsch auf medizinischer Seite, in einer Folgeversion des PDAZN die Daten der Patientenaufnahmen, zum Zweck der statistischen Auswertung durch spezielle Statistikprogramme, exportieren zu können.
Der Statistikteil des PDAZN soll insbesondere zur Beantwortung von häufig gestellten Anfragen dienen und nicht ein Statistikteil im klassischen Sinne darstellen, in dem zum Beispiel die Korrelation mehrere Informationen bei Massendaten untersucht wird.

4.2.2.2 Das Datensicherheitskonzept

Das nachfolgend beschriebene Konzept wurde aufgrund der geforderten Datensicherheit entwickelt.
Der Abschnitt 4.2.2.2.1 verdeutlicht, welche Zugriffsebenen beim PDAZN unterschieden werden müssen und welche Rechte die Anwender auf diesen Ebenen besitzen können. Der sich anschließende Abschnitt 4.2.2.2.2 beschreibt, wie die Zugriffsebenen durch Gruppenzugehörigkeiten der Anwender umgesetzt wurden.

4.2.2.2.1 Die Zugriffsebenen

Zunächst werden die wichtigsten Fakten noch einmal erwähnt, die beim Zugriff auf die Patientendaten zu beachten waren (siehe auch Abschnitt 4.1.2):
Grundsätzlich besitzen die Klinikangehörigen Lese- und Schreibrechte auf die Daten der Patientenaufnahmen in dieser Klinik. Beim Zugriff auf diese, durch Personen anderer Kliniken, dürfen keinen Schreibrechte vergeben werden. Die Personen anderer Kliniken haben höchstens Leserechte. Bestimmte Daten, die für die Archivierung der Patientenaufnahmen notwendig sind, werden nur von den Archivmitarbeitern verändert. Zu diesen Daten gehören die Informationsregister Allgemeine Daten und Diagnosen. Diese Informationsregister wurden in der Analyse, Abschnitt 4.1.1.1, bereits eingeführt, um die analysierten Attribute inhaltlich zu gruppieren. Das Informationsregister Sterberegister darf nur von Archivmitarbeitern gelesen und verändert werden.
Aufgrund der Analysedaten wurden für das PDAZN grundsätzlich drei Zugriffsebenen ermittelt:

- *Administrator-Ebene*
  Auf dieser Ebene des Zugriffs dürfen alle Daten gelesen und verändert werden. Alle Funktionen des PDAZN sind aufrufbar.
  Darüber hinaus können globale Einstellungen bis hin zu Programmveränderungen vorgenommen werden, da die Applikation nicht mit einer Run-Time-Version von Access entwickelt wurde. Die Abbildung 4.13 stellt die Zugriffsmöglichkeiten auf Administratorebene noch einmal graphisch dar.

- *Archiv-Ebene*
  Erfolgt ein Zugriff von dieser Ebene, können alle Daten gelesen und bearbeitet werden. Es gibt lediglich Einschränkungen bei der Pflege allgemeiner Daten. Die Einträge ganz bestimmter Tabellen, wie beispielsweise Kostenträger, Versicherungsstatus etc. dürfen nur von Administratoren verändert werden, da diese Daten beim Import kodiert sind und sichergestellt sein muß, daß die Bedeutung der Kodierung im DVSE und im PDAZN die gleiche ist. Die Abbildung 4.14 veranschaulicht die Zugriffsebene des Archivs.

- *Klinik-Ebene*

  Aus der Abbildung 4.15 wird deutlich, daß nur eine eingeschränkte Bearbeitung der Aufnahmedaten möglich ist. Der lesende Zugriff auf klinikfremde Daten darf nur mit Genehmigung des Klinikverantwortlichen erfolgen.
  Die Funktionen Allgemeine Datenpflege, Statistik und Import von Patientenaufnahmen dürfen nicht benutzt werden.

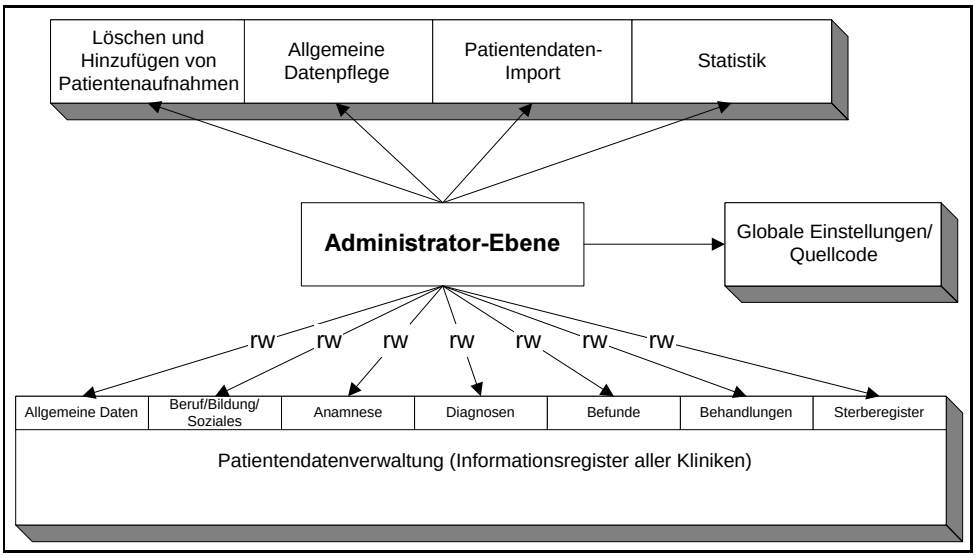


Abbildung 4.13: Rechte auf Administrator-Ebene - Administratoren haben umfassende Rechte

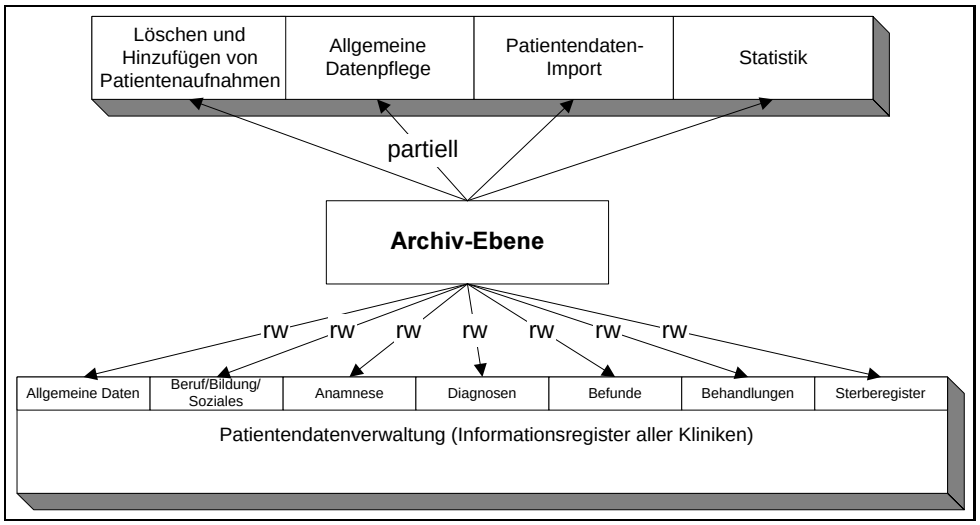


Abbildung 4.14: Rechte auf Archiv-Ebene - Archivmitarbeiter haben nur wenige Einschränkungen

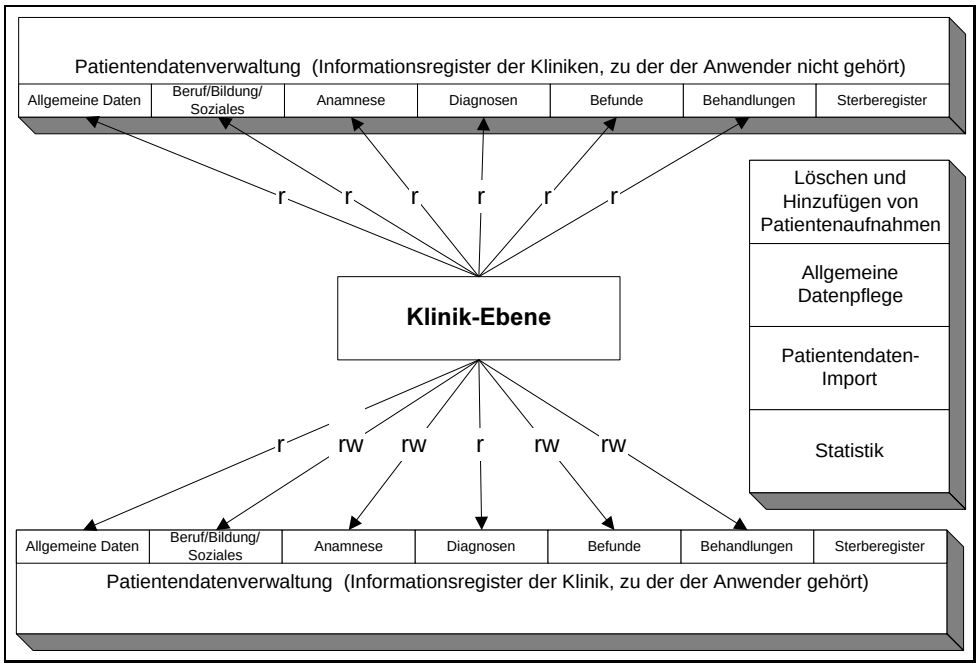


Abbildung 4.15: Rechte auf Klinik-Ebene - Klinikangehörige haben nur eingeschränkten Zugriff auf Daten und Funktionen

4.2.2.2.2 Umsetzung der Zugriffsebenen

Aufgrund der drei Zugriffsebenen mußte ein Konzept entwickelt werden, wie die dadurch unterschiedlichen Rechte der Anwender berücksichtigt werden können.
Im Abschnitt XXX wurde bereits erwähnt, daß für das PDAZN die integrierte Sicherheit, ein Sicherheitsmodus des SQL-Servers, gewählt wurde. Dazu ist es notwendig auf Betriebssystemebene drei Benutzergruppen anzulegen. Diese wurden namentlich wie folgt festgelegt:

Administrator-Ebene $\Longrightarrow$ SQLServerAdmins
Archiv-Ebene $\Longrightarrow$ PDAZNArchivGrp
Klinik-Ebene $\Longrightarrow$ PDAZNUserGrp

Alle Anwender, die Administrator-Rechte erhalten sollen, müssen der ersten Gruppe hinzugefügt werden. In Abschnitt 3.1.1.2 wurde bereits erläutert, daß sie dann das SA-Login des SQL-Servers benutzen und damit alle Rechte für die Administration besitzen. Außerdem besitzen die Mitglieder dieser Gruppe auch alle Rechte beim Datenbank-Front-End, um in Quellcode von Access Änderungen vorzunehmen. Alle Mitarbeiter mit Archivrechten müssen zur Gruppe PDAZNArchivGrp gehören. Die dritte Gruppe enthält alle Anwender die die Rechte auf Klinik-Ebene haben dürfen.
Mit diesen Gruppen besteht zunächst der Zugang zum SQL-Server. Noch besteht aber,

außer für Mitglieder der Gruppe SQL-ServerAdmins, kein Zugriff auf die Datenbank. Der SQL-Server bietet die Möglichkeit, für jede Datenbank gesonderte Benutzergruppen anzulegen. Dadurch wurde für die Datenbank "pdazn" eine Gruppe PDAZNArchivGrp angelegt, die alle Mitglieder enthält, die auf Betriebssystemebene eingetragen werden.
Für die Gruppe PDAZNUserGrp wurde eine Verfeinerung vorgenommen, damit die unterschiedlichen Rechte der Mitglieder innerhalb der Gruppe PDAZNUSerGrp ausgedrückt werden können. Die Verfeinerung wurde durch eine Aufspaltung in weitere 32 Gruppen erreicht. Diese Anzahl ist notwendig, da ein Anwender nur einer Gruppe zugeordnet werden darf. Aus diesem Grund bestand die Notwendigkeit, eine Syntax für den Gruppennamen zu entwickeln, die die Schreib- und Leserechte berücksichtigen kann. Von Vorteil war dabei, daß ein Anwender nur in einer Klinik ein Schreibrecht mit implizitem Leserecht besitzt und auf Daten anderer Kliniken maximal lesenden Zugriff haben darf. Der Syntax des Gruppennamens wird durch die in Abschnitt 4.2.1.1.1 eingeführten Synonyme gebildet. Die Synonyme werden in einer Zeichenfolge aneinandergereiht, wobei die Angabe der ersten Klinik, die Zugehörigkeit des Anwenders zu dieser Klinik darstellt und dadurch Schreib- und Leserechte besitzt. Die Angabe weiterer Synonyme ist optional und stellt die Leserechte bezüglich anderer Kliniken dar. Beispielsweise enthält die Gruppe K1 nur Anwender, die auf die Daten der Kinder- und Jugendklinik (K1) schreibend und lesend zugreifen dürfen, aber auf Daten anderer Kliniken keinen Zugriff haben. Ein anderes Beispiel K4K1K3 verdeutlicht, daß auf die Daten der Psychosomatik (K4) schreibend und lesend, aber auf Daten der Kliniken Kinder- und Jugend, sowie der Psychiatrie nur lesend zugegriffen werden darf. Nachfolgend werden alle gebildeten Gruppennamen genannt, wobei die Vereinigung der Gruppen die Gruppe PDAZNUserGrp bildet:

PDAZNUserGrp = K1 ∪ K2 ∪ K3 ∪ K4 ∪ K1K2 ∪ K1K3 ∪ K1K4 ∪ K2K3 ∪ K2K4 ∪ K3K4 ∪ K1K2K3 K1K2K4 ∪ K1K3K4 ∪ K2K3K4 ∪ K1K2K3K4 ∪ K2K1 ∪ K3K1 ∪ K4K1 ∪ K3K2 ∪ K4K2 ∪ K4K3 ∪ K2K1K3 ∪ K3K1K2 ∪ K2K1K4 ∪ K4K1K2 ∪ K3K1K4 ∪ K4K1K3 ∪ K3K2K4 ∪ K4K2K3 ∪ K2K1K3K4 ∪ K3K1K2K4 ∪ K4K1K2K3

Die Abbildung 4.16 stellt die Zuordnung von Gruppen auf Betriebssystemebene zur DBMS-Ebene graphisch dar. Bekommt ein Mitglied von PDAZNUserGrp beispielsweise andere Rechte, wird es aus der bisherigen Gruppe entfernt und der neuen Gruppe mit anderen Rechten zugeordnet.

4.3 Implementation

Die Implementation umfaßte einerseits die Datendefinition auf dem SQL-Server und andererseits die Implementierung der Applikationssteuerung. Deshalb wird die Implementation in zwei weitere Abschnitte untergliedert. Der erste Abschnitt 4.3.1 stellt wichtige Vorgehensweisen bei der Implementierung der Datenbank auf dem Back-End, dem SQL-Server dar.

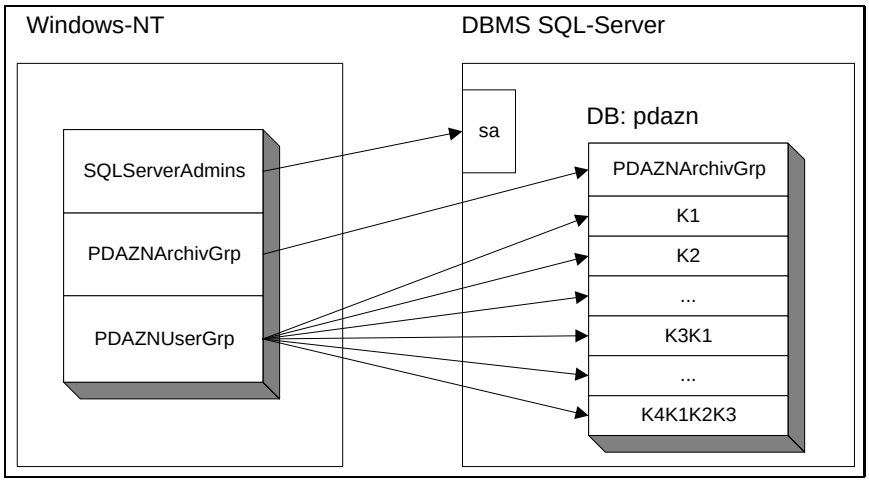


Abbildung 4.16: Gruppenzuordnung von Windows NT zum SQL-Server - Aufspalten der Gruppe PDAZNUserGrp zur Erkennung unterschiedlicher Rechte

Der Abschnitt 4.3.2 befaßt sich mit der Umsetzung der entwickelten Programmstruktur für das Front-End.

4.3.1 Back-End

Um eine Datendefintion durchführen zu können, wurde zunächst eine Datenbank namens "pdazn" angelegt.
Der SQL-Server stellt dazu ein Tool, den *SQL Enterprise Manager* zur Verfügung. Die wichtigsten Einstellungen lassen sich mit diesem Tool auf graphische Weise vornehmen. Bevor nun eine Datenbank angelegt werden kann, ist es notwendig ein Medium zu definieren, auf dem die Datenbank gespeichert werden kann. Dabei wurde beim PDAZN darauf geachtet, daß die Medien für die Datenbank selbst und dem Transaktionsprotokoll getrennt angelegt wurden. Dadurch kann das Transaktionsprotokoll separat gesichert werden. Mehr zu dem Thema Backup und Recovery erfolgt in Abschnitt 4.5.2.
Diese Medien wurden zum einen mit "pdazn_dat" und zum anderen mit "pdazn_log" benannt. Die Datenbank "pdazn" kann nun auf dem Medium "pdazn_dat" und ihr Transaktionsprotokoll auf "pdazn_log" angelegt werden.

Die nächste Aufgabe bestand darin, das in Abschnitt 4.2.2.2 erläuterte Sicherheitskonzept umzusetzen, in dem die dort beschriebenen Gruppen PDAZNArchivGrp, K1, K2, K3, ... als Gruppen der Datenbank "pdazn" angelegt wurden. Anwender, die einen Account auf Betriebssystemebene haben, müssen entsprechend ihrer Rechte in eine der drei Gruppen SQLServerAdmins, PDAZNArchivGrp und PDAZNUSerGrp durch den Administrator eingetragen werden. Ein weiteres Tool des SQL-Servers, der *SQL Security Manager* ist für

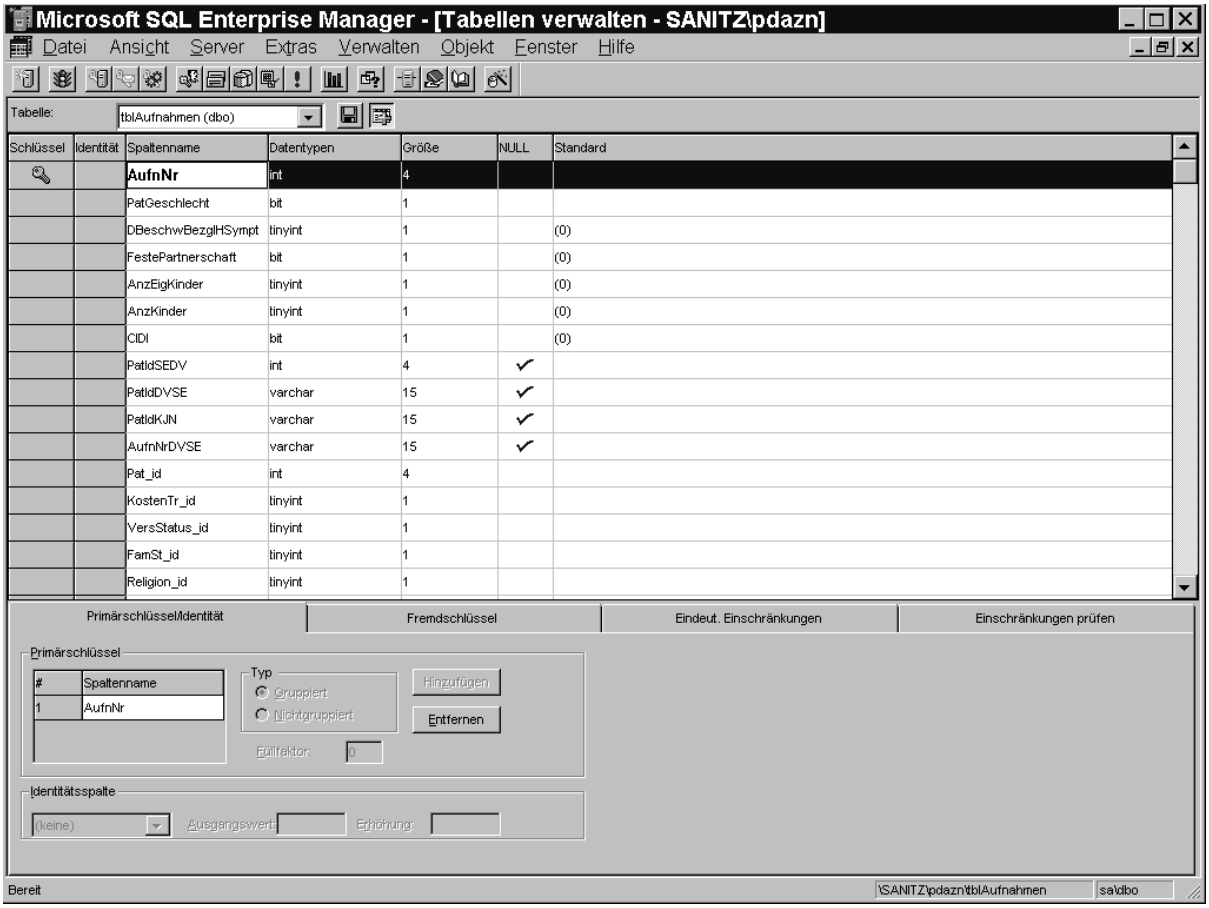


Abbildung 4.17: Datendefinition für tblAufnahmen mit Hilfe des SQL Enterprise Managers

das Eintragen dieser Anwender auf SQL-Server-Seite hilfreich. Dadurch ist es möglich, eine Gruppe des Betriebssystems, zu einer Gruppe der konkreten Datenbank "pdazn" zuzuordnen. Beide Gruppen tragen in diesem Fall den gleichen Namen PDAZNArchivGrp. Die Gruppe SQLServerAdmins benötigt keine Zuordnung, da sie direkt als Administratorgruppe bestimmt wird. Da auf Betriebssystemebene keine 32 Gruppen angelegt werden sollen, wurde die Gruppe PDAZNUserGrp angelegt. Die Mitglieder dieser Gruppe müssen einzeln der entsprechenden Zielgruppe K1, K2, etc. des SQL-Servers zugeordnet werden.

Die Datendefinion wurde unter Zuhilfenahme des SQL Enterprise Managers vorgenommen wurde. Eine andere Möglichkeit besteht darin, die Datendefinition mit Transact-SQL, einem um prozedurale Komponenten erweiterten SQL-Dialekt des SQL-Servers, vorzunehmen.
In Abschnitt 4.2.1.2.1 wurden alle notwendigen Informationen zusammengestellt, die für die Definition notwendig sind. Da der SQL-Server den Standard SQL92 unterstützt, können Primärschlüssel, Fremdschlussel, Standard-Werte angegeben werden. Dazu gehört auch die

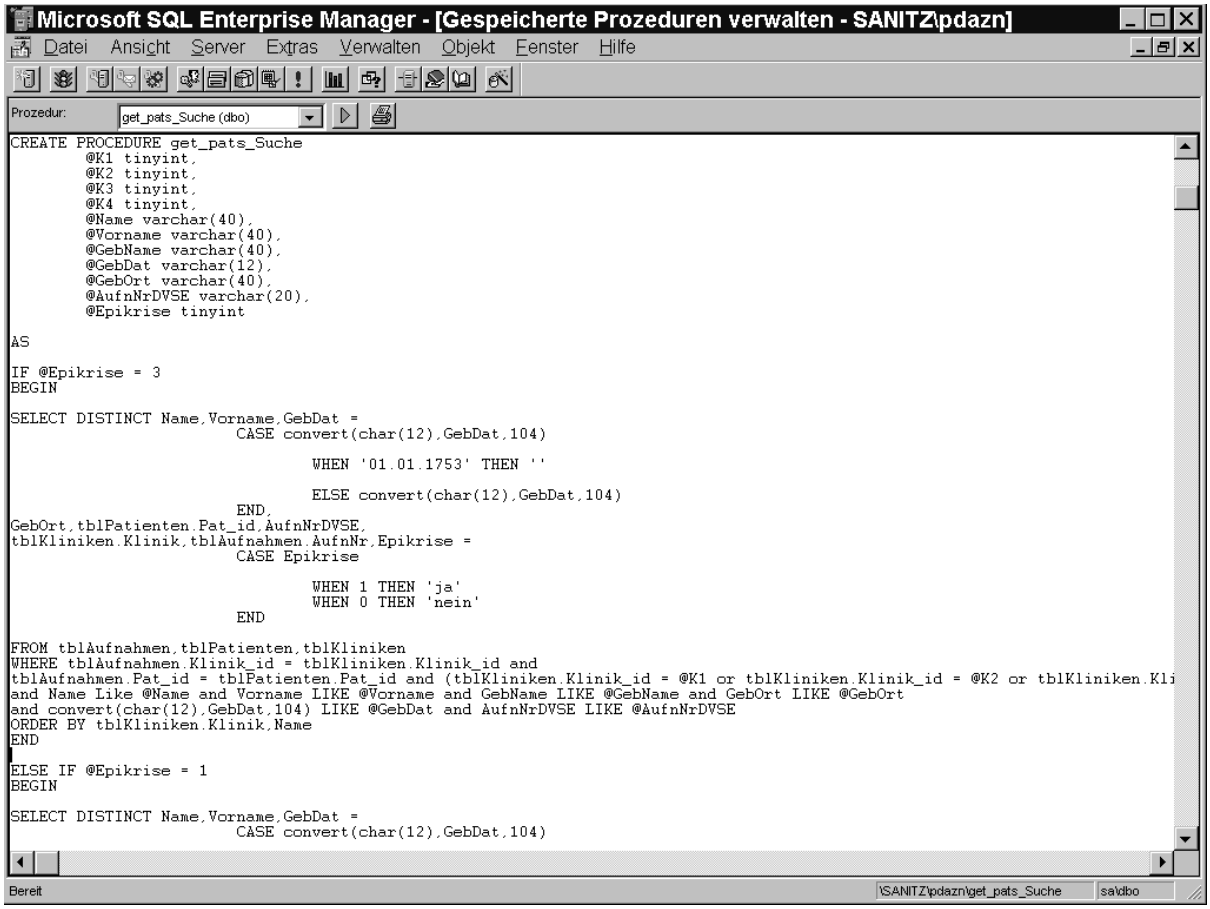


Abbildung 4.18: Definition einer Stored Procedure - Ausschnitt aus der Definition von get_pats_Suche

Angaben über die NULL-Wert-Zulässigkeit der Attribute, sowie die Definition von Check-Klauseln. Die Abbildung 4.17 zeigt einen Ausschnitt der Datendefinion für tblAufnahmen. Dabei werden im unteren Teil die weiteren Angabemöglichkeiten wie Primärschlussel, Fremdschlüssel etc. deutlich.
In Abschnitt 2.2 wurde erläutert, daß das PDAZN in die Client/Server-Architektur "Verteilte Appliaktions-Logik" einzuordnen ist. Die Möglichkeit der Verteilung der Funktionen besteht insbesondere durch die Verwendung der Stored Procedures, die in Abschnitt 3.1.1.4 diskutiert wurden. Die Stored Procedures werden vom Enterprise Manager verwaltet, wobei die eigentliche Definition nicht unterstützt wird. Die Datenzugriffe der Applikation, die in Access implementiert wurde, werden fast ausschließlich über Stored Procedures vorgenommen, um ein Höchstmaß an Zugriffsgeschwindigkeit zu erzielen. Die Abbildung 4.18 stellt einen Teil der Definition der Stored Procedure get_pats_Suche dar. Die Implementation erfolgt mit Transact-SQL, wobei ca. 100 solcher Stored Procedures implementiert wurden.

Beim Aufruf dieser Prozedur werden Parameter für die Suche von Patientenaufnahmen übergeben. Dabei wird auch deutlich, wie gespeicherte Systemprozeduren, wie Convert(...) benutzt werden können.

4.3.2 Front-End

Der größte Teil der Implementation wurde auf dem Front-End vorgenommen, daß mit MS Access 97 entwickelt wurde.
Access stellt standardmäßig umfangreiche Steuerelemente zur Verfügung, die in Formularen eingebettet werden. Bestimmte Steuerelemente wie Button, Listen etc. lösen unter bestimmten Umständen, wie Mausklick etc., bestimmte Ereignisse aus. Diesen Ereignissen können Aktionen zugeordnet werden, in dem zum Beispiel ein Makroaufruf erfolgt. Eine andere Möglichkeit besteht in der Implementierung von Aktionen durch Visual Basic for Applikation, kurz VBA. Für das PDAZN wurde fast ausschließlich die Implementierung durch VBA vorgenommen. In den folgenden Abschnitten werden nicht nur Implementationsbeispiele, sondern auch deren Ergebnisse dargestellt werden.
Der Abschnitt 4.3.2.1 zeigt, wie die Überprüfung der Zugangsrechte unter Zuhilfenahme des DMO- und des DAO-Modells erfolgt. Die weiteren Abschnitte beschreiben, wie die ermittelten Funktionalität, behandelt in Abschnitt 4.2.2.1, implementiert wurde.

4.3.2.1 Überprüfung der Zugangsrechte

Beim Start der Applikation müssen die Rechte des angemeldeten Benutzers festgestellt werden. Dazu wird zunächst überprüft, ob der Anwender eine Login-ID beim SQL-Server hat. Ist dieser Test erfolgreich wird überprüft, ob auch ein Zugangsrecht zur Datenbank "pdazn" besteht. Trifft das nicht zu, wird der Anwender nicht für die Nutzung zugelassen.
Im folgenden werden wichtige Prinzipien des Datenzugriffs, was sowohl die Einbeziehung des DMO-Modells, als auch die Verwendung des DAO-Modells betrifft, am Beispiel des Zugangstests mit der anschließenden Ermittlung der Gruppenzugehörigkeit gezeigt. Dabei erfolgt auszugsweise und in vereinfachter Form eine Darstellung von Quellcode-Teilen aus dem Modul Zugriffssteuerung der Applikation.
Der Zugangstest erfolgt mit Hilfe von Objektinstanzen des SQL-Server-DMO-Modells. Das Modell wurde bereits in Abschnitt 3.1.1.3 durch die Abbildung 3.1 dargestellt.
Die Instanzen werden von Objekten des DMO-Modells in Access wie folgt erzeugt:

Set objSQLsrv = CreateObject("sqlole.sqlserver")

Im konkreten Fall wird eine Instanz vom Objekt SQLServer erzeugt. Anschließend muß mit der Methode Connect eine Verbindung zum SQL-Server aufgebaut werden, die unter Angabe des Servernamens erfolgt:

objSQLsrv.Connect strServerName

Der Server-Name unterscheidet sich je nach Plattform und heißt im Archiv ZNARCHIV. Schlägt diese Methode fehl, existiert keine LOGIN-ID beim SQLServer oder der SQL-Server wurde heruntergefahren. Bei erfolgreichem Verbindungsaufbau muß nun noch überprüft werden, ob der angemeldete Benutzer, auch einen Zugriff auf die Datenbank "pdazn " haben darf. Dazu wird die Eigenschaft UserName des Objektes Databases erfragt, daß in der Hierarchie direkt unter dem Objekt SQLServer steht:

currentUser = objSQLsrv.Databases("pdazn").UserName

Der nachfolgende Test stellt fest, ob dieser Nutzer SA-Rechte, also Administrator-Rechte besitzt:

SaloginTest = objSQLsrv.SaLogin

Im Anschluß wird ermittelt, welcher Gruppe der Nutzer innerhalb der Datenbank pdazn angehört. Da im DMO-Modell keine Eigenschaft mit dieser Angabe existiert, wurde mit Hilfe des DAO-Modells von Access die Gruppenzugehörigkeit überprüft. Dazu wird zunächst eine Instanz der aktuelle Datenbank erzeugt:

Set dbsCurrent = CurrentDb

Mit dieser Voraussetzung kann eine, in diesem Falle temporäre Anfragedefinition, erfolgen:

Set qdfGruppen = dbsCurrent.CreateQueryDef("")

Anschließend wird durch die Connect-Eigenschaft der Datenbanktyp und weitere Parameter angegeben. Die weiteren Parameter sind in der ODBC-Datenquelle gespeichert. Wenn der anschließende Verbindungsaufbau fehlschlägt, fehlt möglicherweise die Definition der Datenquelle, die in Abschnitt 3.1.4.1 gezeigt wurde:

qdfGruppen.Connect = "ODBC;DSN=systempdazn"

Die Ermittlung des Gruppennamens erfolgt dadurch, daß in der Systemtabelle "sysusers" des SQL-Servers nachgesehen wird, ob es eine Zuordnung des aktuellen Benutzernamens zu einer Gruppen-ID (gid) gibt. Durch die Gleichheit von uid und gid wird festgestellt, ob es sich in dieser Tabelle um eine Gruppe handelt:

qdfGruppen.SQL = "SELECT name FROM sysusers where uid=gid and gid in (select gid from sysusers where name = '"& currentUser & "')"

Die eigentliche Ausführung der Abfrage erfolgt mit dem Aufruf der Methode OpenRecordset() des Querydef-Objektes. Dabei wird das Ergebnis an das sogenannte RecordSet-Objekt übergeben:

Set rstGruppen = qdfGruppen.OpenRecordset()

Über die Eigenschaft Name des RecordSet-Objektes kann nun der Gruppenname ermittelt werden:

ServerGrp = rstGruppen!Name

Aufgrund der ermittelten Gruppe, werden jetzt die Rechte des Anwenders festgestellt, die in Abschnitt 4.2.2.2.2, entsprechend der Syntax zugeordnet wurden. Außer bei der Gruppe PDAZNArchivGrp, werden durch die Gruppennamen Schreib- und Leserechte bezüglich der Klinikdaten abgeleitet. Diese Rechte werden dem Anwender im Hauptmenü gezeigt.

Die Abbildung 4.19 zeigt das Hauptmenü des PDAZN, wenn sich der Administrator anmeldet. Das wird durch alle Schreib- und Leserechte deutlich. Ebenso sind keine Funktionen inaktiviert. Das Hauptmenü stellt die Auswahl zwischen den ermittelten Funktionen dar. Die Funktionen auf denen der Anwender keinen Zugriff hat, werden durch inaktivierte Button dargestellt. Die Menüleiste stellt allgemeine Funktionen bereit, wie Bearbeiten, Datei, aber auch Einstellmöglichkeiten für den Administrator, der dadurch in den Entwurfsmodus wechseln und damit Änderungen am Quelltext vornehmen kann. Entsprechend der geforderten Funktionalität, werden die einzelnen Funktionen, wie Patientendatenverwaltung, Allgemeine Datenpflege, Importieren von Patientendaten und die Statistik ausgewählten Beispielen dargestellt werden.

4.3.2.2 Patientendatenverwaltung

In Abschnitt 4.2.2.1.1 wurde die Entwicklung der Programmstruktur für die Patientdatenverwaltung vorgestellt. Aufgrund dieser Struktur wurde ein Suchformular, entsprechend der Abbildung 4.20 entwickelt. Dabei werden Wildcards, wie dargestellt, als default-Werte bei den Attributen eingetragen. Die Voreinstellung des Suchformulars findet dadurch alle Patientenaufnahmen des PDAZN. Die konkreten Attribute bzw. Optionsfelder sind weitgehend selbsterklärend.
Das Ergebnis der Suche wird als Aufnahmeliste im Formular Patientdatenverwaltung, Abbildung 4.21, dargestellt, das alle Funktionen für die Verwaltung von Patientenaufnahmen beinhaltet. Die Aufnahmen einer Liste können nun per Mausklick selektiert werden, um sie

Abbildung 4.19: Hauptmenü des PDAZN am Beispiel eines angemeldeten Administrators

dann anzuzeigen, zu bearbeiten, zu löschen oder aufgrund der zugehörigen Patientendaten eine neue Aufnahme manuell anzulegen. Außerdem ist eine manuelle Neuaufnahme auch ohne Selektion eines bereits vorhandenen Patienten möglich. Weiterhin stehen über dem Listenkopf zusätzlich die Sucheinstellungen.
Stellvertretend für die anderen Kliniken erfolgt für die Neurologie eine Darstellung des Formulars in Abbildung 4.22, das dem Anwender nach Selektion einer Patientaufnahme im Modus Bearbeiten geboten wird. Es gibt bei den Informationsregistern Allgemeine Daten und Diagnosen geringfügige Unterschiede zwischen den Kliniken. Große Unterschiede gibt es dagegen bei den klinikspezifischen Informationsregistern. Die Abbildung 4.22 zeigt, wie die, in Abschnitt 4.2.2.1.1 eingeführten Informationsregister dargestellt werden. Dazu stellt MS Access ab der Version 97 das Steuerelement Register zur Verfügung. Die Informationsregister werden aus Gründen der Datenzugriffsgeschwindigkeit in zwei Modi angeboten. Im Modus Anzeigen werden die Informationsregister vollständig durch das Steuerelement Register dargestellt. Dadurch werden alle Daten zugleich vom Server angefordert. Im Modus Bearbeiten, in Abbildung 4.23, wird dagegen nur das aktuelle Informationsregister mit dem

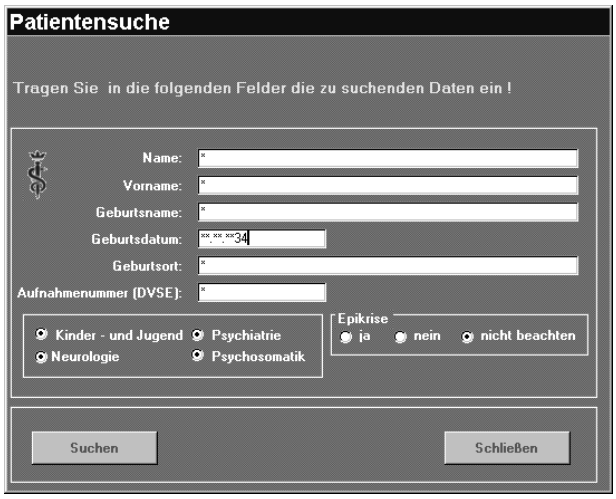


Abbildung 4.20: Formular Patientensuche

Abbildung 4.21: Formular zur Patientendatenverwaltung

Abbildung 4.22: Formular zur Anzeige einer Aufnahme - Beispiel für das Informationsregister Behandlungen der Neurologie

Steuerelement Register angezeigt. Die anderen Informationsregister werden, anlehnend an das Register-Layout des Modus Anzeigen, als Button dargestellt. Die Ursache liegt darin, nur die Daten vom Server zu laden, die auch wirklich benötigt werden. Daten die nur zum Anzeigen benötigt werden, stellen kein Problem bezüglich der Zugriffsgeschwindigkeit dar. Im Modus Bearbeiten sollen dem Anwender jedoch auch sogenannte Kombinationsfelder angeboten werden. Diese Kombinationsfelder, das sind die Felder mit dem nach unten gerichteten Dreieck, enthalten die Werte, die als Eintrag für die entsprechende Information erlaubt sind. Durch die Verwendung vieler Kombinationsfelder, kann die Ausführungsgeschwindigkeit sinken. Deshalb werden die Daten der anderen Informationsregister erst nach Betätigung der jeweiligen Button vom Server angefordert. Ein kleines Beispiel verdeutlicht, wie die Daten bereitgestellt werden:

Bei einem Klick auf den Versicherungsstatus würde beispielsweise eine Liste mit den Ein-

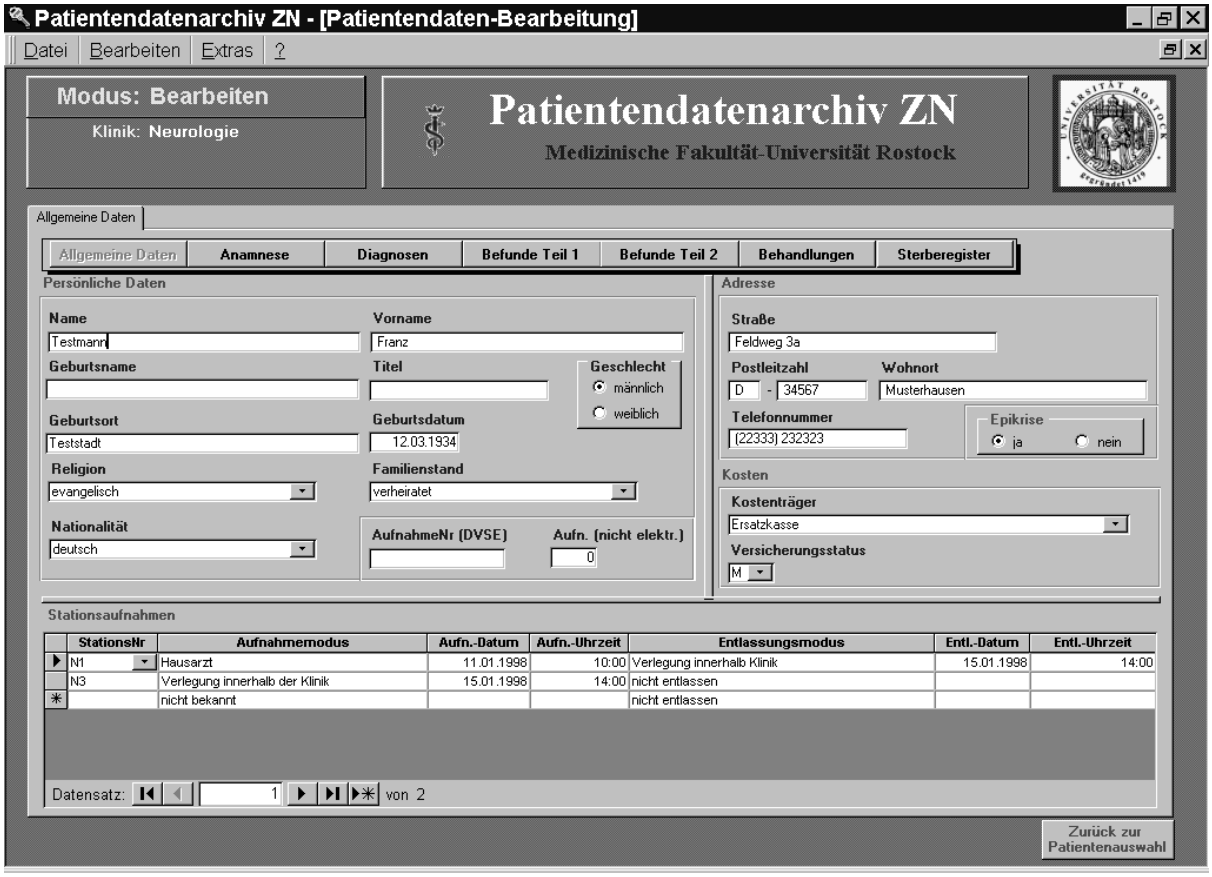


Abbildung 4.23: Formular zur Bearbeitung einer Aufnahme - Beispiel für das Informationsregister "Allgemeine Daten"der Neurologie

trägen[3] F, M, R und U erscheinen. Diese Werte werden durch die Tabelle tblVersStatus auf dem SQL-Server gespeichert. Auf dem Front-End werden sie als verknüpfte Tabellen benutzt (siehe Abschnitt 3.1.5.1). Die Daten eines Informationsregisters werden durch den Aufruf einer Stored Procedure vom SQL-Server zum Front-End übermittelt. Dabei werden die Daten für die Kombinationsfelder in abstrakter Form, durch ihre ID's dargestellt. Diesen ID's werden beim Front-End die zugehörigen Werte aus der verknüpften Tabelle zugeordnet. Beispielsweise wird zunächst die VersStatus_id übermittelt und dann wird auf dem Front-End in der verknüpften Tabelle dbo_tblVersStatus die Zuordnung zur eigentlichen Bezeichnung vorgenommen und repräsentiert.
Da in der Anwendung sehr viele Kombinationsfelder notwendig waren, kann bei höherer Netzbelastung ein Geschwindigkeitsverlust auftreten, dem die Aufsplittung der Informationsregister auf einzelne Formulare entgegen wirken soll. Dadurch werden weitaus weniger

[3]Verwendete Abkürzungen: F-Familie, M- Mitglied, R-Rentner, U-Unbekannt

Abbildung 4.24: Formular zur Auswahl der Datenpflege - Auswahl der zu ändernden Tabelle

Daten angefordert. Im Modus Bearbeiten waren deshalb 27 Formulare und im Modus Anzeigen 4 Formulare notwendig. Auf die Darstellung aller weiteren Formulare soll an dieser Stelle verzichtet werden.

4.3.2.3 Allgemeine Datenpflege

Für diese Funktion wurde ein Formular entworfen, daß die zu ändernden Tabellen in unterschiedlichen Registern, entsprechend ihres Vorkommens in den Informationsregistern, einteilt. Die Abbildung 4.24 zeigt das geöffnete Register Allgemeine Daten. Den einzelnen auszuwählenden Tabellen werden Kliniknamen hinzugefügt, um darzustellen, durch welche Kliniken die Tabellen genutzt werden. Durch Betätigung der Button wird dem Anwender jeweils die Tabelle mit den zu ändernden Daten angeboten. Vollständigkeitshalber zeigt die Abbildung 4.25 ein Beispiel für die Bearbeitungsmöglichkeit der Tabelle Religion, deren Inhalt beim Bearbeiten des Informationsregisters Allgemeine Daten in Form von Kombinationsfeldern angeboten wird.

Abbildung 4.25: Formular zur Datenpflege des Religion-Wertebereiches

4.3.2.4 Import von Patientendaten

Für die Struktur dieser Funktion wurde bereits in Abschnitt 4.2.2.1.3 eine Übersicht gegeben.
Demzufolge wurden nun Formulare entwickelt, die diese Funktionen enthalten. Die Abbildung 4.26 zeigt das Formular Patientendaten-Import. Es gliedert sich in den oberen Teil für die zu importierenden Aufnahmen und in den unteren Teil für die gespeicherten Aufnahmen. Wird dieses Formular geöffnet, werden alle zu importierenden Aufnahmen aufgelistet. Diese Aufnahmen sind in dieser Version mit vollständigen Daten versehen, daß heißt, sie haben alle ein Entlassungsdatum. Der Hintergrund liegt darin, daß keine Aufnahmen importiert werden sollten, die unvollständig sind, da dadurch der eigentliche Import erschwert wird. Werden nämlich alle Ereignisse, wie Aufnahme, Verlegung auf eine andere Station und Entlassung einzeln mit den entsprechenden Daten importiert, muß jedesmal eine Patientenidentifikation erfolgen. Beim Öffnen des Formulars werden noch keine bestehenden Aufnahmen angezeigt. Aufgrund der zu importierenden Aufnahmedaten, kann der Anwender in den vorhandenen Aufnahmen nach den Patienten suchen, die zur neuen Aufnahme passen könnten.
Wurde ein Patient gefunden, der zweifelsfrei zur neuen Aufnahme paßt, müssen in der oberen Liste die zu importierende Aufnahme und in der unteren Liste die Aufnahme markiert werden, zu dessen Patienten die neue Aufnahme gehören soll. Erst dann wird der Button *Als vorh. Pat. importieren* aktiviert dargestellt. Im Beispiel wird angenommen, daß der Patient schon einmal aufgenommen wurde, auch wenn sein Vorname bei der zu importierenden Aufnahme jetzt vielleicht vollständigerweise mit Franz-Peter angegeben wurde.
Vereinbarungsgemäß werden die alten aufnahmeunabhängigen Patientendaten durch die neuen überschrieben.
Wenn kein Patient gefunden wurde, oder Zweifel an der Zuordnung bezüglich eines in der Datenbank gespeicherten Patienten bestehen, wird die Aufnahme als neue Aufnahme importiert.
Werden die Button *Als vorh. Pat. importieren* oder *Als neuen Pat. importieren* betätigt,

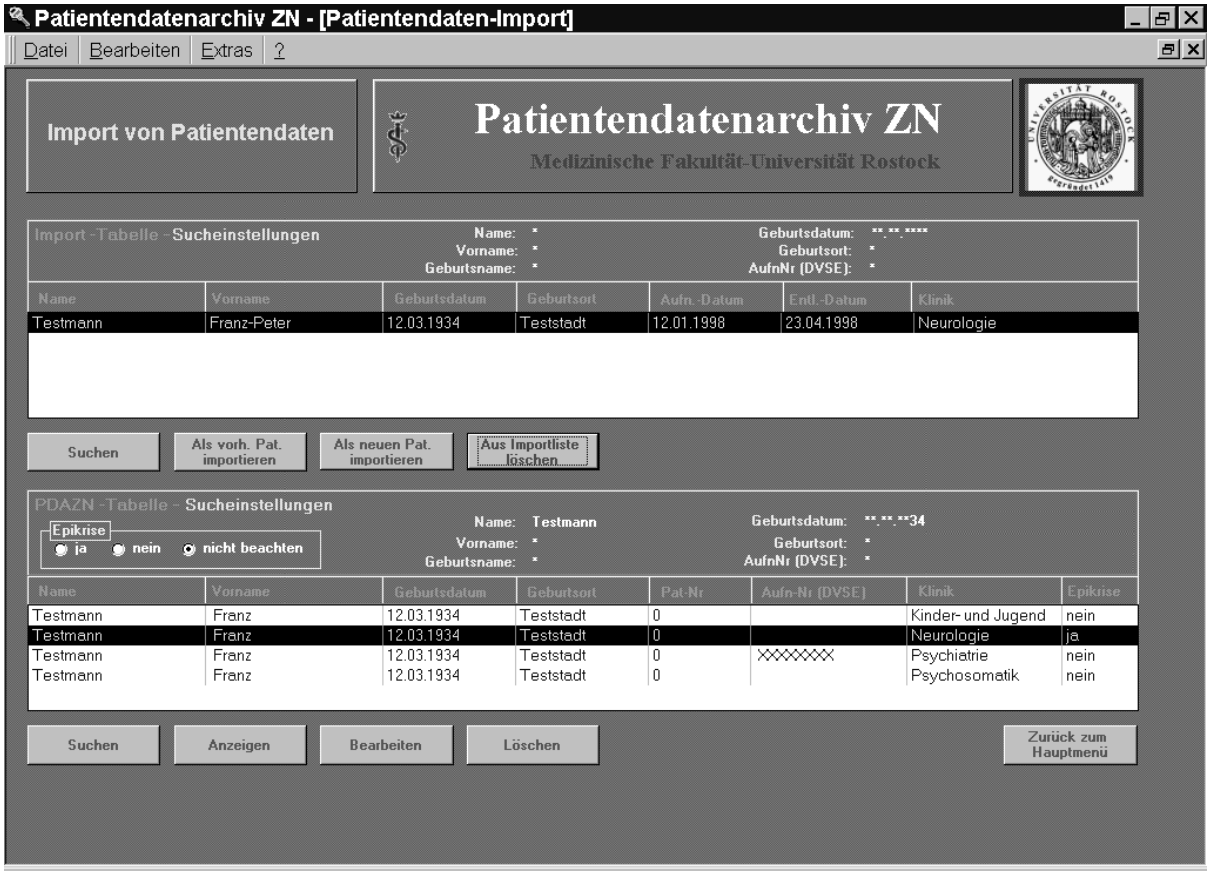


Abbildung 4.26: Formular zum Patientendaten-Import

kann eine weitere Bearbeitung der importierten Aufnahme, mit den bereits beschriebenen Formularen, zur Bearbeitung der Informationsregister, erfolgen.

4.3.2.5 Statistik

Die Statistik umfaßt die Unterstützung der häufigsten Anfragen an das Archiv. Für die in Abschnitt 4.1.1.3.4 analysierten Daten wurde das Formular entsprechend der Abbildung 4.27 entwickelt. Hier hat der Anwender einfache Auswahlmöglichkeiten, die eine Vielzahl von unterschiedlichen Anfragen erlauben. Die einleitend in Abschnitt 1.2 genannte Anfrage,

Geben Sie mir bitte alle Patienten im Alter von 60-70 Jahren, die in dem Zeitraum X mit der Diagnose Y des Diagnosesystems Z stationär aufgenommen wurden!

läßt sich auf sehr einfache Art und Weise durch dieses Formular zusammenstellen. Der Kostenträger wurde im Beispiel durch die Angabe Erstzkasse eingeschränkt.

Abbildung 4.27: Formular zur Statistik - Definition der Anfrage

Das Ergebnis der Anfrage wird als Aufnahmeliste dargestellt, die nicht nur eine Aufzählung sein soll, sondern einzelne Aufnahmen können auch hier sofort bearbeitet oder angesehen werden. Zusätzlich kann diese Liste auch gedruckt werden. Die Abbildung 4.28 zeigt das Ergebnis der in Abbildung 4.27 gestellten Anfrage.

4.4 Test der Anwendung

Bevor die Applikation vollständig die bisherigen Aufnahmebücher etc. ersetzen darf, müssen umfangreiche Tests stattfinden, was insbesondere die Datensicherheit betrifft. Diese Tests wurden im wesentlichen auf der Entwicklungsplattform vorgenommen, da bezüglich der Anwenderplattform keine Unterschiede zu erwarten sind. Getestet wurde insbesondere die Erhaltung der Datenkonsistenz, beispielsweise bei nicht regulärer Beendigung der Applikation, wie Stromausfall, Beenden der Applikation mit dem Task-Manager etc..
Insbesondere wurden umfangreiche Tests bezüglich des Datenaustausches bei Datums-Werten

Abbildung 4.28: Formular zum Statistikergebnis

durchgeführt. Access hat die Eigenschaft, ein Datum, das zwischen dem 01.01.1930 und dem 31.12.1999 liegt, ohne Jahrtausend- und Jahrhundertangabe darzustellen. Eine Datumsangabe von 01.01.1945 bewirkt aber beim SQL-Server eine Darstellung von 01.01.2045, da Access das Datum als 01.01.45 weitergibt. Durch eine geeignete Maßnahme, wird dem SQL-Server das Datum in korrekter Form übergeben. Sie beruht auf den Langenvergleich des Access-Datums und erweitert es bei Notwendigkeit, um die Jahrtausend- und Jahrhundertangabe. Andere Tests bezogen sich auf das Geschwindigkeitsverhalten der Anwendung, um die optimale Implementierungsstrategie zu erhalten. Zu diesem Zweck wurde die Applikation auf die Anwendungsplattform portiert, um dort Testdatensätze zu generieren. Für den Test wurden 50000 Aufnahmedatensätze generiert, um zu überprüfen, wie die Zugriffsgeschwindigkeit des Clients ausfällt. Gerade hier liegt der Schwerpunkt der Client/Server-Anwendung, in dem nur die Daten übermittelt werden, die notwendig sind oder auch die Nutzung der Stored Procedures. Die durchnittliche Anzahl der Patientenaufnahmen beträgt ca. 3000 pro Jahr, wobei zum Beispiel im letzten Jahr ca. 3200 Aufnahmen archiviert wurden. Die Anwendungsplattform, speziell der Server mit einem Pentium Pro 200Mhz und 64 MByte RAM,

ist um Größenordnungen besser ausgestattet als die Entwicklungsplattform und zeigte daher auch wesentlich bessere Ergebnisse als diese, die mit einem Pentium 133 MHz bestückt ist und 32 MByte RAM hatte. Diese Ausstattung ist für die Funktion als Windows NT Server, SQL-Server und Client-Anwendung des PDAZN nur als ausreichend zu bezeichnen. Die Zugriffsgeschwindigkeit war auf der Anwendungsplattform sehr zufriedenstellend, dagegen auf der Entwicklungsplattform gerade noch akzeptabel. Der Grund lag dabei eindeutig in der zu geringen Hauptspeicherausstattung.
Selbstverständlich hängt das Testergebnis von zahlreichen weiteren Faktoren ab. Dazu gehören nach [ChGu+96] im wesentlichen folgende:

- Komplexität der Abfrage
- Geschwindigkeit des Servers und der Clients
- Hauptspeicherausstattung
- Größe der Serverdatenbank
- Netzwerktopologie
- Anzahl der Netzbenutzer

Insbesondere kann hier die Anzahl der Netzbenutzer zu Verzerrungen der Testergebnisse führen.

Seit dem 04. Mai 1998 befindet sich die Anwendung in einer viermonatigen Testphase, die eventuelle Unzulänglichkeiten aufdecken soll. Dazu wird die Applikation bereits von den Archivmitarbeiterinnen benutzt und parallel dazu, die konventionellen Aufnahmebücher geführt. Auftretende Fehler oder Fragen werden dabei notiert, in bestimmten Abständen besprochen und, wenn notwendig, beseitigt.

4.5 Wartung

Die Wartung ist eine Aufgabe des Datenbankadministrators. In Abschnitt 4.5.1 werden allgemeine Aufgaben erläutert. Auf die Besonderheiten beim Backup und Recovery, wird im Abschnitt 4.5.2 hingewiesen.

4.5.1 Allgemeine Aufgaben

Die Wartung der entwickelten Applikation umfaßt nicht nur das DBMS, sondern auch die Applikation selbst. Die umfangreichen Wartungsaufgaben des DBMS und wie sich bestimmte

Aufgaben automatisieren lassen, sind in [Petk96] ausführlich beschrieben. Folgende Einstellungen müssen beispielsweise regelmäßig vom Administrator kontrolliert werden: Anzahl der zulässigen Nutzerverbindungen, Speicherplatz für die angelegten Devices, Größe des Transaktionsprotokolls und der Datenbank etc..
Hinter der Wartung verbergen sich eine Vielzahl von Aufgaben, die sehr schwierig abzuschätzen sind und dadurch hohe Kosten verursachen können. Um den Umfang der Wartung besser einordnen zu können, wird die folgende Definition nach [**?**] gegeben:

Definition 4.2 (Wartung) *Wartung ist die Modifikation eines Softwareproduktes nach seiner Auslieferung. Sie umfat die Fehlerkorrektur, die Verbesserung von Leistungs- und anderen Eigenschaften oder die Anpassung des Produkts an eine vernderte Umgebung.*

Zur Wartung gehört demnach auch die Erweiterung der Applikation dahingehend, daß zum Beispiel die Klinik Psychosomatik, die bisher nur eine Station hat, auch auf zwei oder mehr Stationen erweitert werden kann.
Damit ein gewisser beschrankter Wartungsumfang durch den Fachbereich Informatik möglich ist, wurde die Applikation dort installiert. Selbstverständlich ist diese Applikation eine Testversion und hat keine Datenverbindung zum Archiv des Zentrums für Nervenheilkunde.

4.5.2 Backup und Recovery

Aufgrund der sehr sensiblen Daten ist eine reglmäßige Datensicherung notwendig. Dafür steht auf Seiten der Hardware ein Bandlaufwerk zur Verfügung.
In Absprache mit den Verantwortlichen wurde die wöchentliche Datenbanksicherung und die tägliche Transaktionsprotokollsicherung festgelegt. Das bedeutet, daß Daten die wiederhergestellt werden sollen, nicht älter als eine Woche sein können. Bei einem Systemausfall, sei es durch Festplatten-Crash oder einen anderen Fehler, kann maximal die Arbeit eines Tages verlorengehen. Zu beachten ist, daß nicht nur Daten von Aufnahmen gesichert werden, sondern auch die Stored Procedures bzw. Änderungen an ihnen. Sollten Änderungen an diesen Prozeduren vorgenommen werden, muß anschließend sofort das Transaktionsprotokoll oder die Datenbank gesichert werden.

Die Sicherung der Daten kann beim SQL-Server dynamisch erfolgen, also während andere Benutzer mit der Datenbank arbeiten. Aufgrund der zusätzlichen Belastung des Servers wurde die Archivierungszeit auf 21.00 Uhr festgesetzt. Auf Seiten der Software unterstützt der SQL Enterprise Manager die Datensicherung und Wiederherstellung hervorragend. Die Einzelheiten sollen im folgenden nicht weiter behandelt werden. Die Datensicherung und Wiederherstellung wird ausführlich in [Petk96] gezeigt. Weitere Informationen können der sehr guten Online-Dokumentation des SQL-Servers entnommen werden.

Kapitel 5

Zusammenfassung und Ausblick

Die entwickelte Applikation wird in Zukunft, im erheblichen Maße, die Archivierungsarbeit entlasten. Besonders deutlich wird der Vorteil der elektronischen Datenverarbeitung bei der Beantwortung von Anfragen an das Archiv, wenn ein großer Datenbestand vorliegt. Der Partientendatenimport ermöglicht die Speicherung von wesentlich mehr Informationen zu allgemeinen Patientendaten, als die Aufnahmebücher. Für die Kliniken selbst, ist die Applikation ein Informationssystem geworden, auf das sie mit bestimmten Rechten zugreifen können. Sind allerdings noch detailliertere Informationen zu einer Aufnahme erforderlich, als durch das PDAZN zur Verfügung stehen, muß nach wie vor auf die Patientenakte zurückgegriffen werden.

Da das DBMS auch das Potential besitzt, große Datenobjekte abzuspeichern, besteht in Zukunft auch die Möglichkeit einer Datenbestandserweiterung bezüglich wichtiger Bilder oder beispielsweise die komplette Speicherung des Arztbriefes vorzunehmen. Eine Voraussetzung für die Bilder und Texte besteht darin, daß sie in digitalisierter Form vorliegen.

Aufgrund der Skalierbarkeit des SQL-Servers sind auch in Zukunft noch Leistungsreserven verfügbar, in dem bei Bedarf mehrere Prozessoren unterstützt werden können.
Falls wider Erwarten, die zu verwaltende Datenmenge, die durchnittliche Anzahl von 3000 Aufnahmen pro Jahr, um Größenordnungen übersteigen sollte oder wesentlich mehr klinikspezifische Daten gespeichert werden müssen, wäre auch die Umstellung auf eine verteilte Datenbank denkbar, die durch den SQL-Server unterstützt wird. Sinnvoll wäre diese Architektur der Client/Server-Technologie besonders dann, wenn das PDAZN alle Daten einer Patientenakte speichern würde. Das dabei eingesetzte Modell könnte so aussehen, wie in Abbildung 5.1 dargestellt wird. Die Die Daten würden so verteilt werden, daß eine Abspeicherung der allgemeinen Patientendaten und Diagnosen weiterhin auf dem Server des Archivs erfolgt. Die klinikspezifischen Daten müßten auf den Klinik-Servern gespeichert werden. Der Server im Archiv bleibt weiterhin der primäre Domänen-Controller (PDC). Die Klinik-Server würden mit der Windows NT 4.0 Server-Version und dem SQL-Server ausgestattet werden

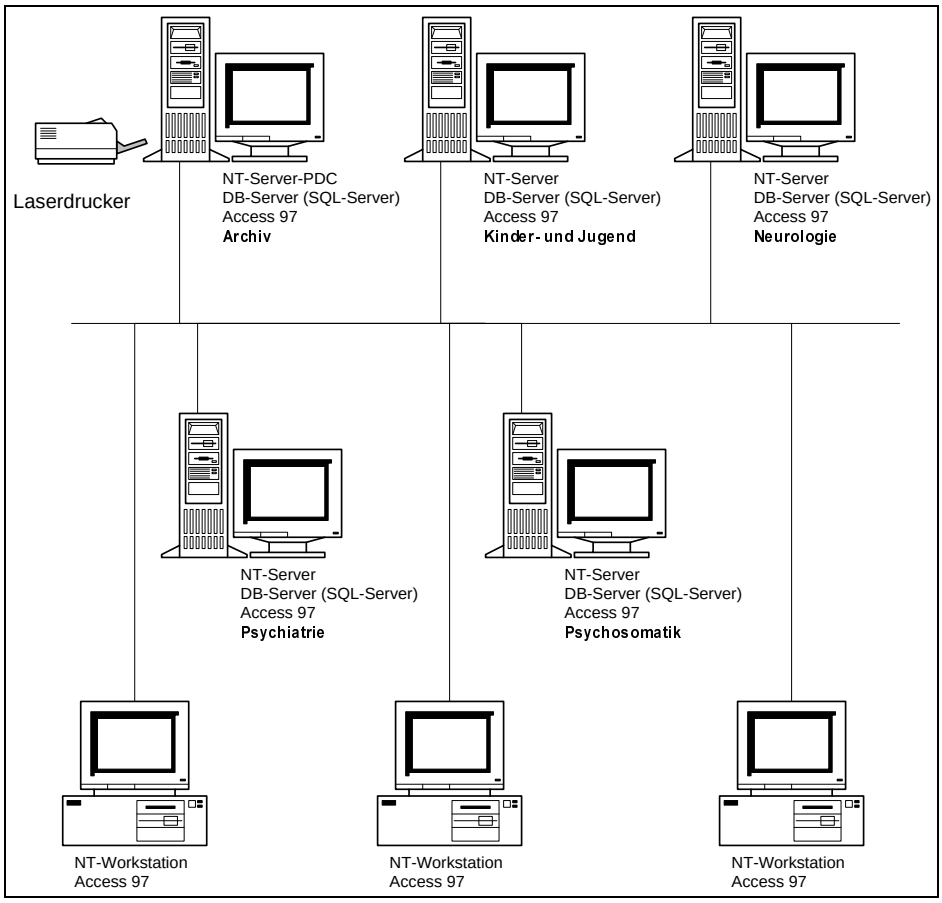


Abbildung 5.1: Domäne ZN bei verteilten Datenbanken - Erweiterung der Domäne ZN um weitere Datenbank-Server

müssen.
Als erweiterbar ist auch der Statistik-Teil des PDAZN zu betrachten, der in der ersten Version nur elementare Anfragen an das Archiv berücksichtigen kann.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[AlNi96] Albrecht, Ralf; Nicol, N.: Programmierung mit Access 7 für Windows 95. Professionelle Anwendungsentwicklung mit Access 7 und VBA. Addison Wesley, Bonn, 1996

[Bers92] Berson, A.: Client/Server Architectur. McGrawHill, 1992 Professionelle Anwendungsentwicklung mit Access 7 und VBA. Addison Wesley, Bonn, 1996

[Bros97] Brosius, G.: Access 97 professionell: Datenbank-Management mit Microsoft Office 97. Addison Wesley, Bonn, 1997

[ChGu+96] Chipman, M.; Gunderloy, M.; Viescas, J.: Das Programmierbuch. Access/SQL-Server. Sybex-Verlag, Düsseldorf,1996

[Clure93] McClure, C.: Softwareautomatisierung. Reengineering Repository Reuse. Hanser Verlag,Mnchen, 1993

[Dada96] Dadam, P.: Verteilte Datenbanken und Client/Server-Systeme. Springer Verlag, Berlin, 1996

[Dapp+96] Dapper, T; Dietrich, C.; Klöppel, B.; u.a.: Windows NT 4.0 im profesionellen Einsatz. Hanser Verlag, München, 1996

[HeSa95] Heuer, A; Saake, G: Datenbanken. Konzepte und Sprachen. Thomson Publishing, Bonn, 1995

[KaMü94] Karer, A; Müller, B.: Client/Server-Technologie in der Unternehmenspraxis. Springer Verlag, Berlin, 1994

[Petk96] Petković, D.: Microsoft SQL Server 6.5. Das Datenbanksystem im BackOffice. Addison Wesley, Bonn, 1996

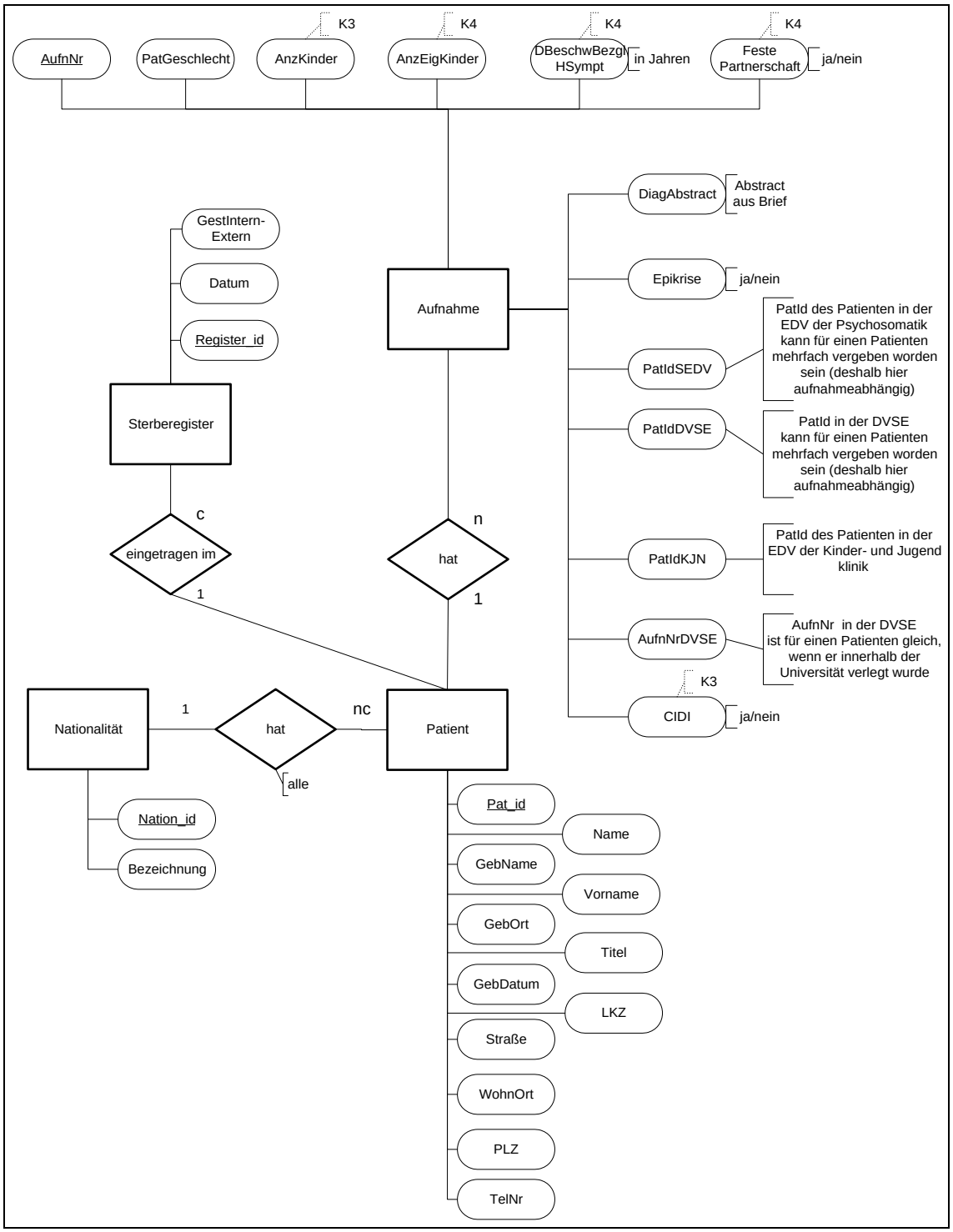
[Schi95] Schimpf, A.: Client/Server-Konzepte. Markt&Technik, Haar, 1995

[TiKo97] Tiemeyer, E.; Konopasek, K.:Professionelles Datenbank-Design mit ACCESS. Vieweg-Verlag, Braunschweig/Wiesbaden, 1997

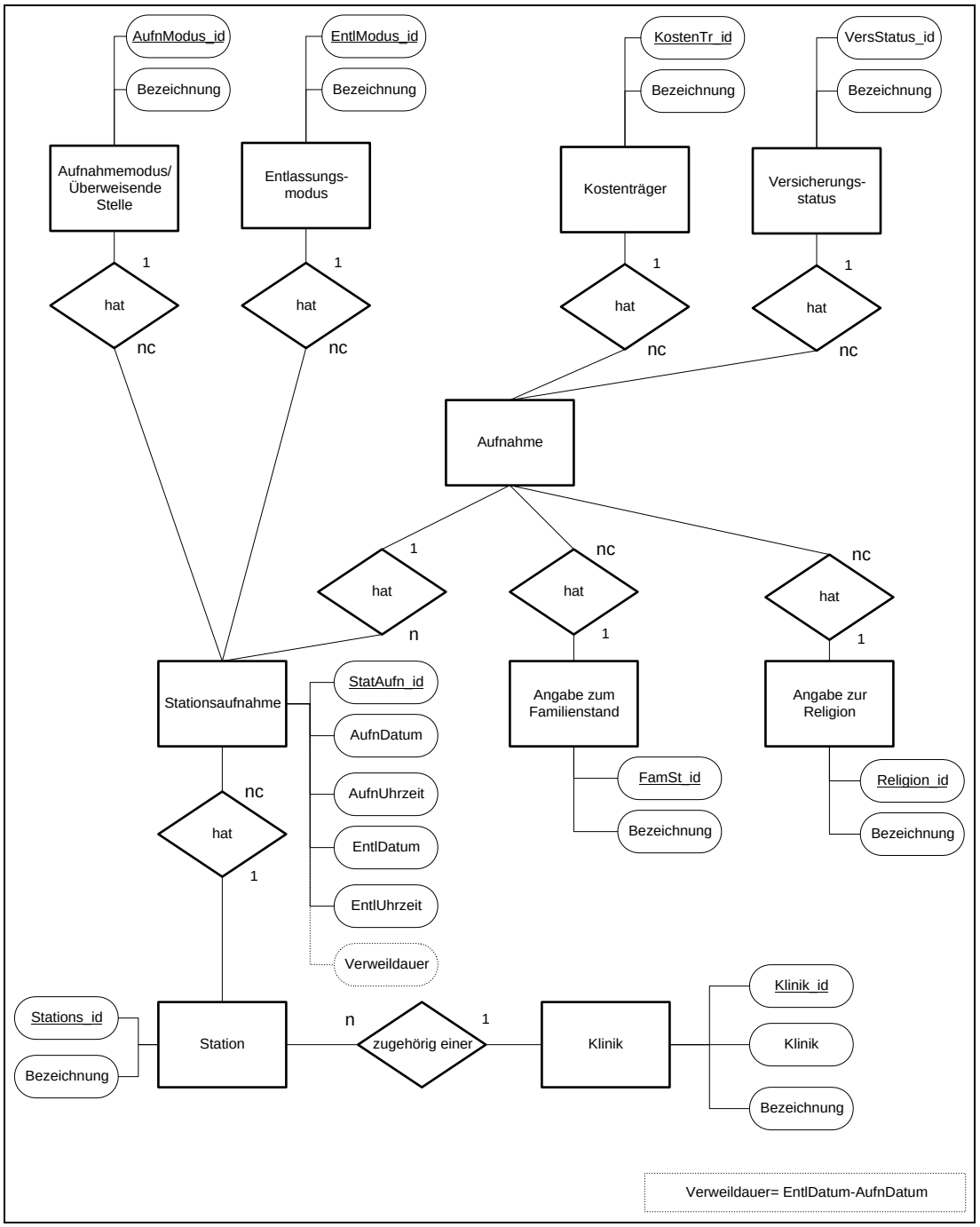
Anhang A

ER-Modelle

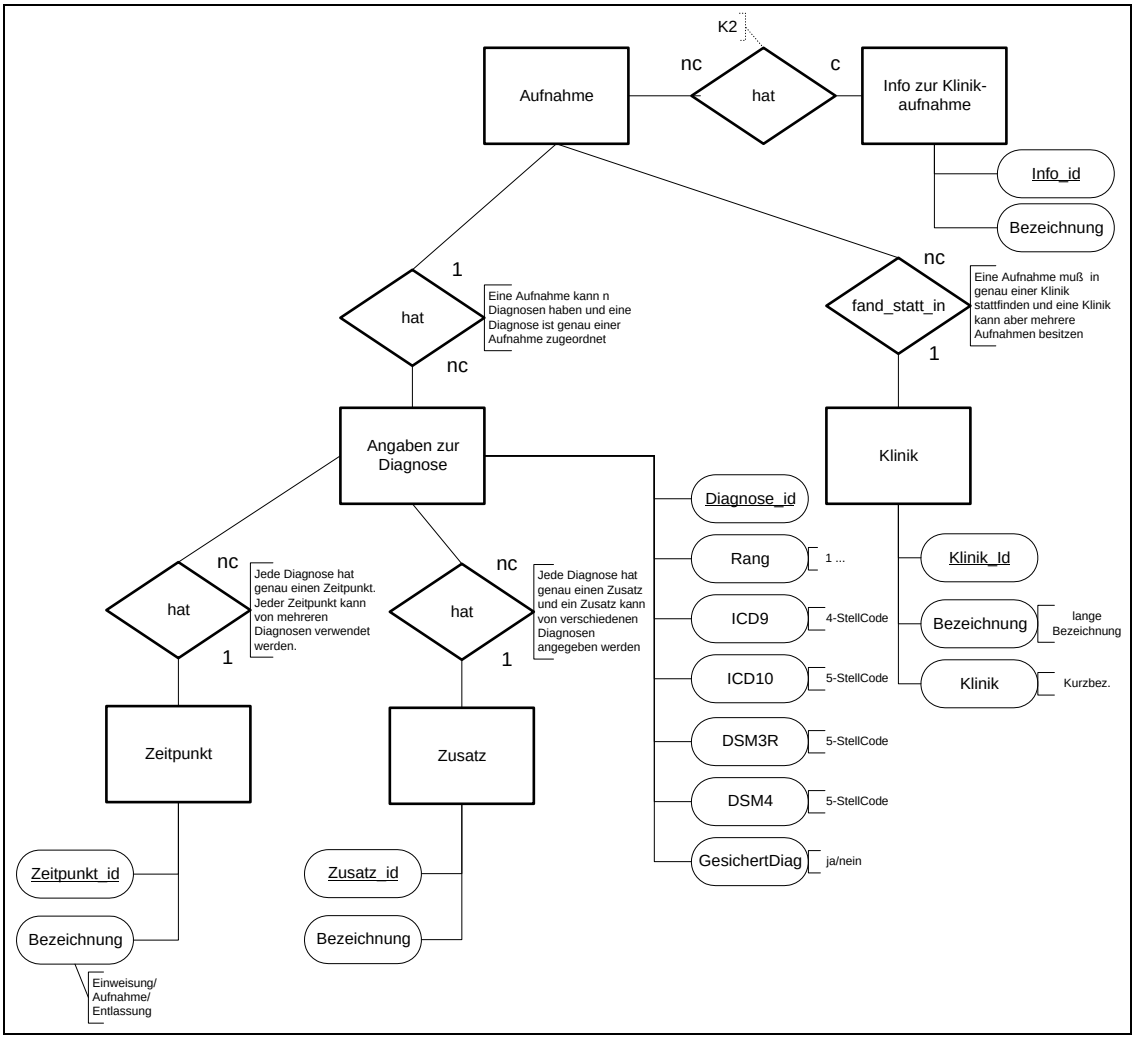
A.1 Gesamt-ER-Modell



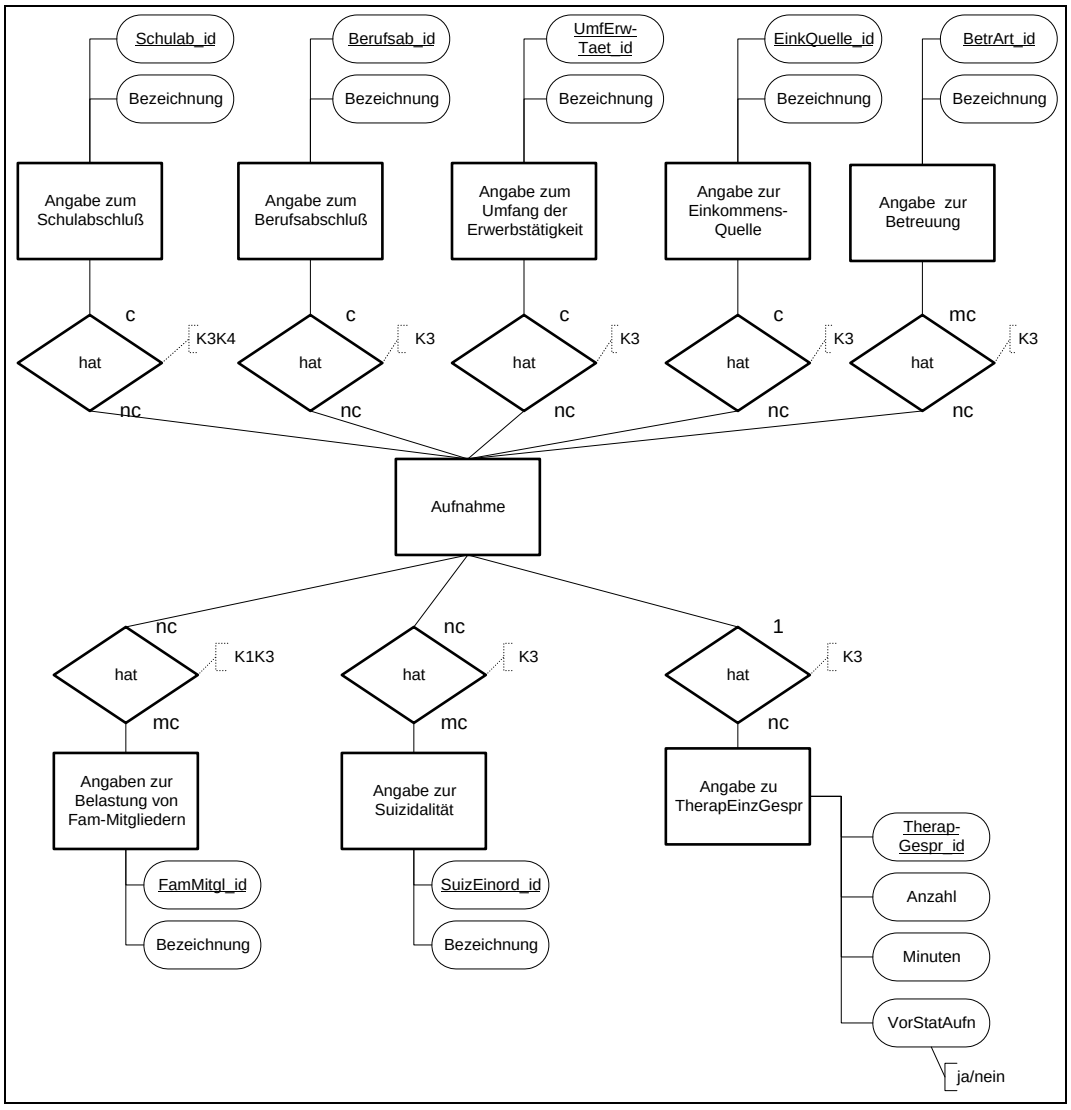
PDAZN Gesamt-ER-Modell-Auszug 1



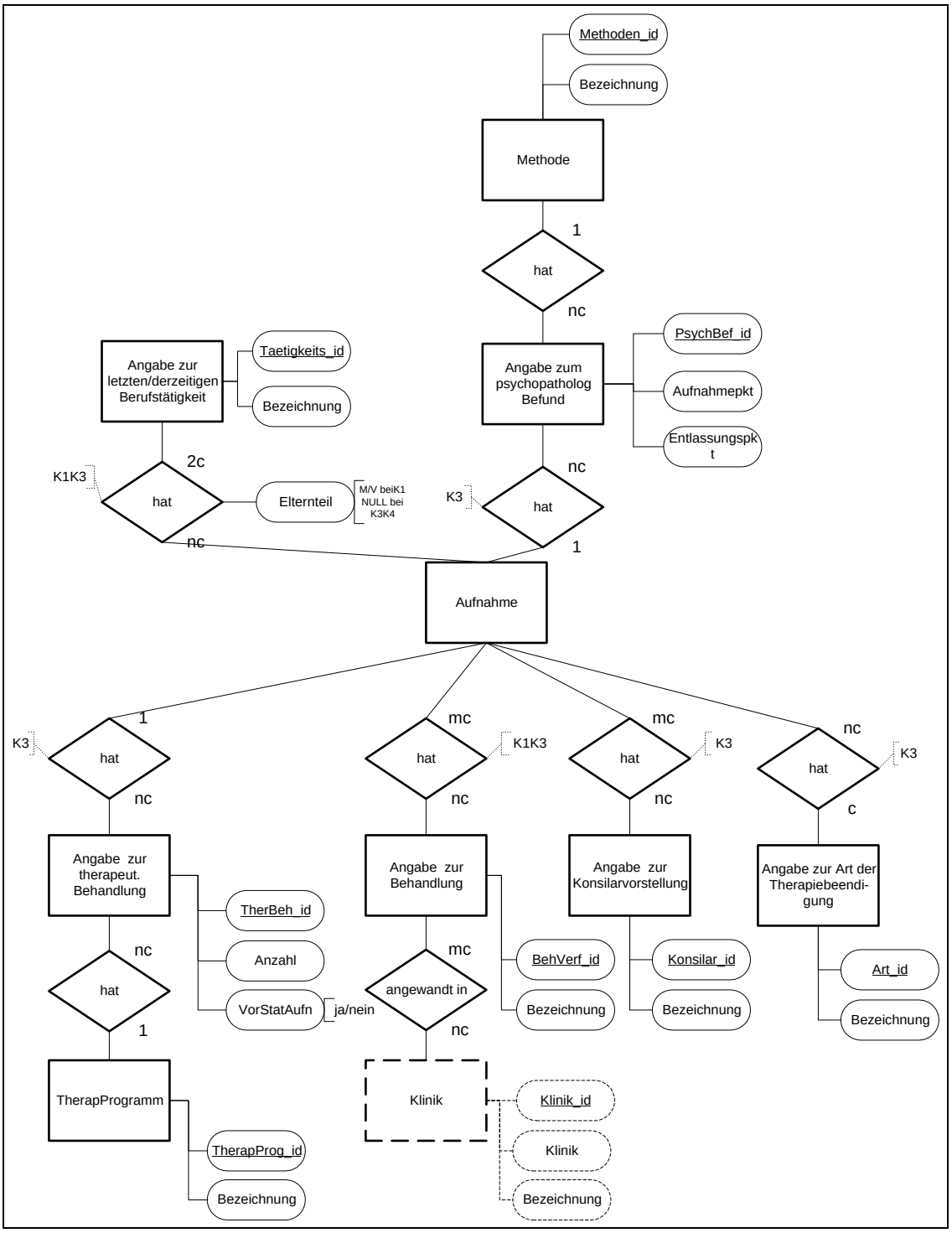
PDAZN Gesamt-ER-Modell-Auszug 2



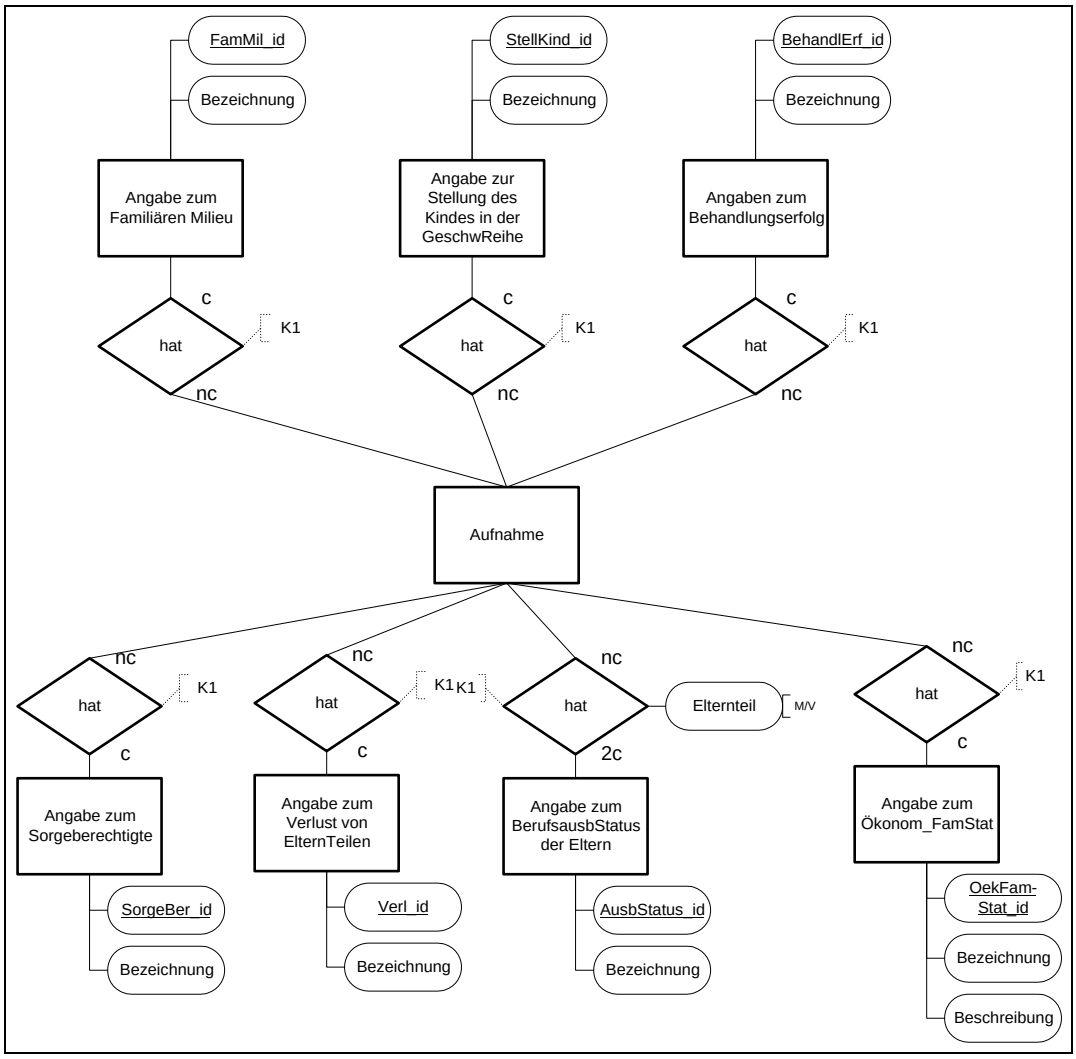
PDAZN Gesamt-ER-Modell-Auszug 3



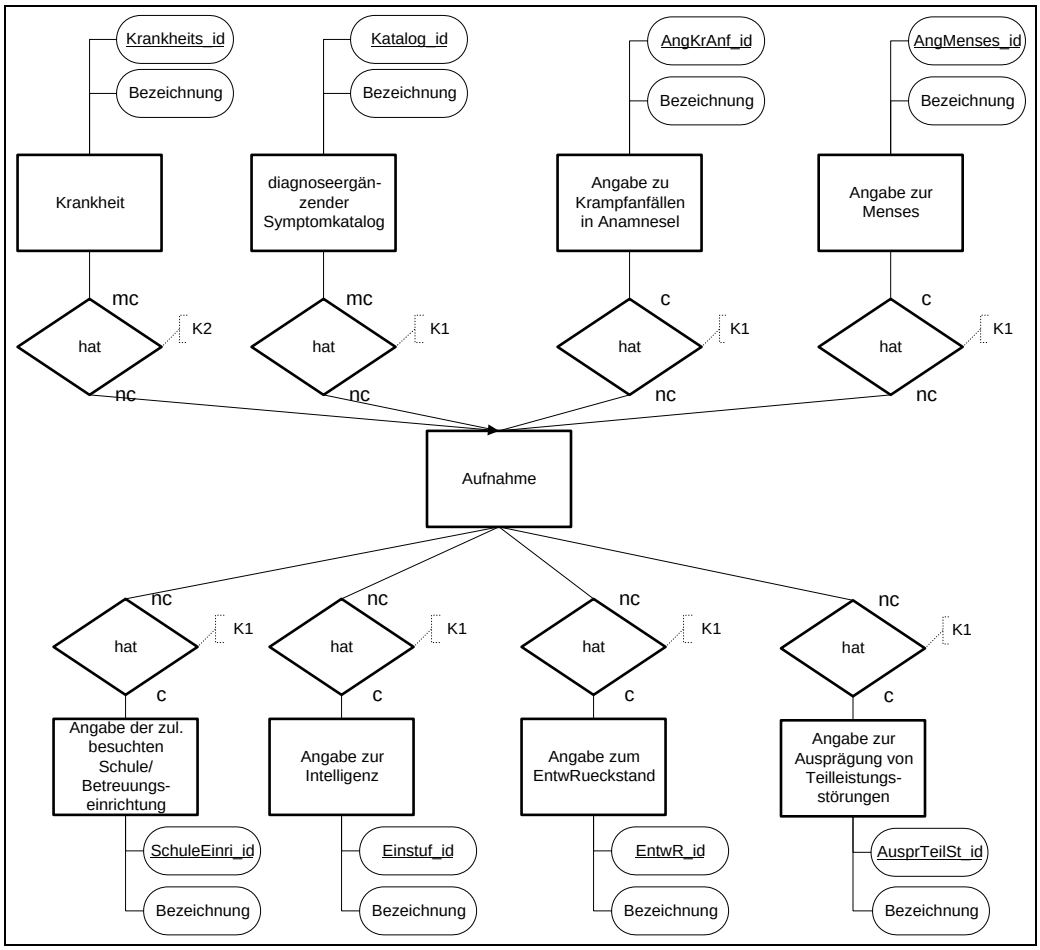
PDAZN Gesamt-ER-Modell-Auszug 4



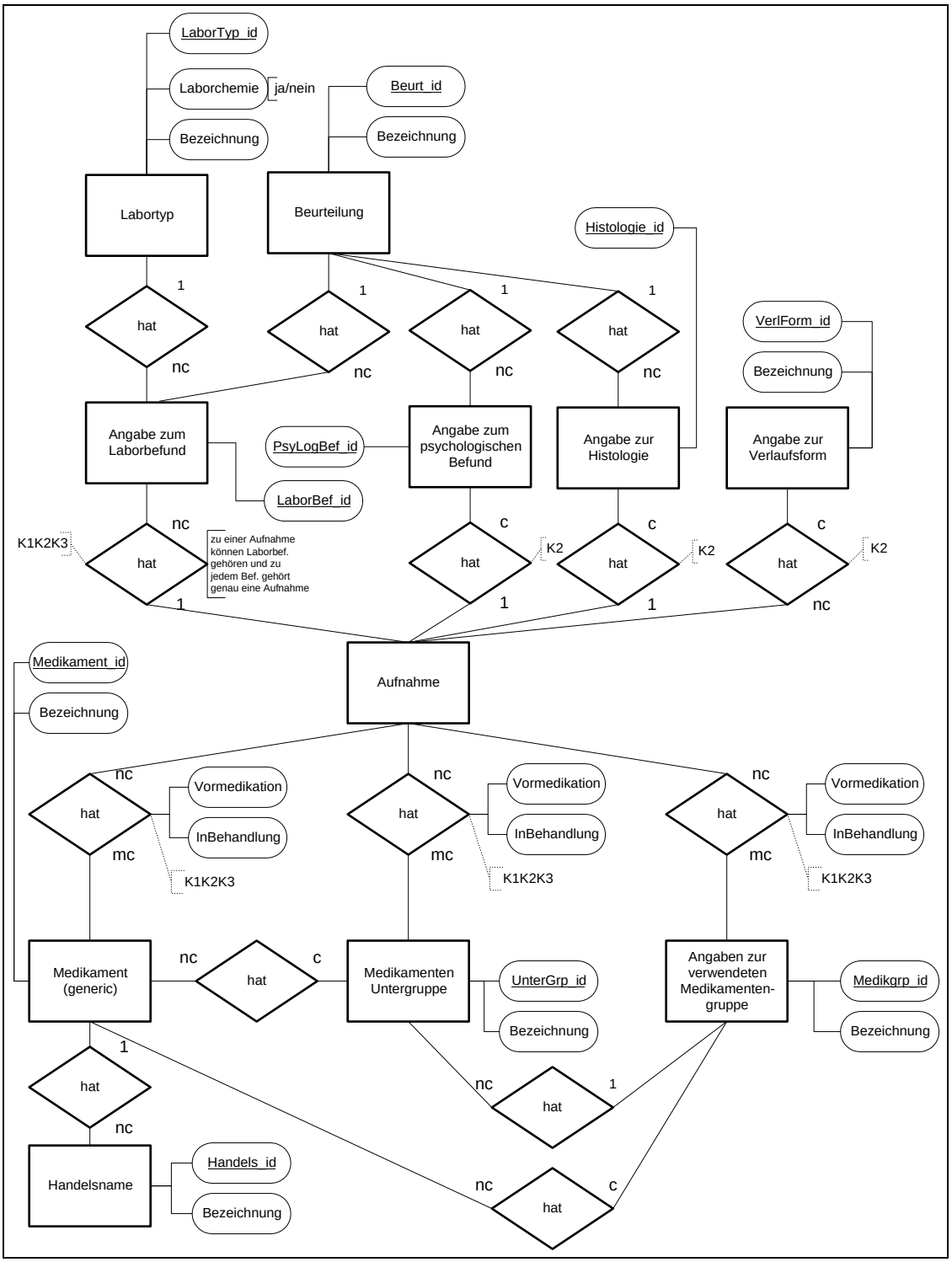
PDAZN Gesamt-ER-Modell-Auszug 5



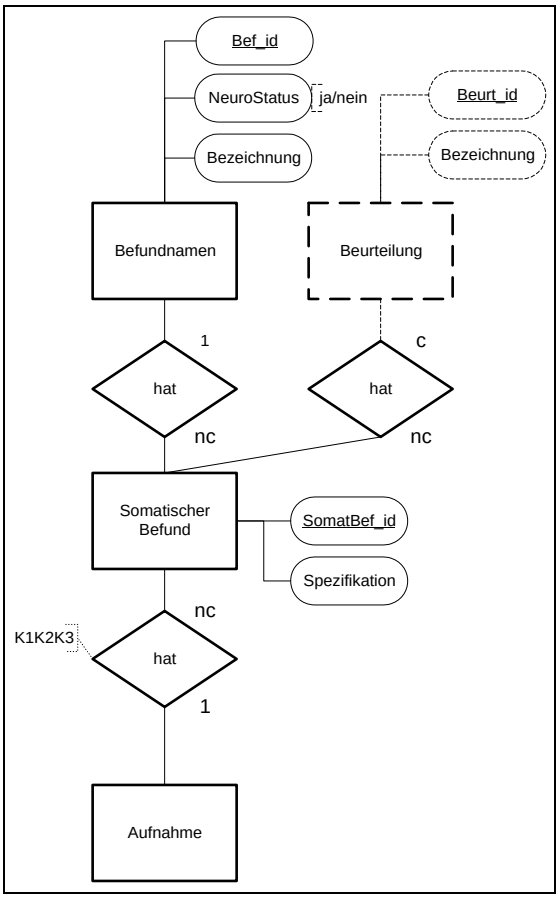
PDAZN Gesamt-ER-Modell-Auszug 6



PDAZN Gesamt-ER-Modell-Auszug 7

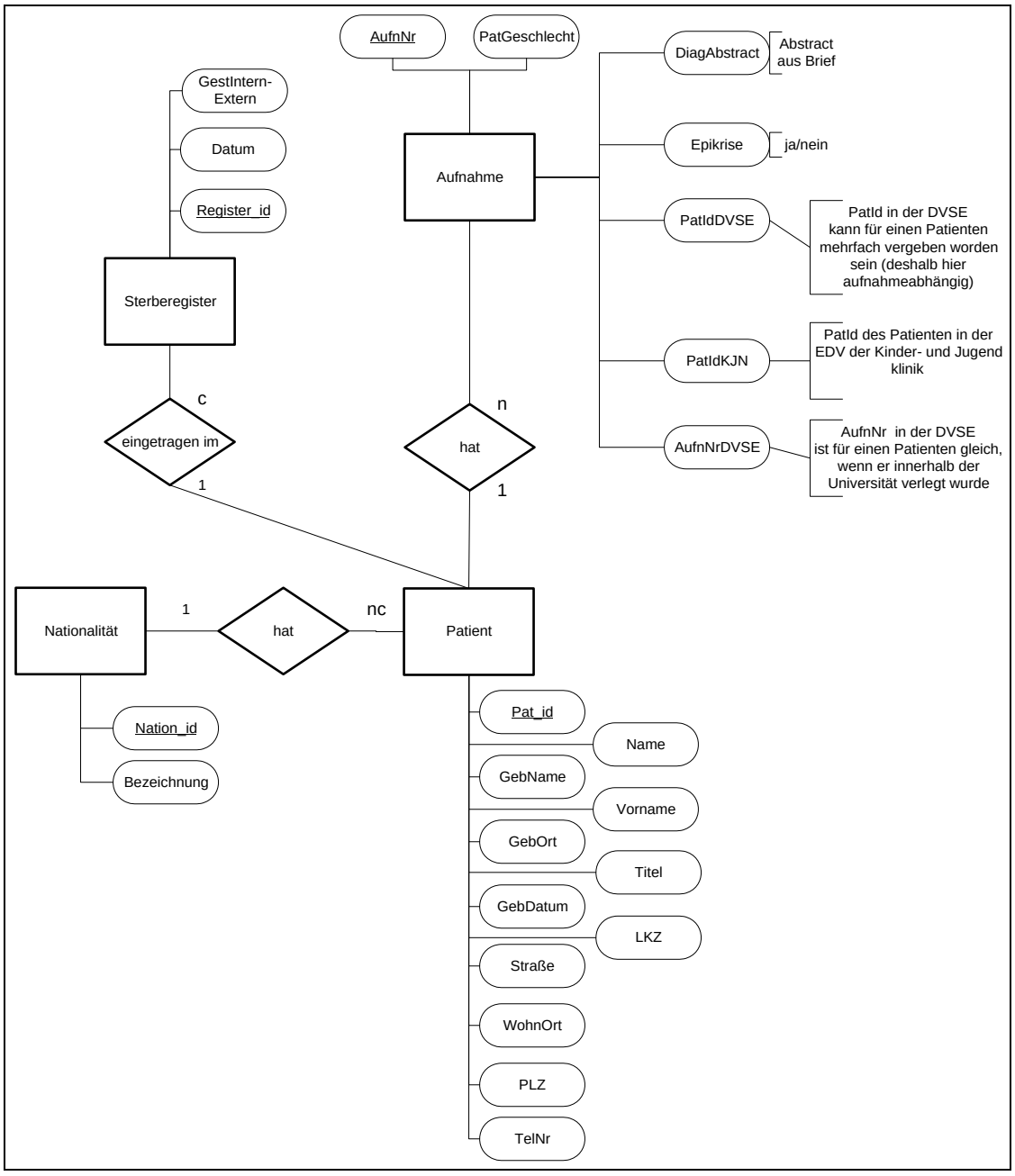


PDAZN Gesamt-ER-Modell-Auszug 8

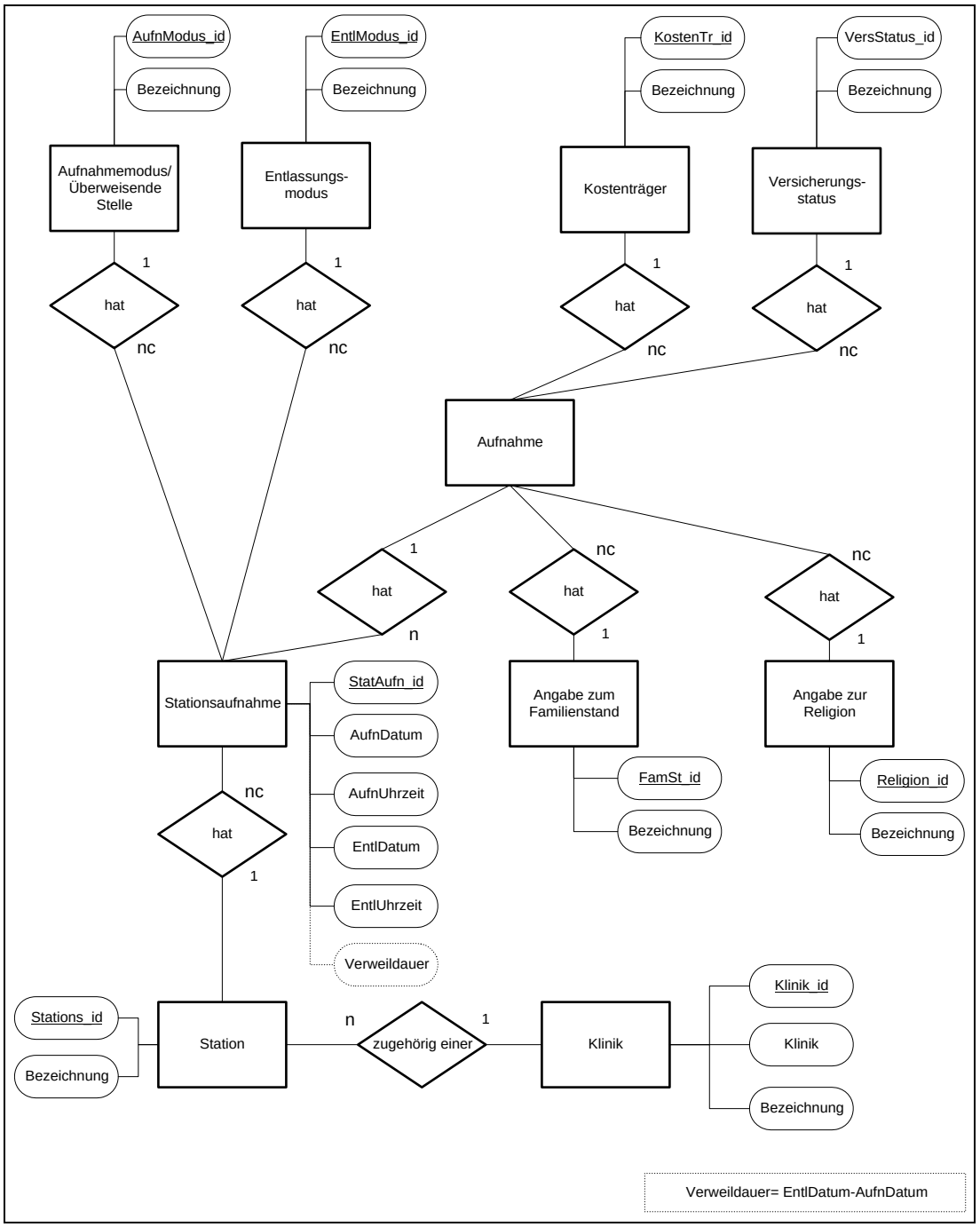


PDAZN Gesamt-ER-Modell-Auszug 9

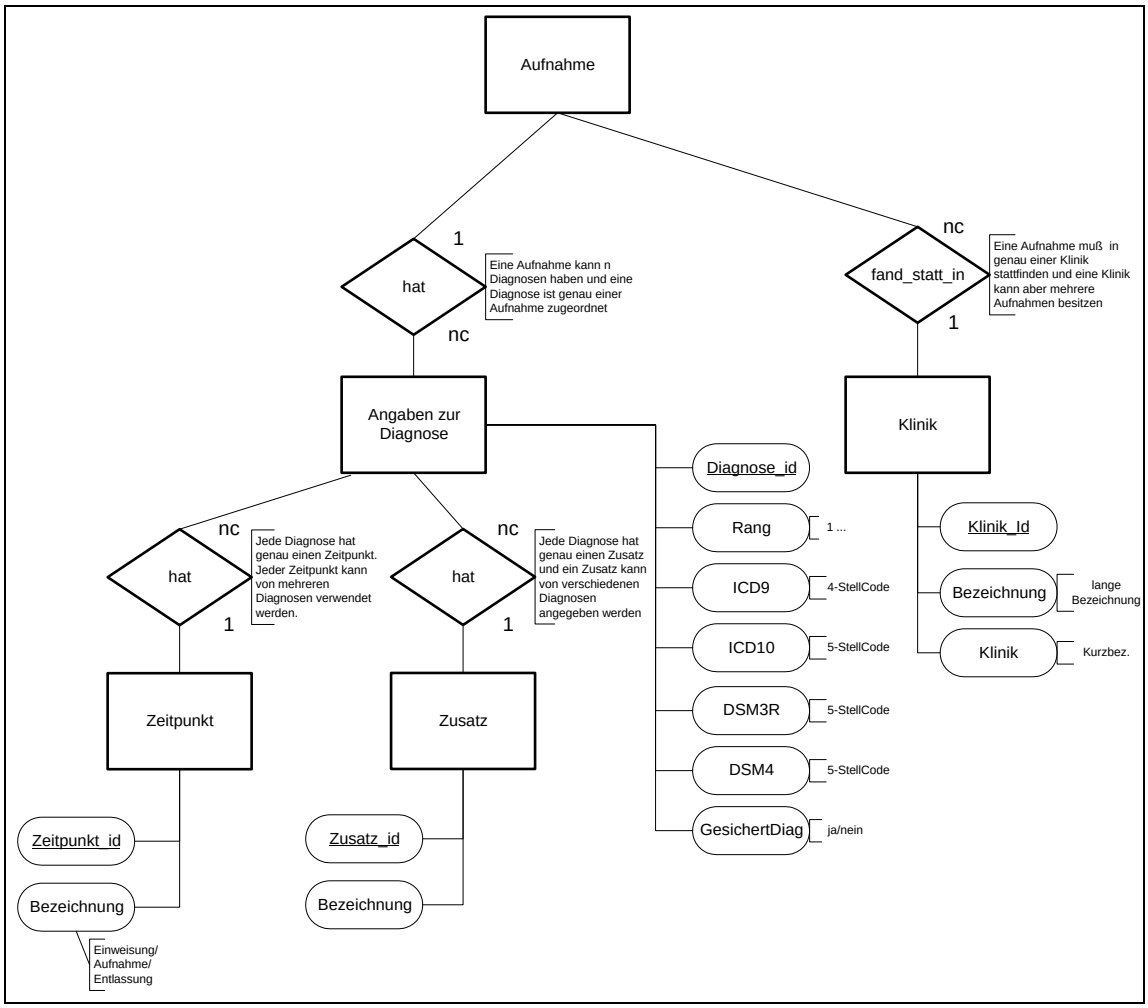
A.2 ER-Modell der Kinder- und Jugendklinik



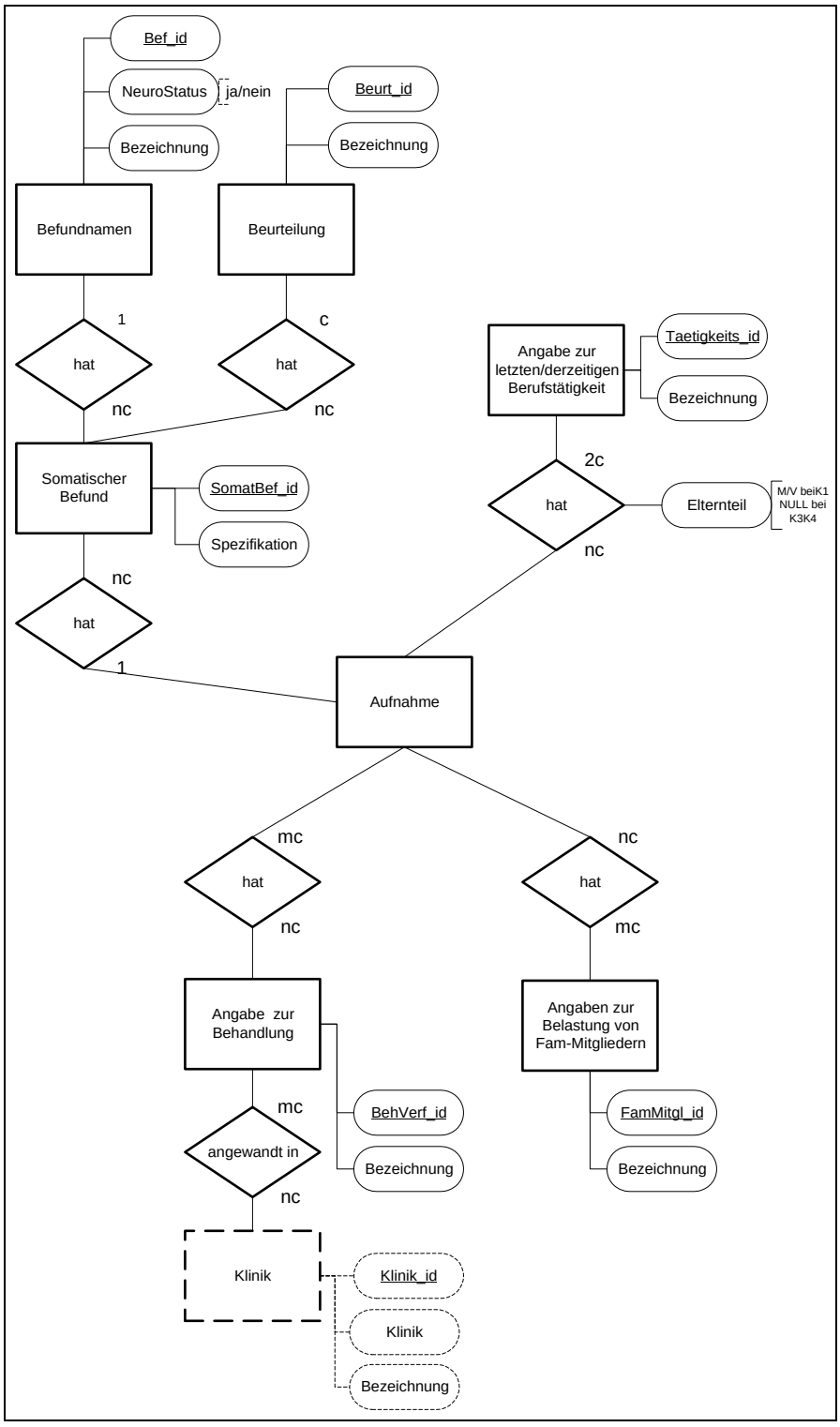
PDAZN Teilmodell Kinder- und Jugendklinik-Auszug 1



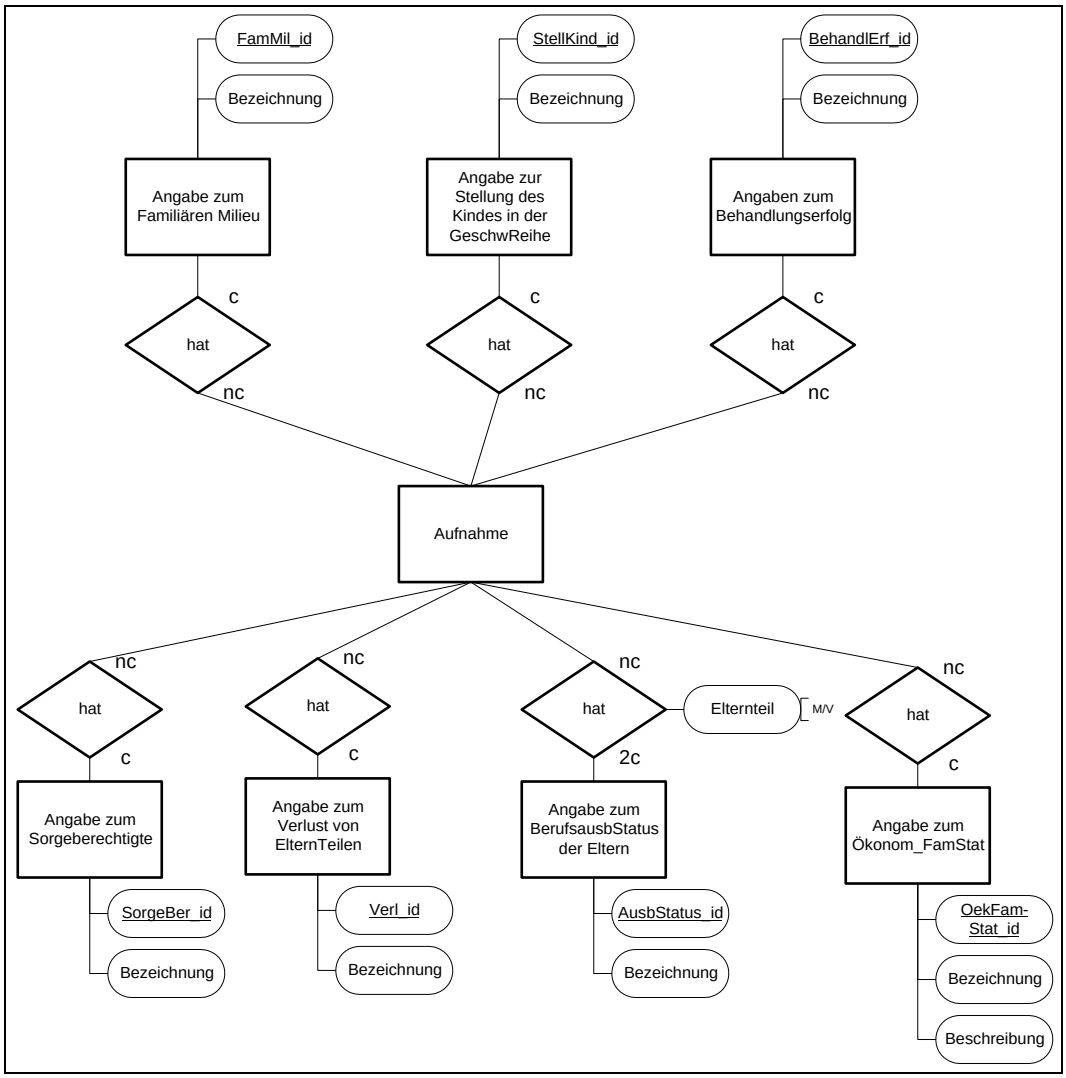
PDAZN Teilmodell Kinder- und Jugendklinik-Auszug 2



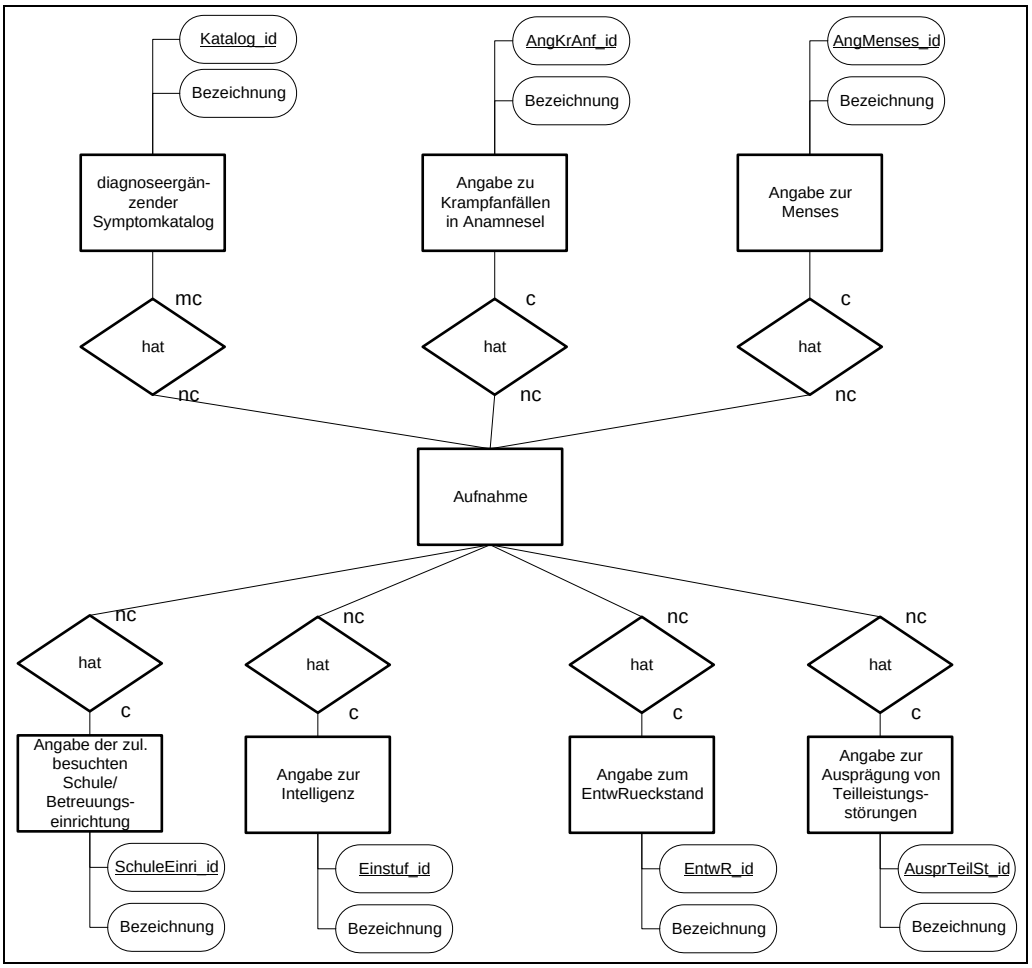
PDAZN Teilmodell Kinder- und Jugendklinik-Auszug 3



PDAZN Teilmodell Kinder- und Jugendklinik-Auszug 4

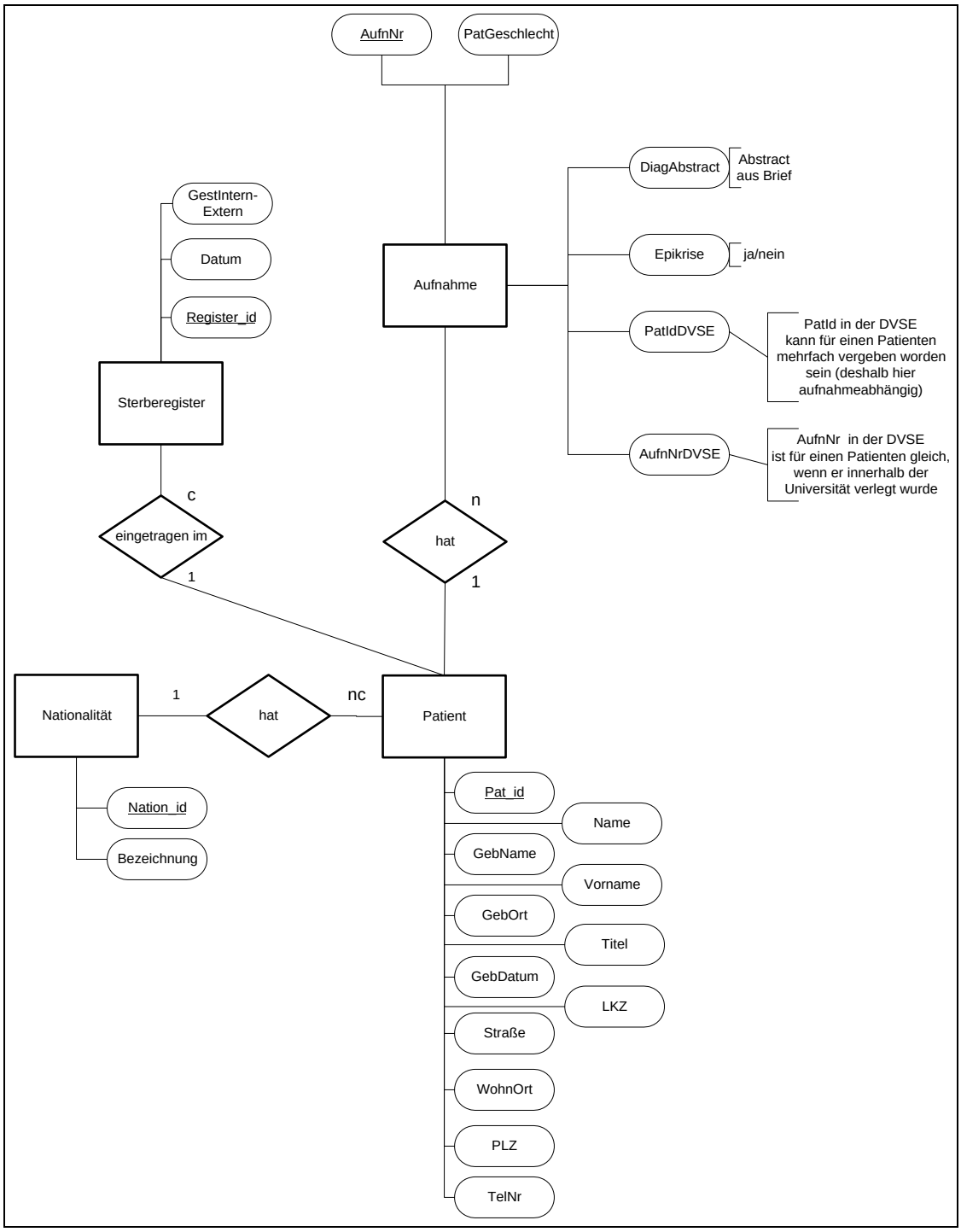


PDAZN Teilmodell Kinder- und Jugendklinik-Auszug 5

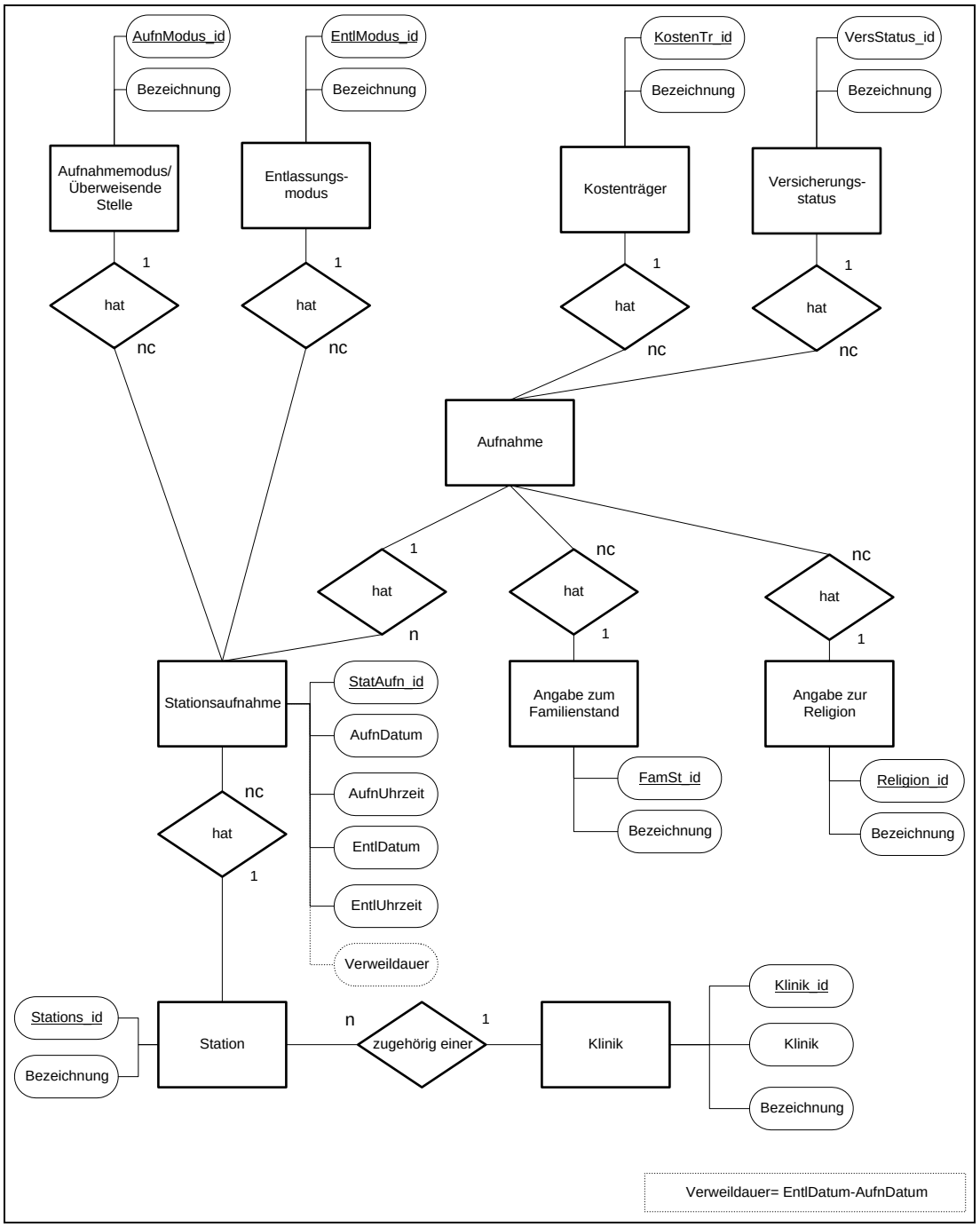


PDAZN Teilmodell Kinder- und Jugendklinik-Auszug 6

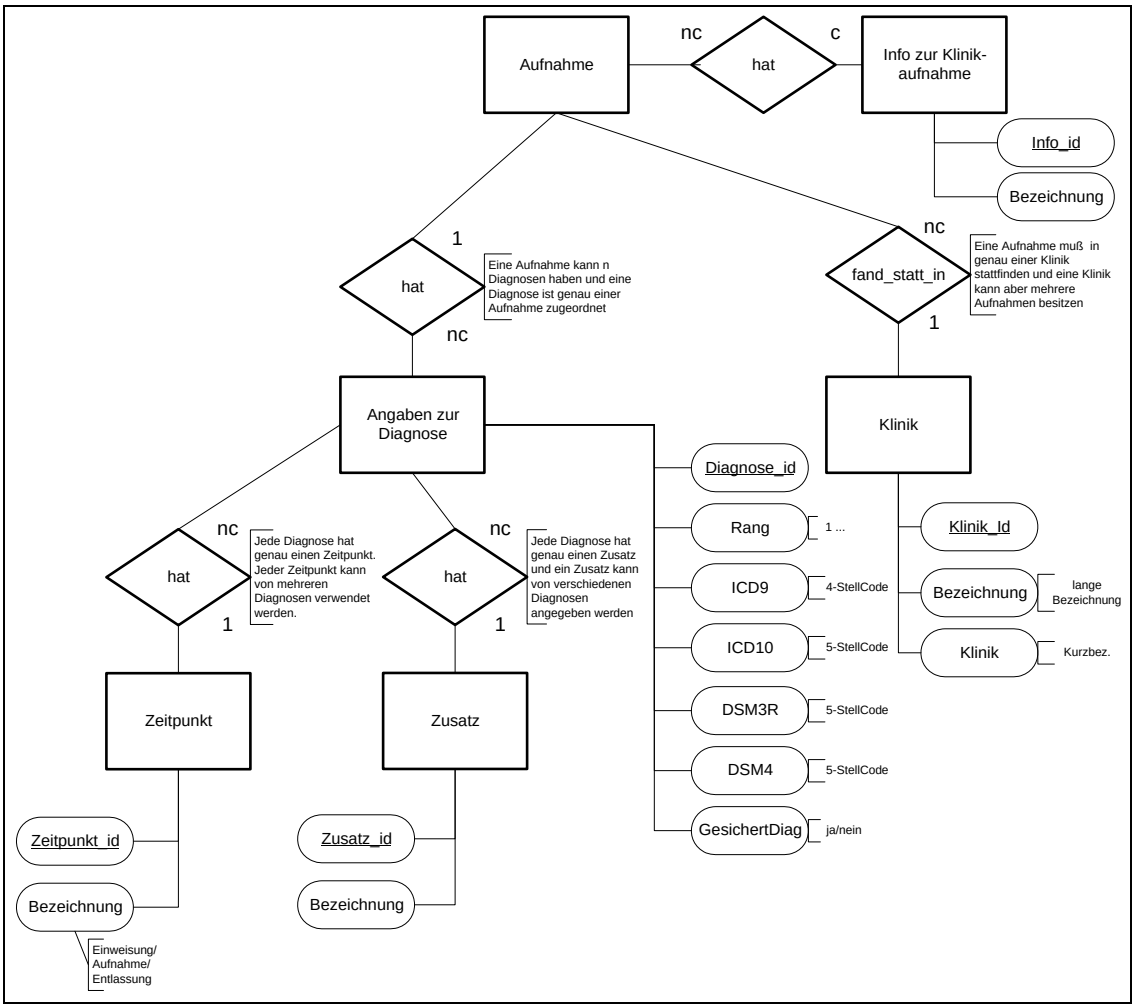
A.3 ER-Modell der Neurologie



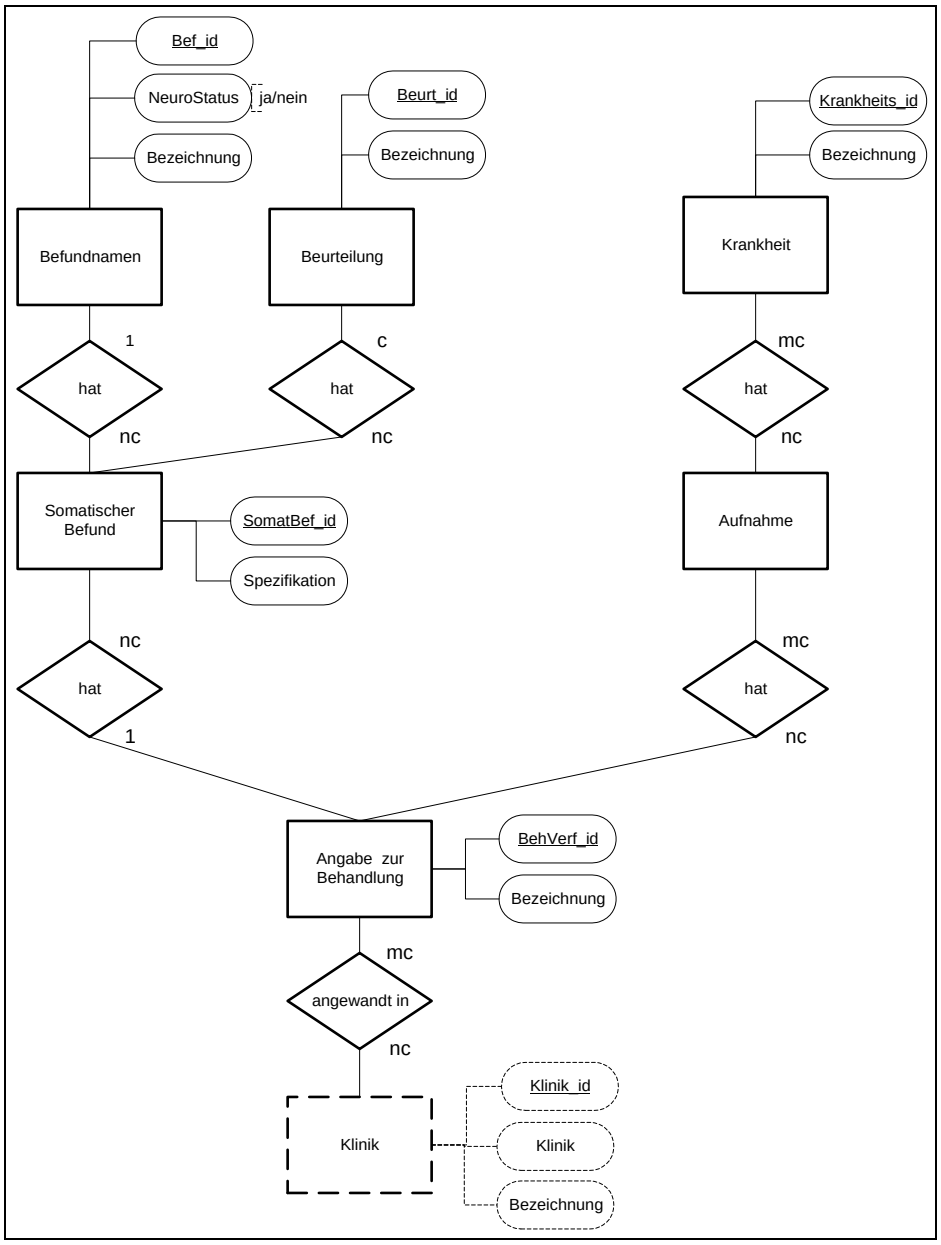
PDAZN Teilmodell Neurologie-Auszug 1



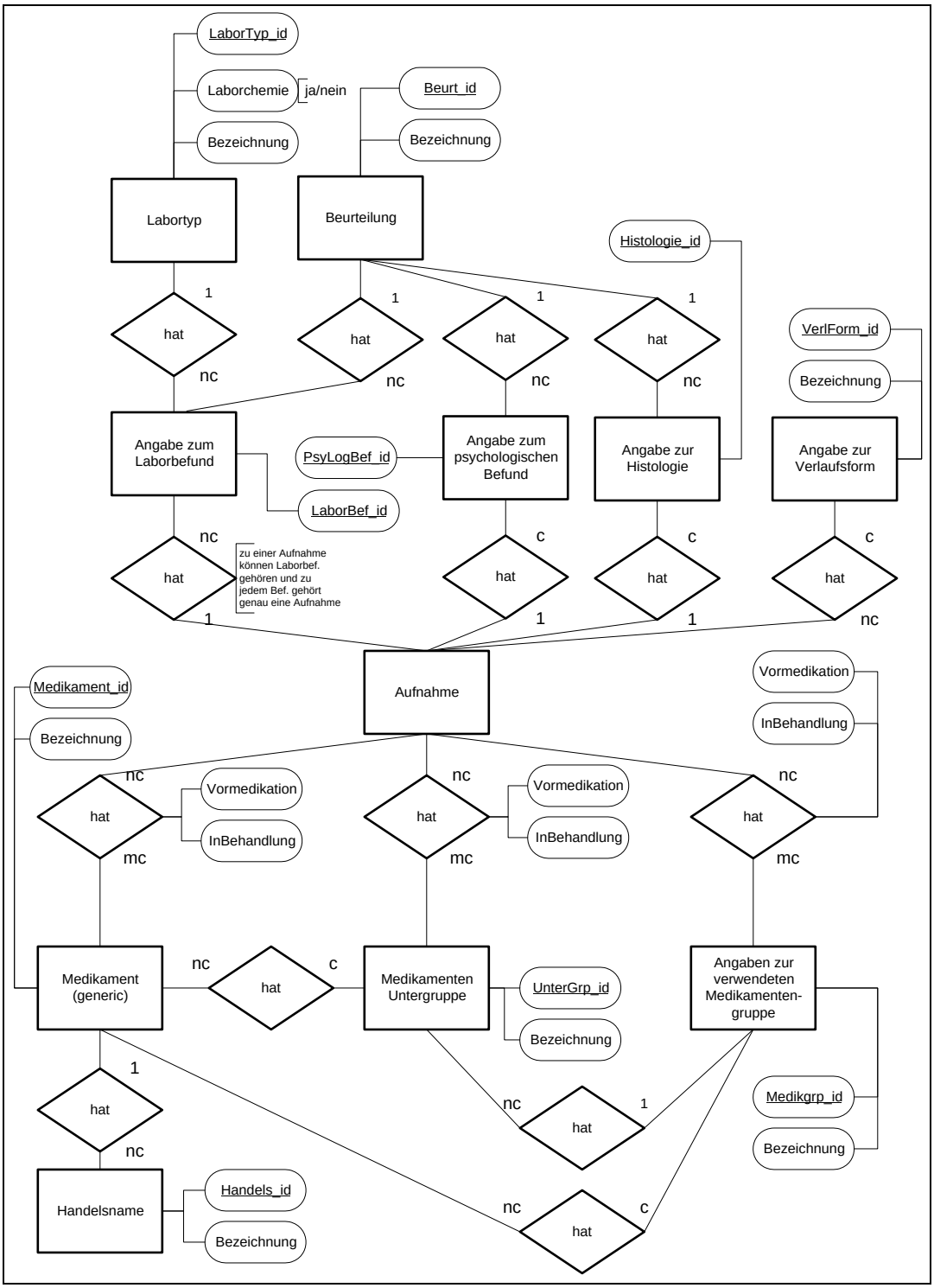
PDAZN Teilmodell Neurologie-Auszug 2



PDAZN Teilmodell Neurologie-Auszug 3

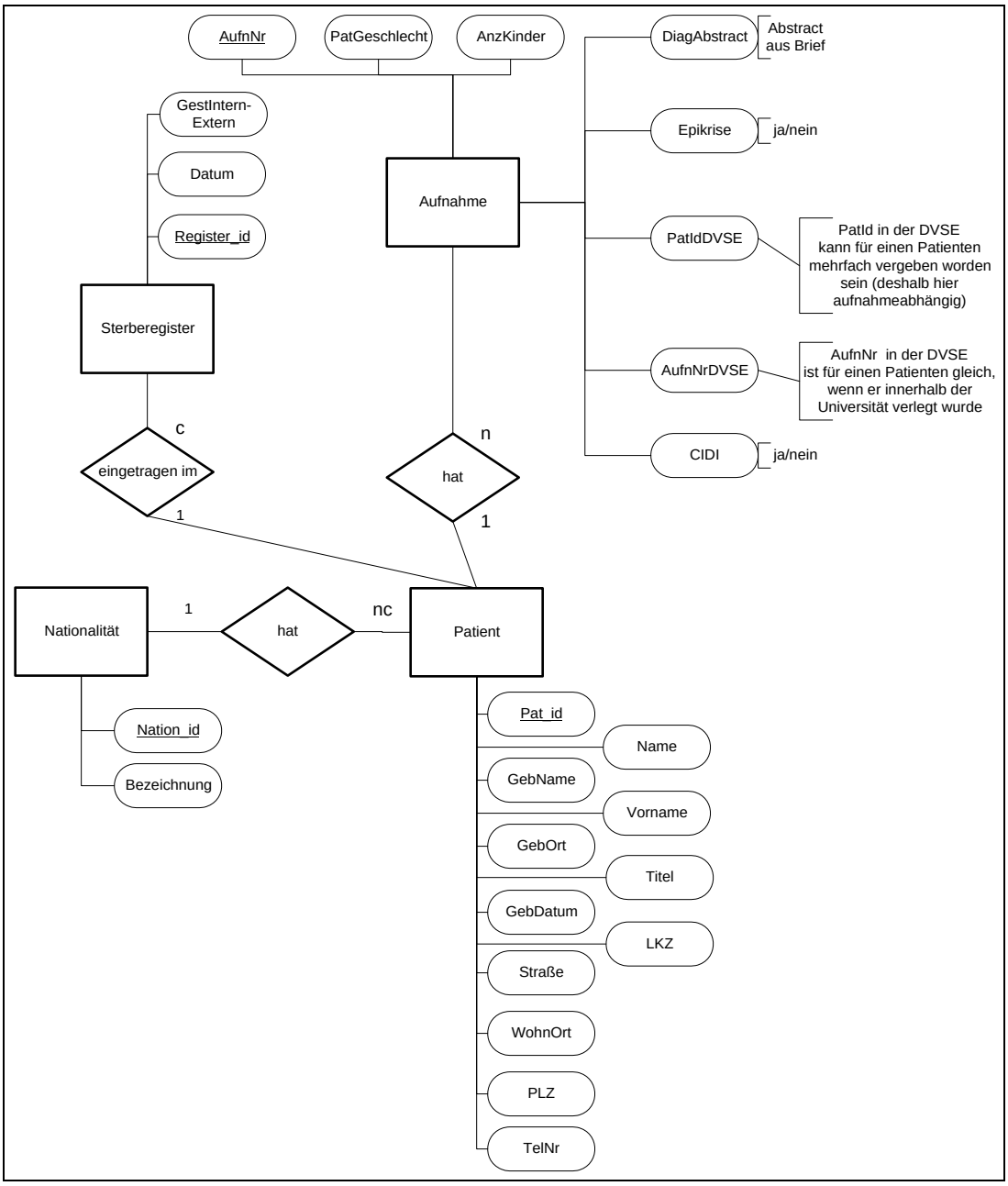


PDAZN Teilmodell Neurologie-Auszug 4

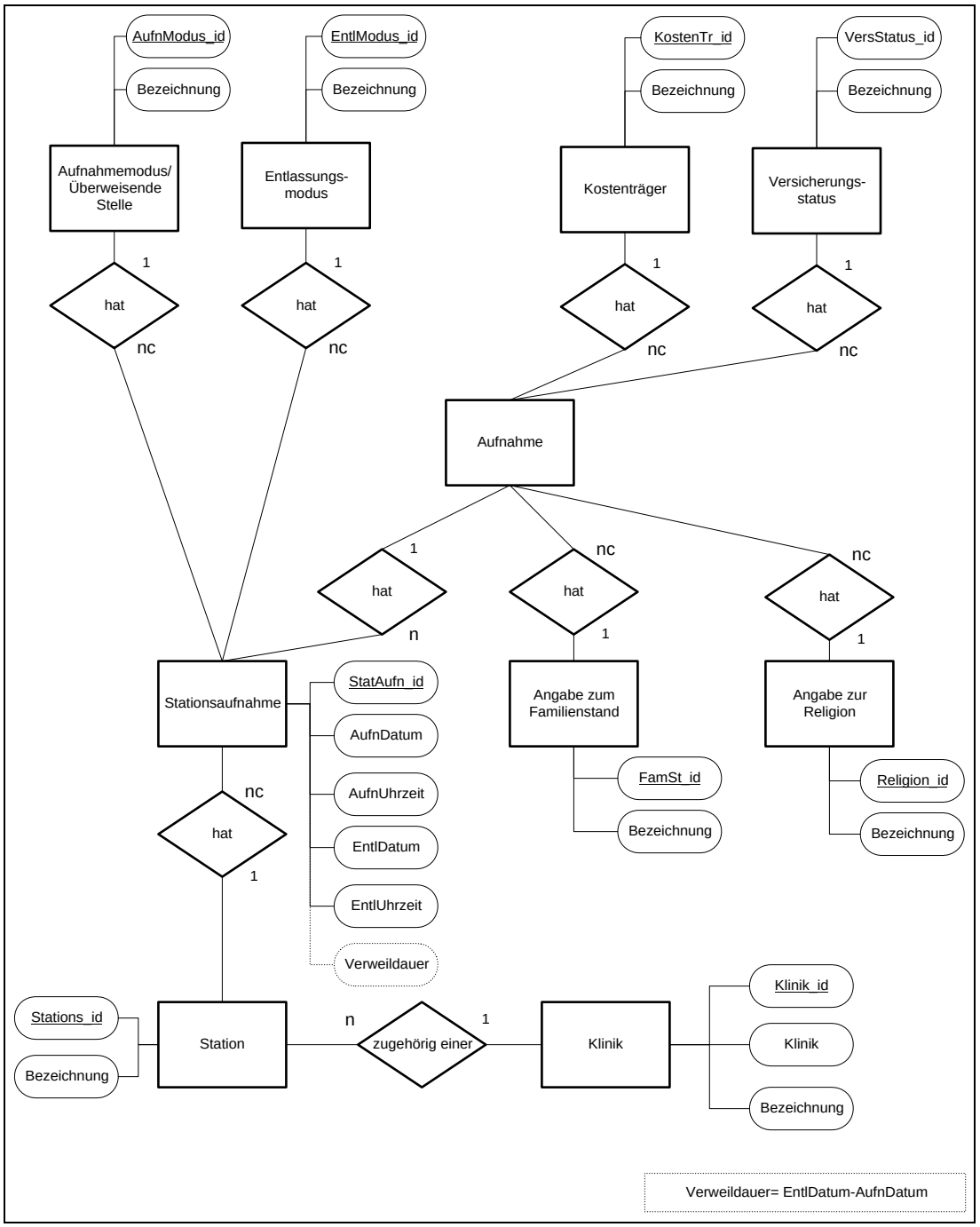


PDAZN Teilmodell Neurologie-Auszug 5

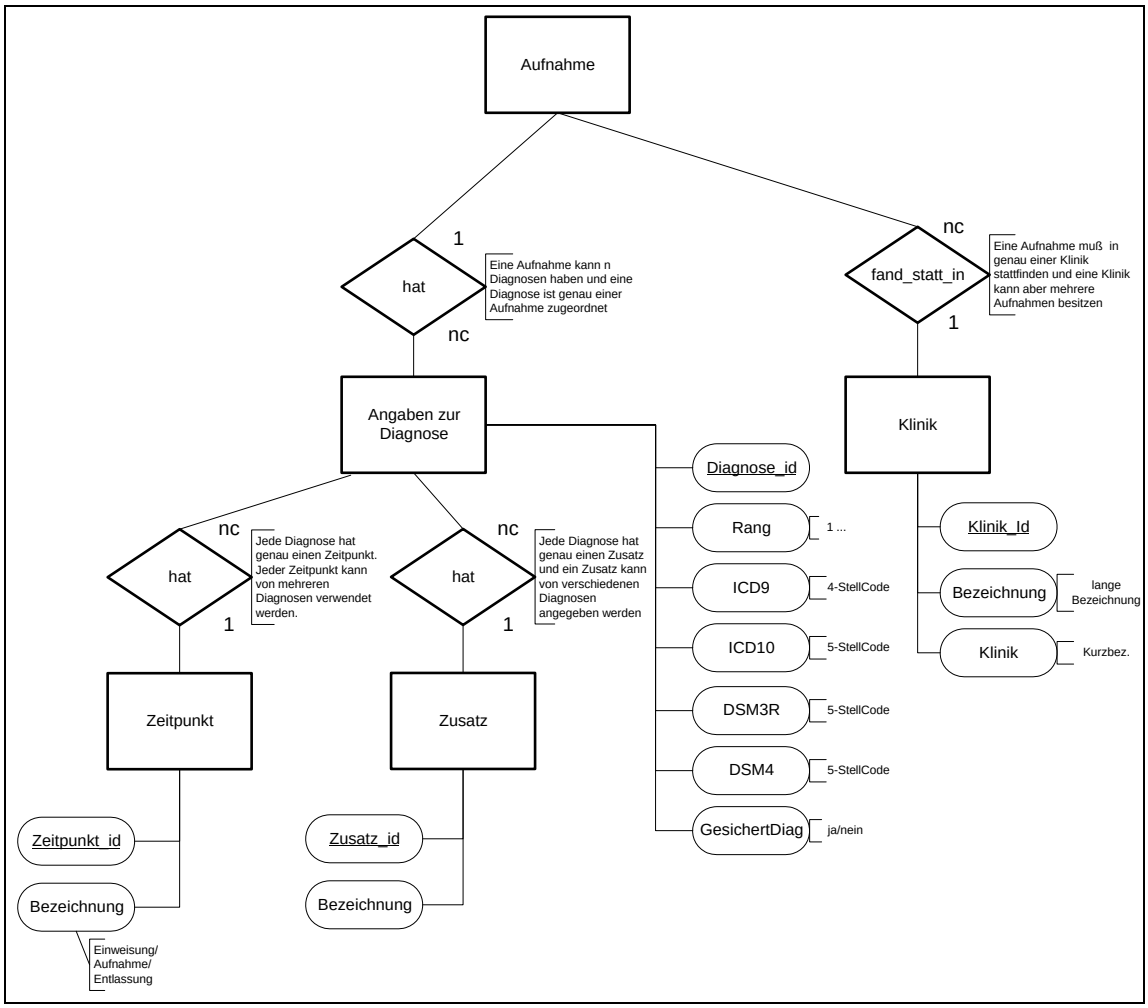
A.4 ER-Modell der Psychiatrie



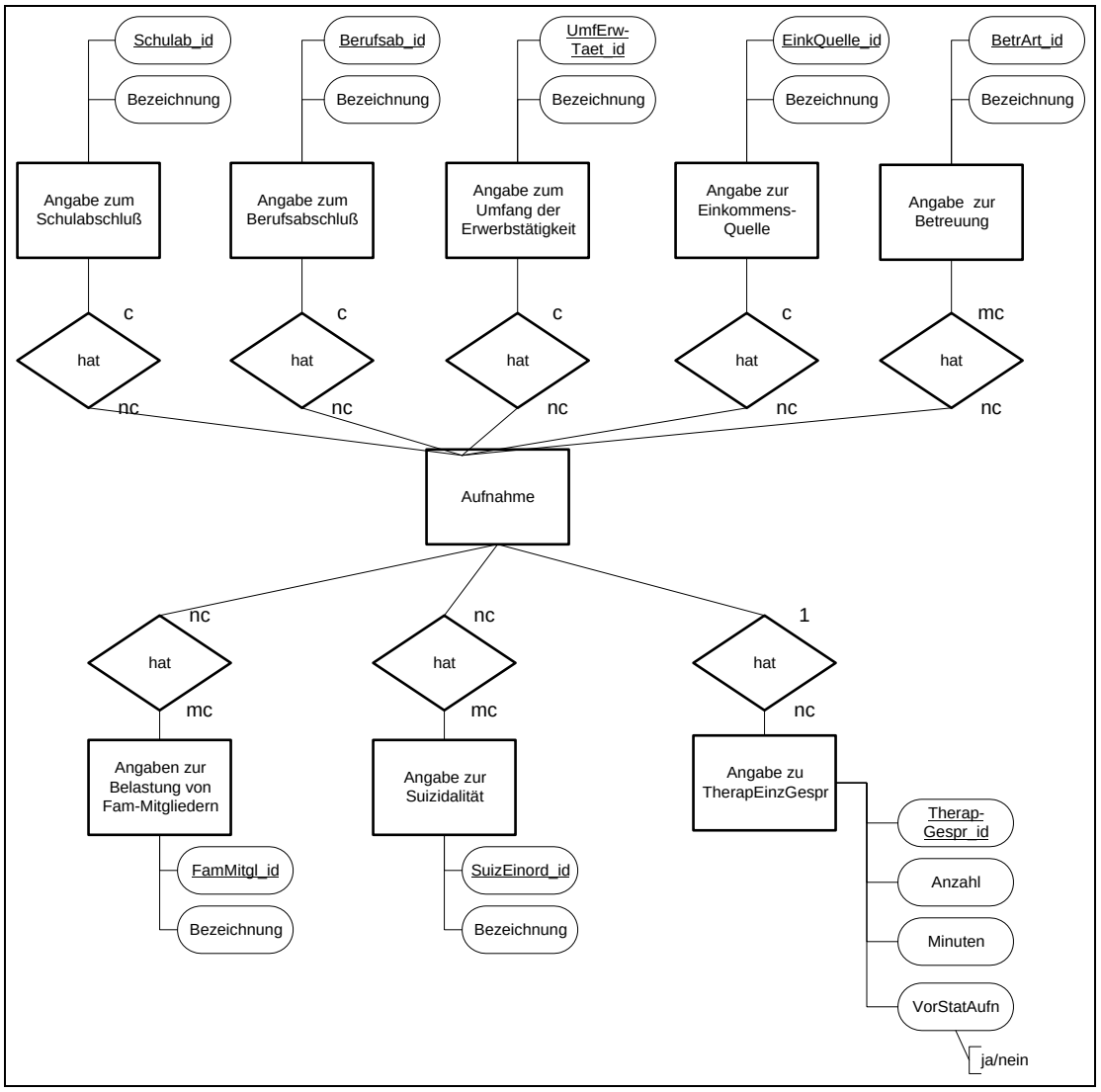
PDAZN Teilmodell Psychiatrie-Auszug 1



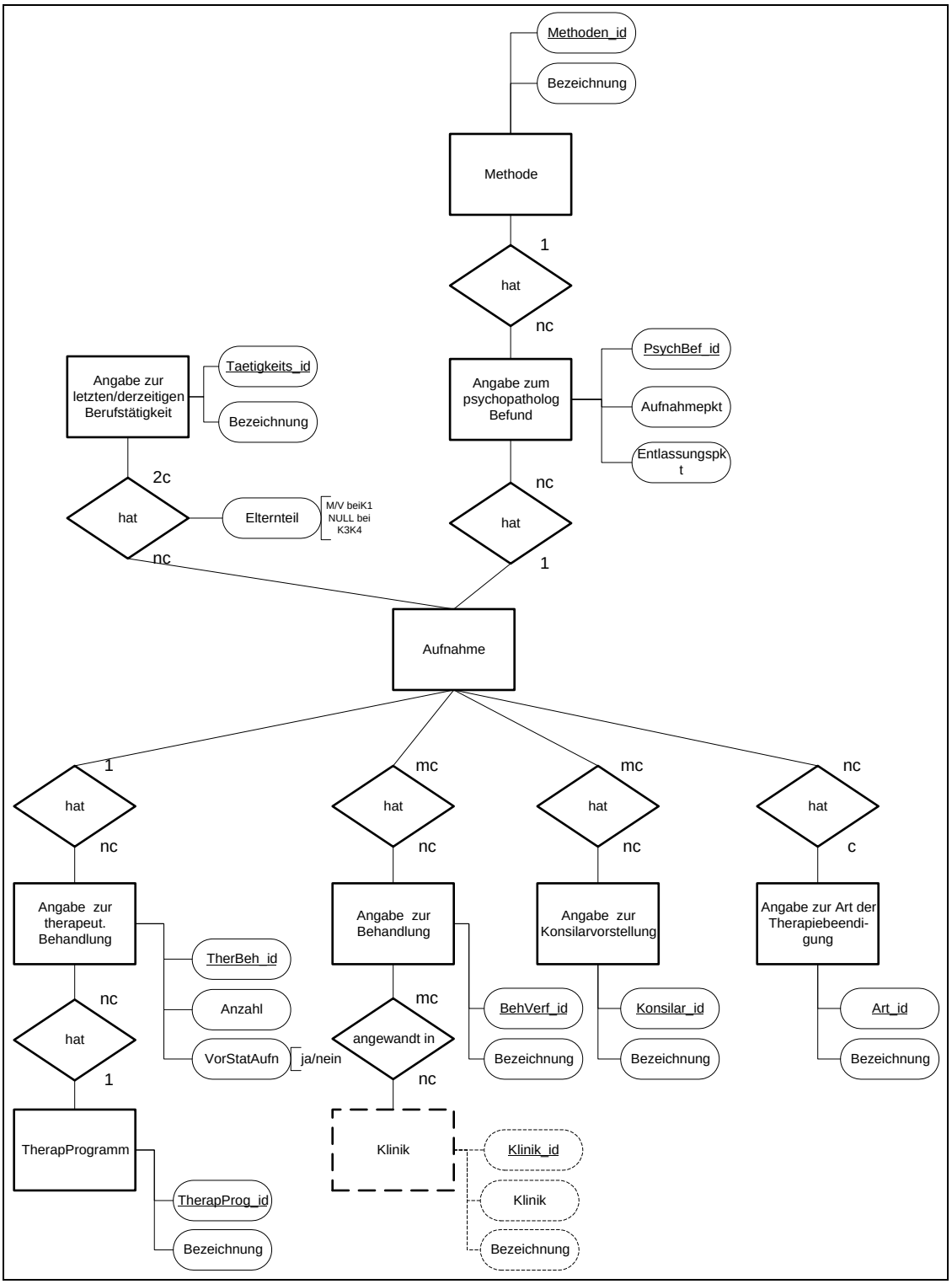
PDAZN Teilmodell Psychiatrie-Auszug 2



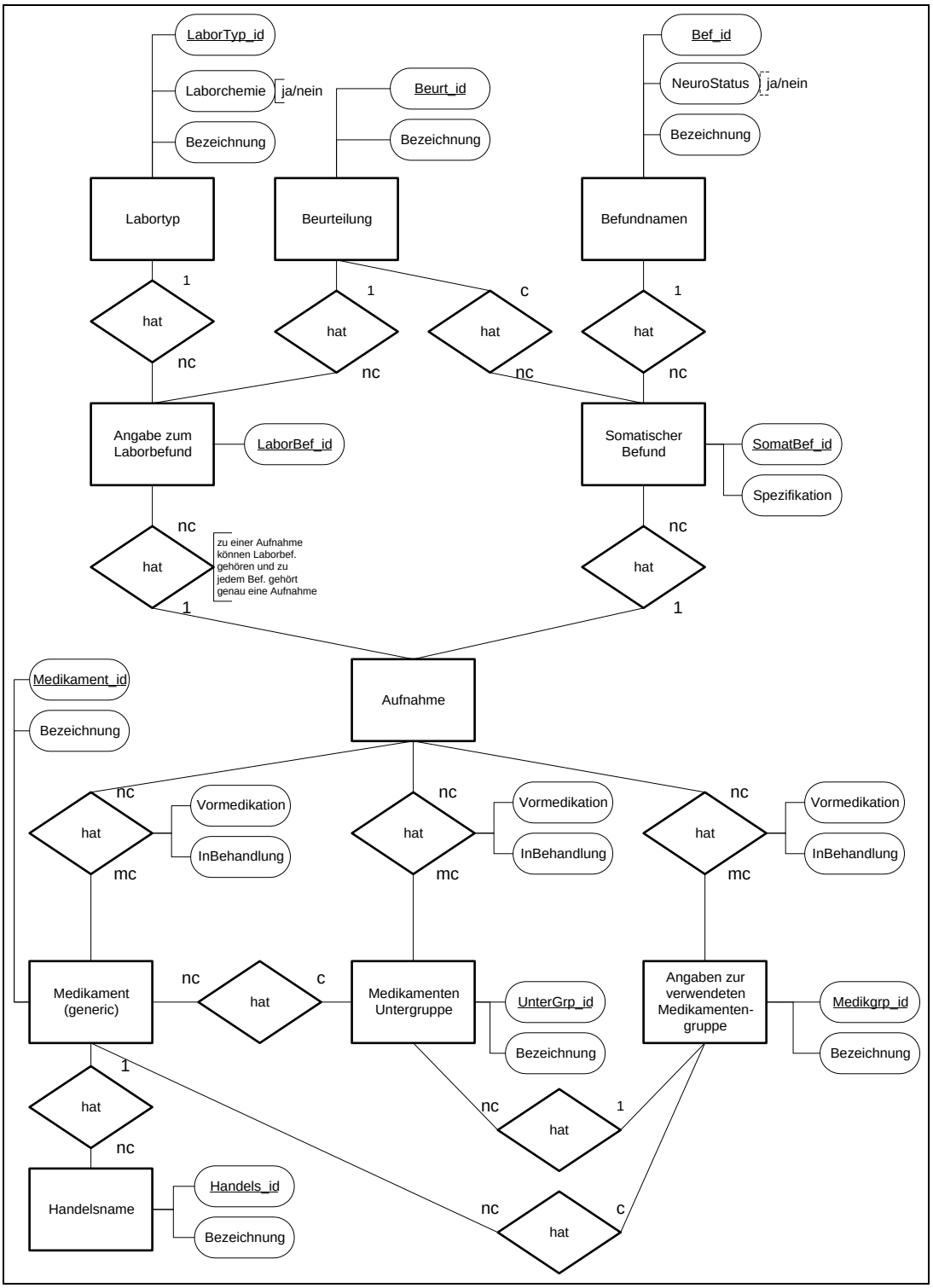
PDAZN Teilmodell Psychiatrie-Auszug 3



PDAZN Teilmodell Psychiatrie-Auszug 4

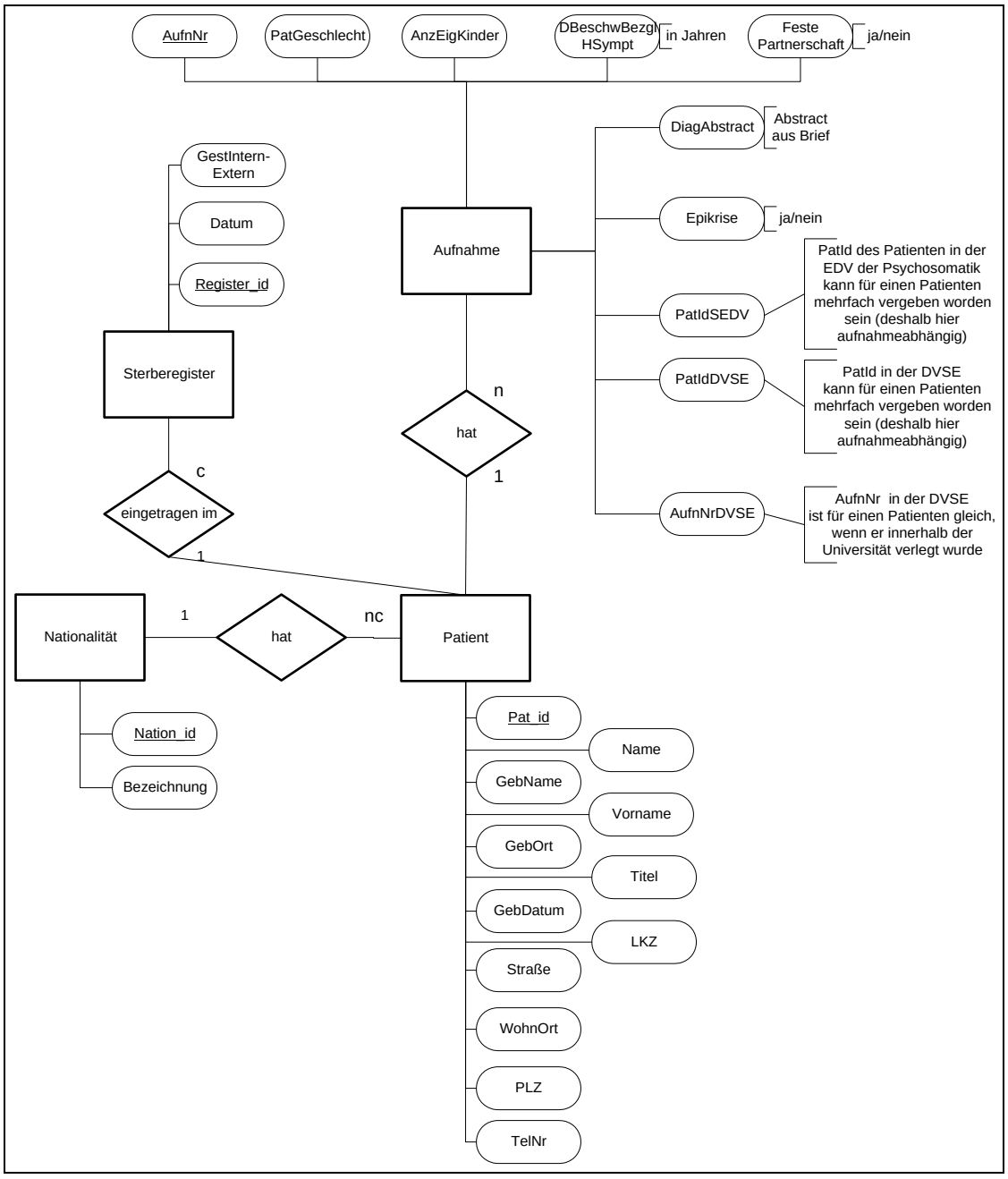


PDAZN Teilmodell Psychiatrie-Auszug 5

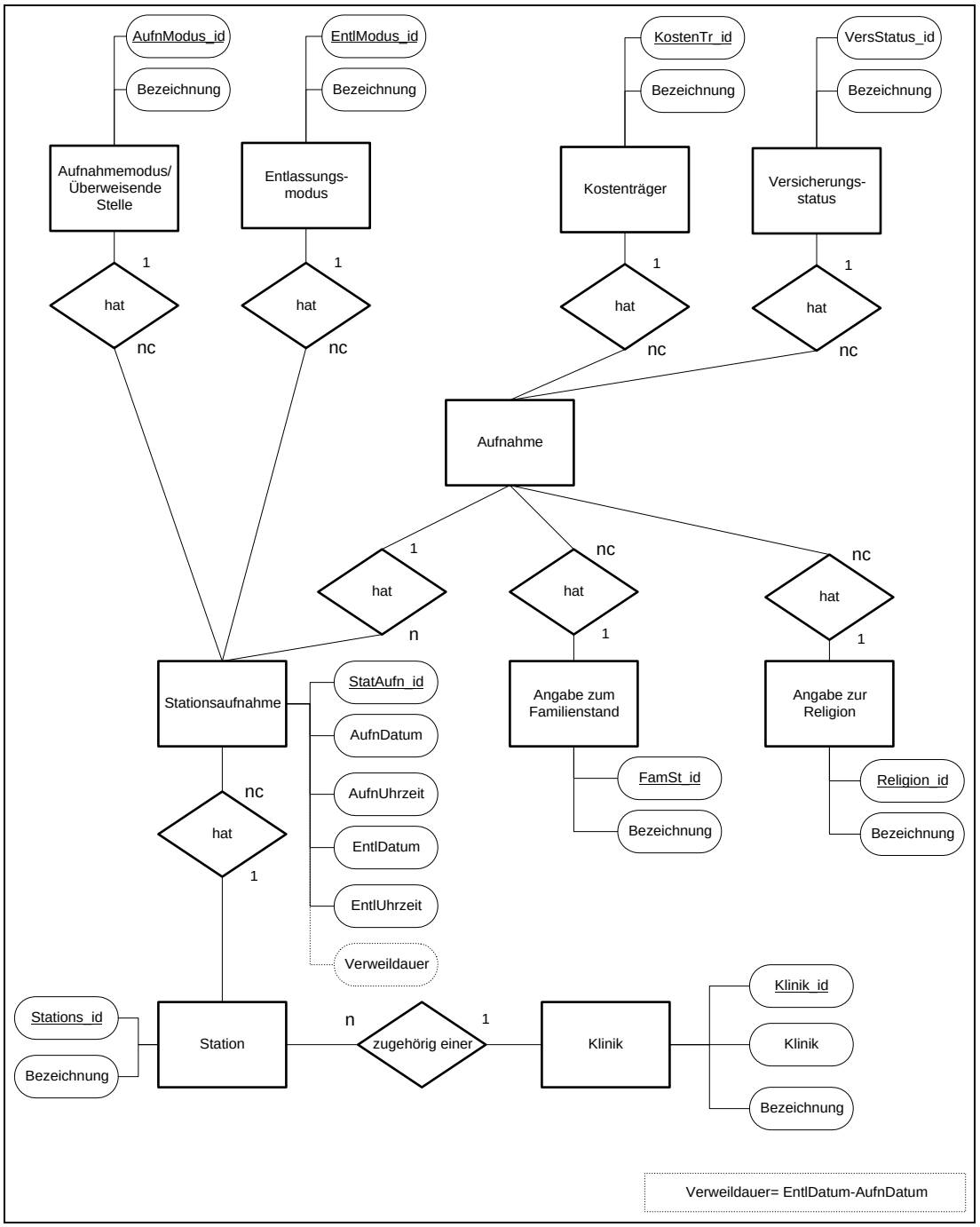


PDAZN Teilmodell Psychiatrie-Auszug 6

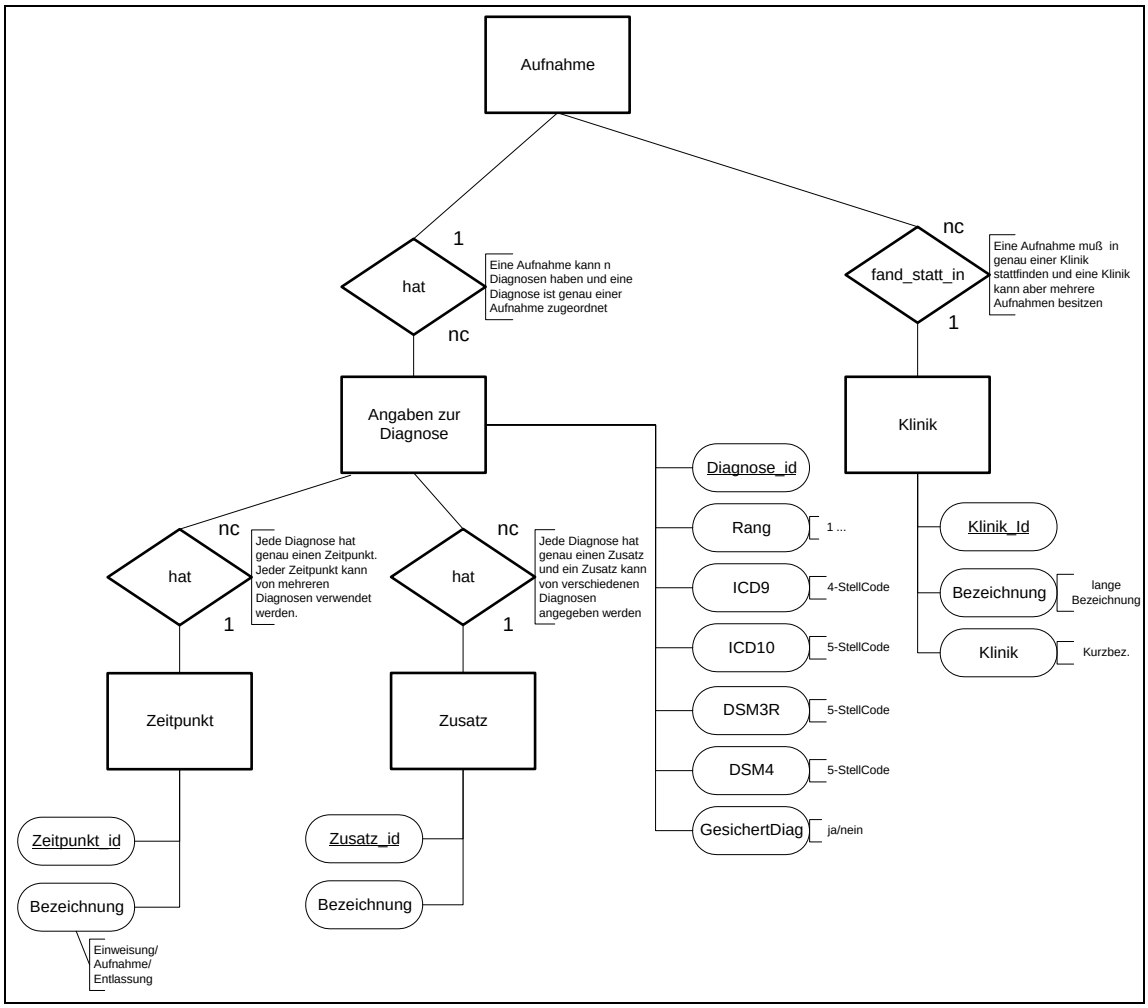
A.5 ER-Modell der Psychosomatik



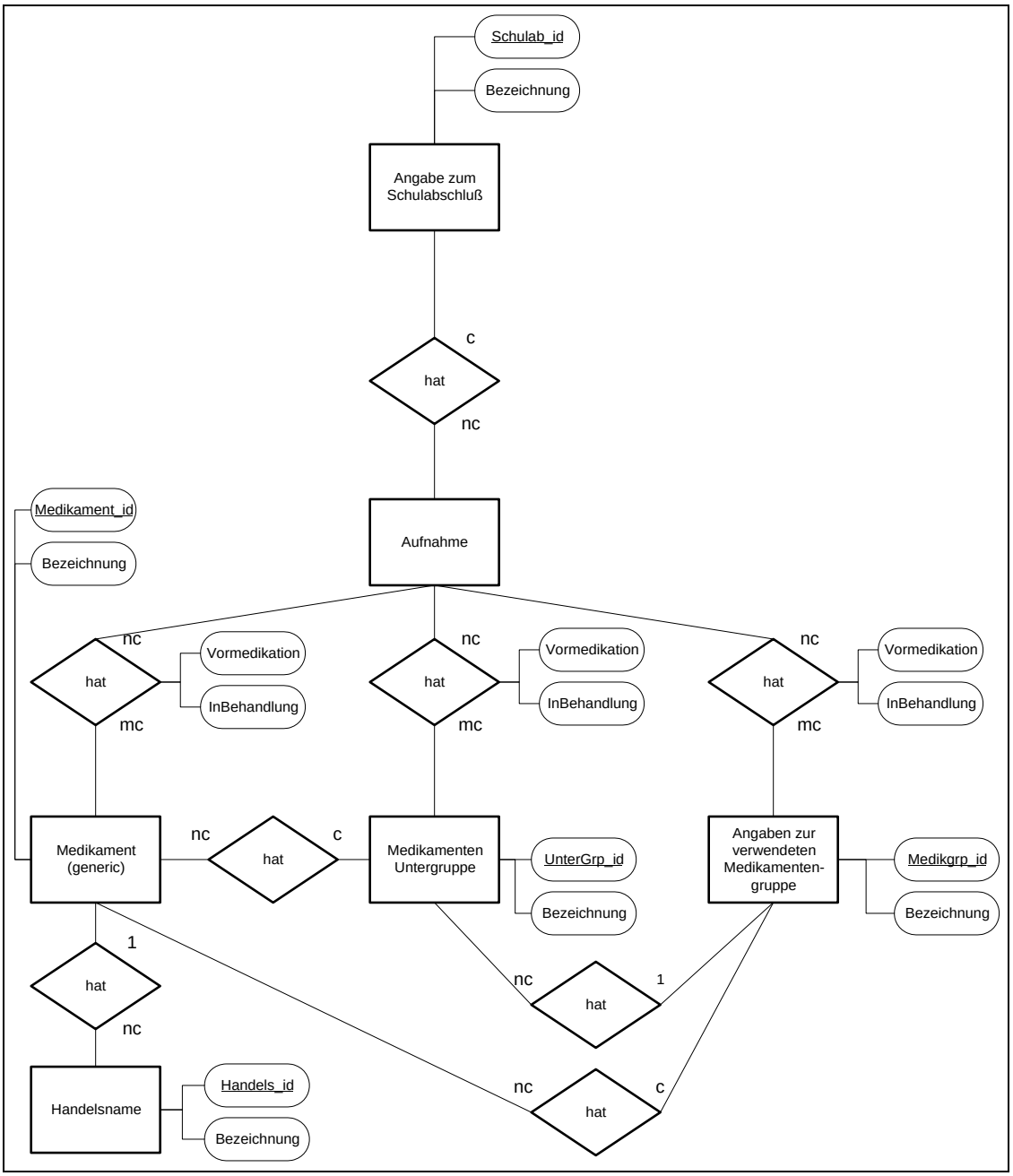
PDAZN Teilmodell Psychosomatik-Auszug 1



PDAZN Teilmodell Psychosomatik-Auszug 2



PDAZN Teilmodell Psychosomatik-Auszug 3



PDAZN Teilmodell Psychosomatik-Auszug 4

Anhang B

Relationenschemata

tblAufnahmen					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
AufnNr	Aufnahmenummer im PDAZN ist nicht konform mit der AufnNr im DVSE	integer		X	
PatGeschlecht	Geschlecht des Patienten zum Zeitpunkt der Aufnahme 0 männlich, 1 weiblich	bit	0/		
DBeschwBezglHSympt	**K4** Dauer (in Jahren) der Beschwerden bzgl. der Hauptsympthomatik	tinyint	0/		
FestePartnerschaft	**K4** Liegt eine feste Partnerschaft vor (true 1/false 0)	bit	0/		
AnzEigKinder	**K4** Anzahl der eigenen Kinder	tinyint	0/		
AnzKinder	**K3** Anzahl der Kinder	tinyint	0/		
CIDI	**K3** Gehört zum psychopatholog. Befund (true 1/false 0)	bit	0/		
PatIdSEDV	**K4** PatNr in EDV der Somatik (numerisch aktuell 5-stellig)	integer	/X		
PatIdDVSE	PatNr im DVSE (alphanumerisch z.Z. 7-stellig)	varchar(10)	/X		
PatIdKJN	**K1** Pat_id in der Kinder- und Jugendklinik	varchar(10)	/X		
AufnNrDVSE	Aufnahmenummer im DVSE	varchar(15)	/X		
Epikrise	Epikrise durchgeführt ?	bit			
Abstract	Auszug aus dem Arztbrief	text			
Pat_id	Patienten-Id	integer			**tblPatienten**
KostenTr_id	Id des Kostenträgers	tinyint			**tblKostenTr**
VersStatus_id	Versicherungsstatus-Id	tinyint			**tblVersStatus**
FamSt_id	Familienstand-Id	tinyint			**tblFamstaende**
Religion_id	Id der Religion	tinyint			**tblReligionen**
Klinik_id	Id der Klinik	tinyint			**tblKliniken**

PDAZN Relationenschemata-Teil 1

tblDiagnosen					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Diagnose_id	Id der Diagnose	integer		X	
Rang	Rang der Diagnose	tinyint			
ICD9	Code von ICD-9 (4-stell)	varchar(10)	/X		
ICD10	Code von ICD-10 (5-stell)	varchar(10)	/X		
DSM3R	Code von DSM3-R (5-stell)	varchar(10)	/X		
DSM4	Code von DSM4 (5-stell)	varchar(10)	/X		
GesichertDiag	Diagnose gesichert? (true 1/false 0)	bit	1/		
AufnNr	Aufnahmenummer	integer			**tblAufnahmen**
Zusatz_id	Zusatz-Id	tinyint			**tblZusaetze**
Zeitpunkt_id	Zeitpunkt-Id	tinyint			**tblZeitpunkte**

tblMedUnterGrpn					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
MedUnterGrp_id	Id der Medikamenten-untergruppe	smallint		X	
Bezeichnung	Bezeichnung der Medikamentenunter-gruppe	varchar(30)			
MedGrp_id	Id der Medikamenten-gruppe	smallint			**tblMedGrpn**

tblTherapEinzGespr					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
TherapEinzGespr_id	Id des therapeutischen Einzelgesprächs	integer		X	
Anzahl	Anzahl der Gespräche mit einer Zeitdauer von X Minuten	smallint	1		
Minuten	Anzahl der Minuten	smallint	10		
VorStatAufn	Fand das Gespräch vor der stationären Aufnahme statt 1 true, 0 false	bit	0		
AufnNr	Aufnahmenummer	integer			**tblAufnahmen**

tblTherapBehandl					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
TherapBeh_id	Id der therapeutischen Behandlung	integer		X	
Anzahl	Anzahl der Behandlungen	smallint	1		
VorStatAufn	Fand das Gespräch vor der stationären Aufnahme statt 1 true, 0 false	bit	0		
AufnNr	Aufnahmenummer	integer			**tblAufnahmen**
TherapProg_id	Id des therap. Programms	tinyint			**tblTherap-Prog**

PDAZN Relationenschemata-Teil 2

tblPsychopatBefunde					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
PsychBef_id	Id des psychopathologischen Befundes	integer		X	
Gesamtpkte	Gesamtpunkte	tinyint	/X		
Aufnahmepkte	Aufnahmepunkte	tinyint	/X		
Entlassungspkte	Entlassungpunkte	tinyint	/X		
AufnNr	Aufnahmenummer	integer			**tblAufnahmen**
Methoden_id	Id der Methode	tinyint			**tblPsychoPat-Meth**

tblSomatBefunde					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**
SomatTyp_id	Id des Typs	tinyint		X	**tblSomat-Typen**
Beurt_id	Id der Beurteilung	tinyint			**tblBeurtei-lungen**

tblPsychologBefunde					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**
Beurt_id	Id der Beurteilung	tinyint			**tblBeurteil-ungen**

tblStationsaufnahmen					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
StatAufn_id	Id der Stationsaufnahme	integer		X	
AufnDatum	Datum der stationären Aufnahme	datetimetime	/X		
AufnUhrzeit	Uhrzeit der stationären Aufnahme	time	/X		
EntlDatum	Datum der Entlassung	datetime	/X		
AufnMod_id	Aufnahmemodus-Id	tinyint			**tblAufnModi**
EntlMod_id	Entlassungsmodus-Id	tinyint			**tblEntlModi**
Stations_id	Id der Station	smallint			**tblStationen**
AufnNr	Aufnahmenummer	integer			**tblAufnahmen**

tblStationen					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Stations_id	Id der Station	smallint		X	
StationsNr	Stationsnummer einer Klinik	varchar(10)	/X		

tblFamstaende					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
FamSt_id	Familienstand-Id	tinyint		X	
Bezeichnung	Bezeichnung des Familienstandes	varchar(30)			

PDAZN Relationenschemata-Teil 3

tblPatienten					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Pat_id	Patienten-Id	integer		X	
Name	Name des Patienten	varchar(40)	/X		
GebName	Geburtsname	varchar(40)	/X		
Vorname	Vorname des Patienten	varchar(40)	/X		
Titel	Titel des Patienten	varchar(20)	/X		
GebOrt	Geburtsort	varchar(40)	/X		
GebDat	Geburtsdatum	datetime	/X		
Wohnort	Wohnort (Hauptwohnung)	varchar(40)	/X		
LKZ	Laenderkennzeichen	varchar(10)	/X		
PLZ	Postleitzahl	varchar(10)	/X		
Strasse	Straßenname und Hausnummer	varchar(40)	/X		
TelNr	Telefonnummer	varchar(20)	/X		
Nation_id	Nation-Id	tinyint			**tblNationali-taeten**
AnzNichttEltrVorK1	Anzahl der Aufnahmen in K1, die nicht elektronisch gespeichert sind	tinyint			
AnzNichttEltrVorK2	Anzahl der Aufnahmen in K2, die nicht elektronisch gespeichert sind	tinyint			
AnzNichttEltrVorK3	Anzahl der Aufnahmen in K3, die nicht elektronisch gespeichert sind	tinyint			
AnzNichttEltrVorK4	Anzahl der Aufnahmen in K4, die nicht elektronisch gespeichert sind	tinyint			

tblNationalitaeten					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Nation_id	Nation-Id	tinyint		X	
Bezeichnung	Nationalität z.B. auch türkisch/deutsch	varchar(30)			

tblReligionen					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Religion_id	Religion-Id	tinyint		X	
Bezeichnung	Religionsbezeichnung	varchar(30)			

tblAufnModi					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
AufnMod_id	Aufnahmemodus-Id	tinyint		X	
Bezeichnung	Bezeichnung des Aufnahmemodus bzw. der Überweisenden Stelle	varchar(50)			

PDAZN Relationenschemata-Teil 4

tblEntlModi					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
EntlMod_id	Entlassungsmodus-Id	tinyint		X	
Bezeichnung	Bezeichnung des Entlassungsmodus	varchar(50)			

tblKostenTr					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
KostenTr_id	Id des Kostenträgers	tinyint		X	
Bezeichnung	Bezeichnung des Kostenträgers	varchar(50)			

tblVersStatus					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
VersStatus_id	Versicherungsstatus-Id	tinyint		X	
Bezeichnung	Bezeichnung des Versicherungsstatus	varchar(1)			

tblZusaetze					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Zusatz_id	Zusatz-Id	tinyint		X	
Bezeichnung	Bezeichnung des Zusatzes bei der Diagnose	varchar(30)			

tblZeitpunkte					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Zeitpunkt_id	Zeitpunkt-Id	tinyint		X	
Bezeichnung	Bezeichnung des Zeitpunktes einer Diagnose	varchar(30)			

tblInfZurKlinikAufn					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Info_id	Id der Information	tinyint		X	
Bezeichnung	Bezeichnung der Information zur Klinikaufnahme	varchar(30)			

tblKliniken					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Klinik_id	Id der Klinik	tinyint		X	
Klinik	Kurzbezeichnung	varchar(20)			
Bezeichnung	Bezeichnung der Klinik	varchar(80)	/X		

PDAZN Relationenschemata-Teil 5

tblSchulabschluesse					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Schulabschl_id	Id des Schulabschlusses	tinyint		X	
Bezeichnung	Bezeichnung des Schulabschlusses	varchar/30			

tblBerufsabschluesse					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Berufsabschl_id	Id des Berufsabschlusses	tinyint		X	
Bezeichnung	Bezeichnung des Berufsabschlusses	varchar(30)			

tblUmfErwTaet					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
UmfErwTaet_id	Id des Umfangs der Erwerbstätigkeit	tinyint		X	
Bezeichnung	Bezeichnung des Umfangs der Erwerbstätigkeit	varchar(30)			

tblEinkQuellen					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
EinkQuelle_id	Id der Einkommensquelle	tinyint		X	
Bezeichnung	Bezeichnung der Einkommensquelle	varchar(30)			

tblBetreuungen					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Betr_id	Id der Betreuung	tinyint		X	
Bezeichnung	Bezeichnung der Betreuung	varchar(30)			

tblFamilienBelast					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
FamMitgl_id	Id der/des Familienmitglieder/es	tinyint		X	
Bezeichnung	Name des/der Mitglieds/er	varchar(30)			

tblZeitlEinordSuiz					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
SuizEinord_id	Id des Zeitraums der Suizidalität	tinyint		X	
Bezeichnung	zeitliche Einordnung der Suizidalität (momentan, Suizidversuch in Anamnese, nicht vorhanden)	varchar(30)			

PDAZN Relationenschemata-Teil 6

tblMedGrpn					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
MedGrp_id	Id der Medikamenten-gruppe	smallint		X	
Bezeichnung	Bezeichnung der Medikamentengruppe	varchar(30)			

tblMedikamente					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Medikament_id	Id des Medikamentes	smallint		X	
Bezeichnung	Bezeichnung des Medikamentes	varchar(30)			

tblBerufstaetigkeiten					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Taetigkeits_id	Id der Tätigkeit	tinyint		X	
Bezeichnung	Bezeichnung der Tätigkeit	varchar(30)			

tblPsychopatMeth					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Methoden_id	Id der Methode	tinyint		X	
Bezeichnung	Bezeichnung der Methode	varchar(30)			

tblTherapProg					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
TherapProg_id	Id des therap. Programms	tinyint		X	
Bezeichnung	Bezeichnung des therapeutisches Programms	varchar(30)			

tblBehandlungen					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Behandl_id	Id der Behandlung	tinyint		X	
Bezeichnung	Bezeichnung der Behandlung/Behand-lungsverfahren	varchar(30)			

tblKonsile					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Konsilar_id	Id des Konsilars	tinyint		X	
Bezeichnung	Bezeichnung des Konsilars	varchar(50)			

tblTherapieBeend					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
TherapBeend_id	Id der Beendigung	tinyint		X	
Bezeichnung	Art der Beendigung	varchar(30)			

PDAZN Relationenschemata-Teil 7

tblFamMilieu					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
FamMil_id	Id des Milieus	tinyint		X	
Bezeichnung	Bezeichnung des Milieus	varchar(30)			

tblStellKindGeschwR					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
StellKind_id	Id der Stellung in der Reihe	tinyint		X	
Bezeichnung	Bezeichnung der Stellung in der Geschwisterreihe	varchar(30)			

tblBehandlErfolge					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
BehandlErf_id	Id des Behandlungserfolges	tinyint		X	
Bezeichnung	Bezeichnung des Behandlungserfolges	varchar(30)			

tblSorgeberechtigte					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
SorgeBer_id	Id der Sorgeberechtigten	tinyint		X	
Bezeichnung	Bezeichnung der Sorge-berechtigten	varchar(30)			

tblVerlElternteil					
attribute	**Description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Verl_id	Id des Verlusteintrags	tinyint		X	
Bezeichnung	Bezeichnung des Verlustes	varchar(30)			

tblBerufsAusbStat					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
AusbStatus_id	Id des Ausbildungsstatus	tinyint		X	
Bezeichnung	Bezeichnung des Berufs-ausbildungsstatus	varchar(30)			

tblOekonFamStatus					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
OekFamStat_id	Id des ökonomischen Familienstatus	tinyint		X	
Bezeichnung	Bezeichnung des ökonomischen Familienstatus	varchar(30)			
Beschreibung	Beschreibung des ökonomischen Familienstatus	varchar(80)			

PDAZN Relationenschemata-Teil 8

tblDiagSymptomKat					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Katalog_id	Id des ergänzenden Eintrages	tinyint		X	
Bezeichnung	Bezeichnung des ergänzenden Eintrages	varchar(30)			

tblNeuroStatus					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
NeuroStatus_id	Id des neurologischen Status	tinyint		X	
Bezeichnung	Bezeichnung des neurologischen Status	varchar(30)			

tblAngKrampfAnf					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
AngKrAnf_id	Id der möglichen Angaben	tinyint		X	
Bezeichnung	Bezeichnung der Angaben (keine sicheren Angaben, ja, nein)	varchar(30)			

tblAngMenses					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
AngMenses_id	Id des Eintrages	tinyint		X	
Bezeichnung	Bezeichnung des Eintrages	varchar(30)			

tblSchuleBetreinr					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
SchuleEinri_id	Id der Schule/Betreuungsein-richtung	tinyint		X	
Bezeichnung	Bezeichnung der Schule bzw. Betreuungseinrichtung	varchar(30)			

tblIntelligenzEinstuf					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Einstuf_id	Id der Intelligenzein-stufung	tinyint		X	
Bezeichnung	Bezeichnung der Einstufung	varchar(30)			

tblEntwicklRueckst					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
EntwR_id	Id des Entwicklungsrück-standes	tinyint		X	
Bezeichnung	Bezeichnung des Ent-wicklungsrückstandes	varchar(30)			

PDAZN Relationenschemata-Teil 9

tblAuspTeilLeisStoer					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
AusprTeilSt_id	Id der Ausprägung von Teilleistungsstörung	tinyint		X	
Bezeichnung	Bezeichnung der Teilleistungsstörung	varchar(30)			

tblSomatTypen					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
SomatTyp_id	Id des Typs	tinyint		X	
Bezeichnung	Bezeichnung des Typs	varchar(30)			
Laborchemie	Gehört der Typ zur Laborchemie 1 true, 0 false	bit	0/		

tblBeurteilungen					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Beurt_id	Id der Beurteilung	tinyint		X	
Bezeichnung	Bezeichnung der Beurteilung	varchar(30)			

tblAngVerlFormen					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
VerlForm_id	Id der Verlaufsform	tinyint		X	
Bezeichnung	Bezeichnung der Verlaufsform	varchar(30)			

tblRegister					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Register_id	Id im Register	integer		X	
Datum	Sterbedatum	datetime			
GestInternExtern	Intern oder extern verstorben	bit			

tblAufnKlinikInf					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Info_id	Id der Information	tinyint			**tblInfZurKlinikAufn**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsSchulab					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Schulabschl_id	Id des Schulabschlusses	tinyint			**tblSchulabschluesse**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

PDAZN Relationenschemata-Teil 10

tblPatsBerufsab					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Berufsabschl_id	Id des Berufsabschlusses	tinyint			**tblBerufsab-schluesse**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsUmfErwT					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
UmfErwTaet_id	Id des Umfangs der Erwerbstätigkeit	tinyint			**tblUmfErw-Taet**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsEinkQ					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
EinkQuelle_id	Id der Einkommensquelle	tinyint			**tblEink-Quellen**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsBetreu					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Betr_id	Id der Betreuung	tinyint		X	**tblBetreu-ungen**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsBelFamMitgl					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
FamMitgl_id	Id der/des Familienmitglieder/es	tinyint		X	**tblFamilien-Belast**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsSuizid					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
SuizEinord_id	Id des Zeitraums der Suizidalität	tinyint		X	**tblZeitlEinord Suiz**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsMedGrp					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
MedGrp_id	Id der Medikamenten-gruppe	smallint		X	**tblMedGrpn**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**
Vormedikation	Fand diese Medikation als Vormedikation statt (true =1, false = 0)	bit			
InBehandlung	Fand diese Medikation in der Behandlung statt (true =1, false = 0)	bit			

PDAZN Relationenschemata-Teil 11

tblMGrpHATMedik					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Medikament_id	Id des Medikamentes	smallint		X	**tblMedikamente**
MedGrp_id	Id der Medikamentengruppe	smallint			**tblMedGrpn**

tblMUntGrpHAT-Medik					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Medikament_id	Id des Medikamentes	smallint		X	**tblMedikamente**
MedUnterGrp_id	Id der Medikamentenuntergruppe	smallint			**tblMedUnterGrpn**

tblPatsBehandl					
attribute	**Description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Behandl_id	Id der Behandlung	tinyint		X	**tblBehandlungen**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsKonsile					
attribute	**Description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Konsilar_id	Id des Konsilars	tinyint		X	**tblKonsile**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsTherapBeend					
attribute	**Description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
TherapBeend_id	Id der Beendigung	tinyint			**tblTherapieBeend**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsBerufsTaetigk					
attribute	**Description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Taetigkeits_id	Id der Tätigkeit	tinyint		X	**tblBerufstaetigkeiten**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**
Elternteil	K1: Elternteil des Patienten (M/V) K3: Wenn eigene Tätigkeit: NULL-Wert	varchar(1)	/X		

PDAZN Relationenschemata-Teil 12

tblPatsFamMil					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
FamMil_id	Id des Milieus	tinyint			**tblFamMilieu**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsStGeschwReih					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
StellKind_id	Id der Stellung in der Reihe	tinyint			**tblStellKind-GeschwR**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsBehandlErfolg					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
BehandlErf_id	Id des Behandlungserfolges	tinyint			**tblBehandl-Erfolge**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsSorgeBer					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
SorgeBer_id	Id der Sorgeberechtigten	tinyint			**tblSorgebe-rechtigte**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsVerlustEltern					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Verl_id	Id des Verlusteintrags	tinyint			**tblVerlEltern-teil**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsBerufAusStElt					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
AusbStatus_id	Id des Ausbildungsstatus	tinyint		X	**tblBerufs-AusbStat**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**
Elternteil	Elternteil des Patienten (M/V)	varchar(1)			

tblPatsOekFamStat					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
OekFamStat_id	Id des ökonomischen Familienstatus	tinyint			**tblOekon-FamStatus**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

PDAZN Relationenschemata-Teil 13

tblPatsDiagSymptomKat					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Katalog_id	Id des ergänzenden Eintrages	tinyint		X	**tblDiagSymptomKat**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsNeuroStat					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
NeuroStatus_id	Id des neurologischen Status	tinyint		X	**tblNeuroStatus**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsKrampfAnf					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
AngKrAnf_id	Id der möglichen Angaben	tinyint			**tblAngKrampfAnf**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsMensesAngab					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
AngMenses_id	Id des Eintrages	tinyint			**tblAngMenses**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsZulBesSchul					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
SchuleEinri_id	Id der Schule/Betreuungseinrichtung	tinyint			**tblSchuleBetrEinr**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsIntelligenz					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Einstuf_id	Id der Intelligenzeinstufung	tinyint			**tblIntelligenzEinstuf**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsEntwRueckst					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
EntwR_id	Id des Entwicklungsrückstandes	tinyint			**tblEntwicklRueckst**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblPatsAusprTeilleist					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
AusprTeilSt_id	Id der Ausprägung von Teilleistungsstörung	tinyint			**tblAuspTeilLeisStoer**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

PDAZN Relationenschemata-Teil 14

tblPatsVerlaufssForm					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
VerlForm_id	Id der Verlaufsform	tinyint			**tblAngVerl-Formen**
AufnNr	Aufnahmenummer	integer		X	**tblAufnahmen**

tblVerstorben					
attribute	**description**	**type**	**default/ NULL**	**prim_key**	**foreign_key**
Register_id	Id im Register	integer			**tblRegister**
Pat_id	Patienten-ID	integer		X	**tblPatienten**

PDAZN Relationenschemata-Teil 15

Erklärung

Ich erkläre, daß ich die vorliegende Arbeit selbständig und nur unter Vorlage der angegebenen Literatur und Hilfsmittel angefertigt habe.

Steffen Murr Sanitz, 15. Juni 1998

Thesen

1. Die entwickelte Applikation PDAZN ist in den Client/Server-Architekturtyp "Verteilte Applikations-Logik" einzuordnen. Dadurch wird eine Verteilung der Gesamtaufgabe auf Client und auf Server erreicht.

2. Ein hervorragendes graphisches Front-End stellt MS Access 97 dar. Für das Back-End kommt der MS SQL-Server zum Einsatz, der ein Sicherheitskonzept bietet, das der Sensibilität der Daten angemessen ist.

3. Die Entwicklung einer Client/Server-Applikation erfordert, die Belastung des Netzwerkes in den Mittelpunkt zu stellen. Das Upsizing einer Desktop- zu einer Client/Server-Anwendung ist aufgrund dessen sehr schwierig.

4. Die Einführung des relationalen Datenbanksystems im Archiv des Zentrums für Nervenheilkunde, bringt eine wesentliche Erleichterung der Archivierungarbeit mit sich. Es werden nicht alle Daten der Patientenakte gespeichert, sondern nur bestimmte Archiv- und Klinik-spezifische Informationen.

5. Durch die entwickelte Patientendatenverwaltung stehen den Archivmitarbeitern eine Vielzahl von Funktionen zur Wartung der Aufnahmedaten zur Verfüng.

6. Die Pflege spezieller Wertebereiche wird durch die "Allgemeine Datenpflege" gewährleistet. Diese Wertebereiche garantieren die Verwendung eines einheitlichen Vokabulars, das insbesondere bei der Suche von Informationen notwendig ist.

7. Der Import von Patientendaten aus dem Datenarchiv der Medizinischen Fakultät, stellt dem Archiv wesentlich mehr Informationen, als vor Einführung des PDAZN zur Vefügung und erleichtert die Archivierungsarbeit erheblich. Eine Automatisierung des Imports ist aufgrund des Zuordnungsproblems von Patienten neuer Aufnahmen, zu bereits in der Datenbank gespeicherten Patienten, nicht möglich.

8. Ein Statistikteil erlaubt die Beantwortung von häufig gestellten Anfragen an das Archiv, auf der Grundlage der gespeicherten Informationen.

9. Die entwickelte Applikation ist außer einem Archivierungs- auch ein Informationssystem. Durch entsprechende Rechtevergabe können Mitarbeiter der Kliniken auf Informationen von Patientenaufnahmen zugreifen.

10. Die bestehende Client/Server-Architektur läßt bei Bedarf die Erweiterung zu einer "Verteilten Datenbank" zu. Das eingesetzte DBMS unterstützt diesen Architekturtyp. Die Möglichkeit ist in Betracht zu ziehen, wenn alle Daten einer Patientenakte gespeichert werden sollen.