SFB-Bericht 27/88, SFB 124, University of Kaiserslautern, 1988

Ein Modell zur Integration der Zeit in relationale Datenbanksysteme

Wolfgang Käfer
Universität Kaiserslautern

SFB 124
Bericht-Nr. 27/88
Universität Kaiserslautern
Erwin-Schrödinger-Straße
6750 Kaiserslautern

Überblick

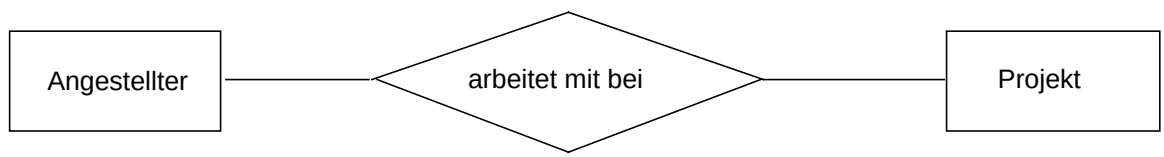
Bei der Darstellung eines Ausschnitts der realen Welt (Miniwelt), der mit einem Datenbanksystem (DBS) nachgebildet werden soll, wurde bisher meist auf den Aspekt der "Zeit" nicht eingegangen. Im Zuge der Entwicklung von DBS für sogenannte Non-Standard-Anwendungen wurden erste Schritte mit diesem Ziel unternommen. Dabei zeigte sich, daß eine Übernahme des uns geläufigen Zeitbegriffs in die Darstellungsschemata eines DBS gewisse Schwierigkeiten in sich birgt.

Die vorliegende Arbeit untersucht und präzisiert deshalb zunächst den Begriff "Geschichte" als zeitliche Folge von Objektzuständen. Es wird gezeigt, daß mit diesem Konzept auch kontinuierliche Geschichte näherungsweise dargestellt werden kann. Nach einer einfachen Klassifizierung von zeitbehafteten Anfragen werden Konstrukte zur Unterstützung der einzelnen Anfragearten vorgestellt. Abschließend wird ein SQL-basierter Sprachvorschlag entworfen, dessen Brauchbarkeit an einem Beispiel demonstriert wird.

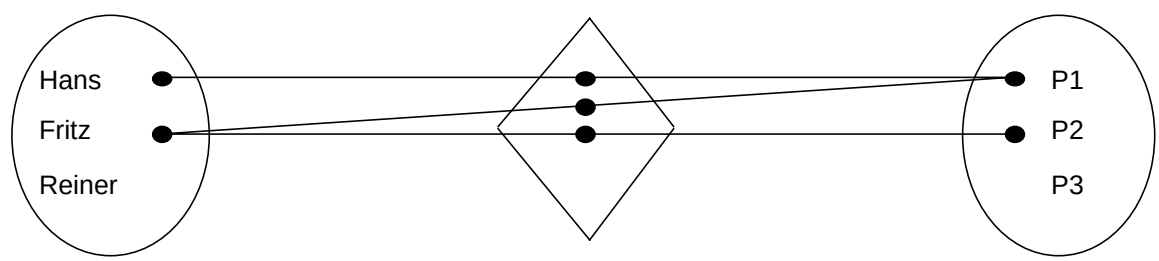
1. Das Geschichtsmodell

1.1 Was ist "Geschichte" ?

Bei der Bildung unseres Geschichtsbegriffs wird vom Entity-Relationship-Modell (E/R-Modell) /Ch76/ ausgegangen. Dieses Modell sieht vor, daß ein "Gegenstand von Interesse" der realen Welt, ein Entity, durch eine Menge von Merkmalen beschrieben und identifiziert wird. Alle Entities, die durch die gleichen Merkmale beschrieben werden können, bilden einen Entity-Typ. Alle Ausprägungen eines Typs können zu einer Entity-Menge zusammengefaßt werden. Beziehungen zwischen Entities, genauer: zwischen Entity-Mengen, werden als Relationships bezeichnet. Analog zu den Entity-Typen und -Mengen können Relationship-Typen und Relationship-Mengen gebildet werden. Weiterhin können auch Relationships durch Merkmale charakterisiert werden. Bild 1a zeigt ein E/R-Diagramm für die Mitarbeit von Angestellten bei Projekten. In Bild 1b sind einige Ausprägungen graphisch dargestellt. Dabei wird deutlich, daß nur ein Schnappschuß der Miniwelt erfaßt ist - die zeitliche Veränderung der Entities und Relationships ist nicht repräsentiert. Soll dies geschehen, muß zunächst der Begriff der Zeit näher charakterisiert werden. Die Zeit soll so in das Modell übernommen werden, wie sie "uns" geläufig ist, d.h., sie wird mit Kalender und Uhr gemessen. Im Gegensatz zu technischen oder mathematischen Zeitbegriffen, die Zeit meist nur als (un)endliche Folge von (halb)geordneten Zeitpunkten betrachten, gibt es **feste Zeiteinheiten** mit einer minimalen Granularität (Sekunden, Minuten, Stunden, Tage, Wochen, ..) und auch der Begriff einer "**globalen gemeinsamen Zeit**" bereitet keine Probleme - man verwendet beispielsweise die "Greenwich Time".



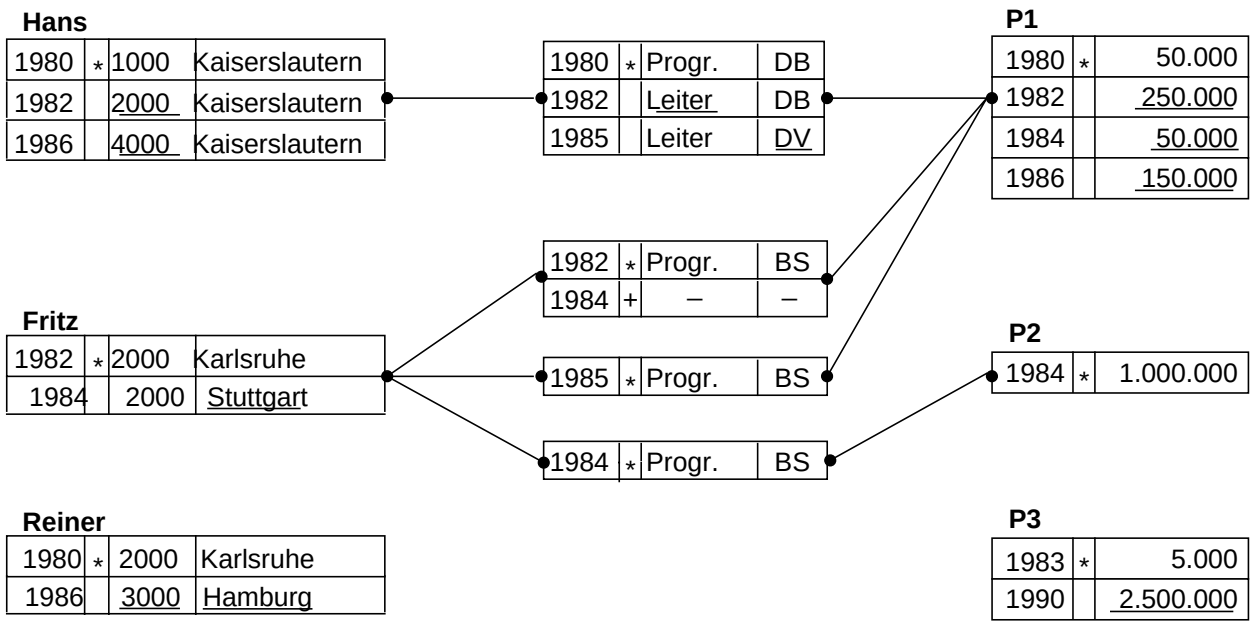
a) E/R - Diagramm (Primärschlüsselattribute sind durch Fettdruck gekennzeichnet.)



b) Einige Ausprägungen

Bild 1: Mitarbeit von Angestellten bei Projekten

Innerhalb dieser gemeinsamen Zeit können sich die Entities verändern oder genauer: die Merkmalswerte der Entities bzw. der Relationships können ihre Werte ändern, Entities bzw. Relationships können neu entstehen



a) Geschichte der Angestellten, Projekte und der Mitarbeit von Angestellten bei Projekten

1980	*	**Hans**	1000	Kaiserslautern	*	Progr.	DB	*	**P1**	50.000
	*	**Reiner**	2000	Karlsruhe		–	–		–	–
1982		**Hans**	2000	Kaiserslautern		Leiter	DB		**P1**	250.000
	*	**Fritz**	2000	Karlsruhe	*	Progr.	BS		**P1**	250.000
1984		**Hans**	2000	Kaiserslautern		Leiter	DB		**P1**	50.000
		Fritz	–	–	+	–	–		**P1**	–
		Fritz	2000	Stuttgart	*	Progr.	BS	*	**P2**	1.000.000
1985		**Hans**	2000	Kaiserslautern		Leiter	DV		**P1**	50.000
		Fritz	2000	Stuttgart	*	Progr.	BS		**P1**	50.000
1986		**Hans**	4000	Kaiserslautern		Leiter	DV		**P1**	150.000
		Fritz	–	–	+	–	–		**P2**	–
		Reiner	3000	Hamburg		–	–		–	–
1988		–	–	–		–	–	*	**P3**	5.000
1990		–	–	–		–	–		**P3**	2.500.000

b) Geschichten der Objekte " Angestellter - arbeitet mit bei - Projekt "

Legende:
*: Geburt
+: Tod
-: unbesetzt
_: Änderung

Der Zeitwert hat die Bedeutung "gilt ab". Das Ende der Gültigkeit ist implizit durch den Zeitwert der nächst neueren Aussage bestimmt. Aus diesem Grund ist es notwendig, daß für den Tod eines Objektes ein Zeitwert ohne jede weitere Information abgespeichert wird /MS83/. Dieser Sachverhalt wird durch die "-"-Zeichen dargestellt. Änderungen von Merkmalswerten werden durch Unterstreichung verdeutlicht. Die Primärschlüssel sind durch Fettdruck dargestellt.

Bild 2: Geschichte der Angestellten, Projekte und Mitarbeit der Angestellten bei Projekten

oder bestehende können entfernt werden. Zeichnet man die Änderungen auf, so entsteht eine Geschichte der Miniwelt bzw. die Geschichte eines jeden Entities bzw. einer jeden Relationship innerhalb der Miniwelt. Bild 2 zeigt die Entities und Relationships aus Bild 1, allerdings in ihrer zeitlichen Entwicklung. In Bild 2a wurde die Geschichte eines jeden Entities und einer jeden Relationship separat aufgezeichnet: die Relationship "arbeitet mit bei", identifiziert durch "Hans" und "P1", kann direkt in ihrer zeitlichen Entwicklung betrachtet werden. Diese Darstellung ist jedoch ungeeignet, wenn ein Überblick über größere Teile oder über die gesamte Miniwelt benötigt wird, beispielsweise derart: Wer arbeitete 1984 bei welchem Projekt (in welcher Funktion) mit? Bild 2b zeigt eine Darstellung, in der die Frage leicht beantwortet werden kann. Die Verwendung einer globalen gemeinsamen Zeit erlaubt es, die Darstellungen ineinander überzuführen.

Die in Bild 2a dargestellten Tabellen beschreiben die Ausprägungen der **zeitbehafteten** Entity- und Relationship-Typen aus Bild 1, in dem der aktuelle (neueste) Zustand dargestellt worden war. Insbesondere sei darauf hingewiesen, daß lediglich mehr Informationen zu den schon in Bild 1 vorhandenen Entities und Relationships aufgezeichnet wurde, daß aber keine neuen Entities oder Relationships hinzugefügt wurden. Dies ist ein grundsätzlich anderes Vorgehen als beispielsweise in /Fe85/. Dort werden zusätzliche Relationship-Typen eingeführt, um den Aspekt der Zeit zu modellieren. In dem hier gewählten Ansatz soll die Darstellungsmächtigkeit der Entity- und Relationship-Typen erhöht werden. Zur besseren Unterscheidung werden deshalb die zeitbehafteten Entity- und Relationship-Ausprägungen als **Geschichtsausprägungen** bezeichnet. Jede Geschichtsausprägung besteht aus einer oder mehreren **Aussagen**, die den **Zustand** einer Entity- oder Relationship-Ausprägung zu einem Zeitpunkt beschreiben, d.h., eine Aussage entspricht einer herkömmlichen Entity- oder Relationship-Ausprägung. Trägt man alle Aussagen zu einer Entity-Ausprägung in der

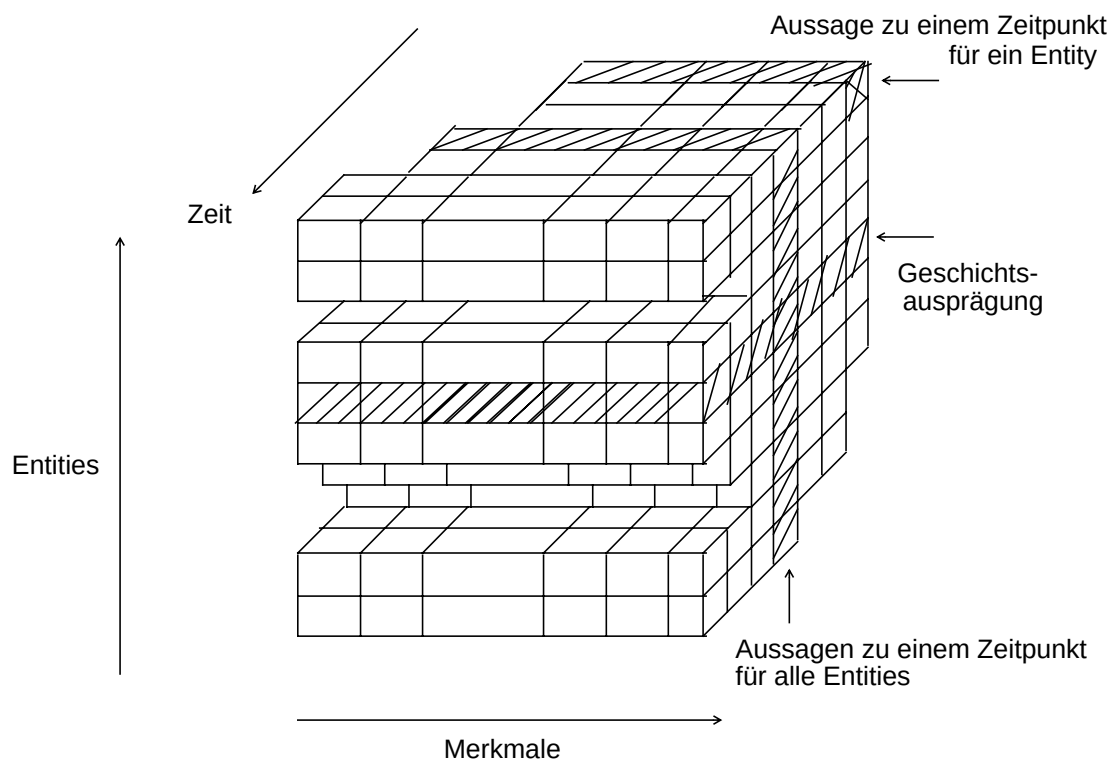


Bild 3: "Geschichtswürfel"

dritten Dimension eines ansonsten zweidimensionalen Schemas von Entity-Relationen auf, so ergibt sich der zur Veranschaulichung in Bild 3 dargestellte "Geschichtswürfel".

Diese abstrakte Sichtweise der Geschichte legt eine Erweiterung nahe: warum soll die Folge der Aussagen in der Gegenwart enden? Durch die Hinzunahme von zukünftigen Aussagen umfaßt die abstrakte Geschichte sowohl die Vergangenheit und die Gegenwart als auch die Zukunft. Zur Adaption dieser drei Begriffe muß allerdings eine fiktive Konstante **NOW** eingeführt werden, die den aktuellen Zeitpunkt, die Gegenwart, symbolisiert. Durch das ständige Fortschreiten dieser "Konstanten" wird die Geschichtsentwicklung, d.h. das "Wandern" von Aussagen aus der Zukunft über die Gegenwart in die Vergangenheit, realisiert. Der Ausdruck "Konstante" ist trotzdem sinnvoll, da für die Dauer der Auswertung der mit NOW bestimmte Zeitpunkt als Gegenwart betrachtet wird, auch wenn die Auswertung eine meßbare Zeitdauer besitzt. Anders ausgedrückt: die Evaluierung von NOW zu einem bestimmten Zeitpunkt friert die Datenbank für die Dauer der aktuellen Auswertung ein. Nachdem geklärt wurde, was unter "Geschichte" zu verstehen ist, soll im folgenden die Geschichtsschreibung untersucht werden.

1.2 Wie kann Geschichte dargestellt werden ?

Bei der Betrachtung von Geschichte lassen sich vier verschiedene Geschichtsarten unterscheiden. Im folgenden sollen diese Geschichtsarten kurz anschaulich erläutert werden (vgl. /Hä84, Kl83/).

Bei der ersten Art von Geschichte bleibt eine Aussage solange gültig, bis eine neue Aussage bekannt wird, d.h., ein Zustand bleibt solange gültig, bis er durch ein neues Ereignis invalidiert wird. Man spricht deshalb von **zustandserhaltender Geschichte** (Bild 4a). Die Zeit-Werte-Funktion stellt eine Treppenfunktion dar. Die in Kapitel 1.1 verwendeten Beispiele waren von diesem Geschichtstyp.

Bei der zweiten Art von Geschichte ändert sich der Zustand (kontinuierlich) mit der Zeit. Die Art der Änderung ist dabei nicht durch Ereignisse oder andere erkennbare Faktoren bestimmt. Der Ausdruck **zustands-**

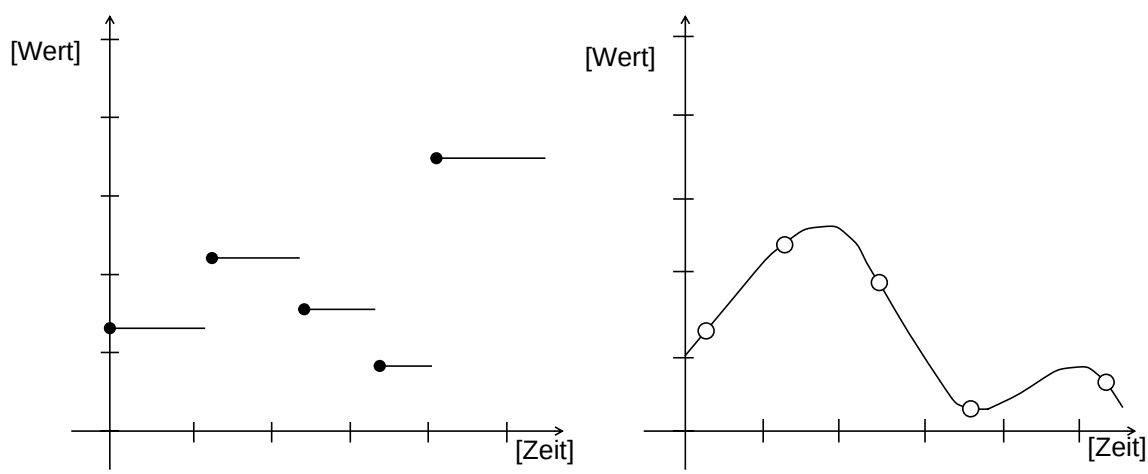


Bild 4: a) zustandserhaltende Geschichte b) zustandsverändernde Geschichte

verändernde Geschichte (Bild 4b) charakterisiert diesen Sachverhalt. Die Zeit-Werte-Funktion kann als stetige Funktion betrachtet werden.

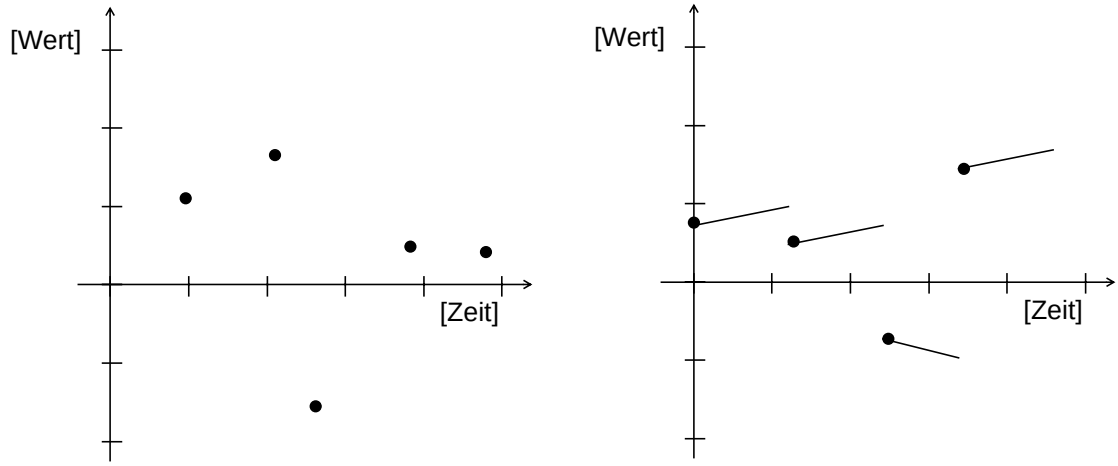


Bild 5: a) ereignisorientierte Geschichte b) ableitbare Geschichte

Die dritte Art von Geschichte beschreibt Zustände, die nicht wie bisher über einen ganzen Zeitraum gelten, sondern nur zu bestimmten Zeitpunkten Werte annehmen, d.h., es werden nur Folgen von Ereignissen betrachtet. Der Begriff **ereignisorientierte Geschichte** (Bild 5a) liegt deshalb nahe. Der Verlauf kann als diskrete Sprungfunktion dargestellt werden.

Die vierte und letzte Art von Geschichte unterscheidet sich in ihrer mathematischen Nachbildung von den vorhergehenden Geschichtsarten dadurch, daß die Funktion, nach der sich die Änderung vollzieht, bekannt ist. Allein durch die Kenntnis der Funktion kann der Zustand zu jedem beliebigen Zeitpunkt bestimmt werden. Man spricht deshalb auch von **ableitbarer Geschichte** (Bild 5b). Die Art der Funktion (Treppenfunktion, stetige Funktion, Sprungfunktion) ist dabei prinzipiell ohne Bedeutung.Bei der Untersuchung von zeitlichen Abläufen zeigt sich, daß die oben genannten vier Gechichtsarten oft nicht in ihrer reinen Form, sondern in Mischformen auftreten. Beispielsweise ist die Geschichte eines Kontos eine Überlagerung einer ableitbaren Geschichte (Zinsen) mit einer ereignisorientierten Geschichte (Ein-/Auszahlungen). Bei dieser Art der Überlagerung werden mathematisch gesehen zwei Zeit-Wert-Funktionen addiert. Es ist jedoch auch eine andere Art der Überlagerung denkbar, die beispielsweise bei physikalischen Versuchen auftreten könnte. Hier werden nicht zwei Funktionen addiert, sondern es werden mehrere zeitlich lokale Funktionen zu einer größeren zeitlich lokalen oder auch zu einer zeitlich totalen Zeit-Wert-Funktion aneinandergesetzt. Dies ist z.B. sinnvoll, wenn das beobachtete Phänomen sich in großen Zeitabschnitten entsprechend einer gegebenen Funktion verhält (ableitbare Geschichte) und nur in bestimmten Zeitabschnitten von dieser Funktion abweicht (zustandsverändernde Geschichte)

Alle vier Geschichtsarten lassen sich trotz ihres heterogenen Aussehens auf das in Kapitel 1.1 vorgestellte abstrakte Geschichtsbild zurückführen. Allen Geschichtsarten gemeinsam ist die Tatsache, daß das Objekt zu jedem beliebigen Zeitpunkt durch seinen momentanen Zustand, also durch eine Aussage, beschrieben werden kann. Im Falle der ereignisorientierten Geschichte entspricht jede Aussage einem Ereignis in der Miniwelt. Bei allen anderen Geschichtsarten bestehen die Geschichtsausprägungen aus Aussagen zu besonders **charakteristischen Zuständen** z.B. direkt nach einer Zustandsänderung, aus denen mit

geeigneten Funktionen alle anderen Zustände ("zu jedem beliebigen Zeitpunkt") errechnet oder zumindest näherungsweise bestimmt werden können.

Zur Illustration der obigen Diagramme und zum besseren Verständnis wird im folgenden zu jeder Geschichtsart ein Beispiel vorgestellt:

- zustandserhaltende Geschichte (z.B. Abteilungszugehörigkeit)
  Als charakteristische Aussagen werden sinnvollerweise die Werte direkt nach Zustandsübergängen verwendet. Das Ergebnis der Auswertefunktion ergibt sich somit aus der jüngsten Aussage der Geschichtsausprägung, deren Gültigkeitszeit gleich oder kleiner als der Anfragezeitpunkt ist.
- zustandsverändernde Geschichte (z.B. Fieberkurve)
  Die charakteristischen Aussagen und die Auswertefunktion müssen vom Benutzer dem DBS zur Verfügung gestellt werden, d.h., es werden eine Menge von Meßwerten (zu bestimmten Zeitpunkten) abgespeichert. Mit Hilfe einer vom Benutzer zur Verfügung gestellten Funktion können dann vom DBS aus diesen Meßwerten (und dem Zeitpunkt der Messung) die Funktionswerte zu beliebigen Zeitpunkten approximiert werden.
- ereignisorientierte Geschichte (z.B. Konto-Ein- und -Auszahlungen)
  Bei der ereignisorientierten Geschichte werden die Ereignisse als charakteristische Aussagen verwendet. Die Aufgabe der Auswertefunktion beschränkt sich damit auf das Selektieren einer durch ihre Gültigkeitszeit bestimmten Aussage aus einer Geschichtsausprägung.
- ableitbare Geschichte (z.B. Kontostand)
  Im Falle des Kontostandes kann aus dem gegebenen Zeitpunkt, den seit diesem Zeitpunkt gültigen Zinssätzen, dem "bisherigen" Kontostand und einer "Zinsformel" der exakte Kontostand ermittelt werden. Im Gegensatz zur zustandsverändernden Geschichte kann der gesamte Geschichtsverlauf "exakt" berechnet werden. Es ist insbesondere nicht notwendig, daß der Kontostand quasi stichprobenweise aufgezeichnet wird.

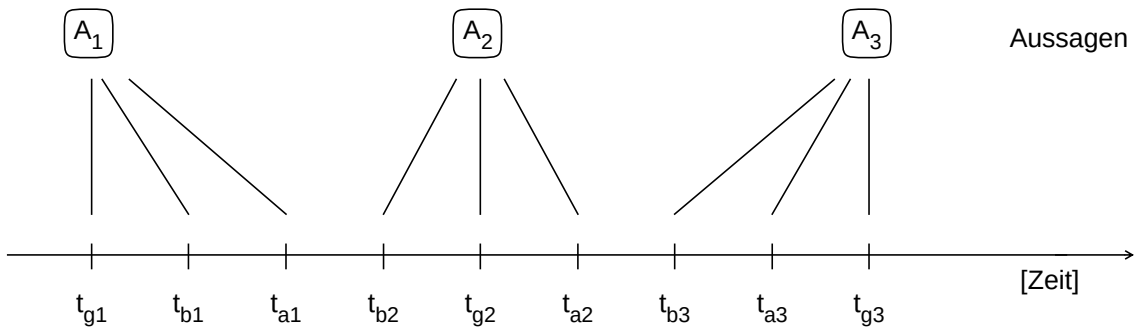
1.3 Wie wird Geschichte geschrieben ?

Die Geschichtsschreibung beginnt mit der Erzeugung einer Geschichtsausprägung, genauer: mit der Erzeugung der ersten Aussage zu einem Entity. Das intuitive Geschichtsverständnis legt es nahe, daß im weiteren Verlauf der Zeit neue Informationen zu dem Entity bekannt werden und auch als Aussagen an die aktuelle Geschichtsauprägung angefügt werden. Diese Idealvorstellung von Geschichte ist aber in praktischen Anwendungen nicht realisierbar, da nicht vorausgesetzt werden kann, daß Informationen zu einem Entity immer in der zeitlichen Reihenfolge ihrer "Gültigkeit" bekannt werden. Beispielsweise werden Lohnerhöhungen von Angestellten oftmals rückwirkend zum Jahresanfang gewährt. Dies Beispiel aus unserem Alltagsleben zeigt deutlich, daß die Beobachtung oder Feststellung eines Sachverhaltes und der Zeitpunkt, an dem der Sachverhalt zu wirken beginnt, nicht übereinstimmen müssen. Damit ist auch klar, daß der Zeitpunkt der Aufzeichnung eines Sachverhaltes a priori keine Korrelation mit ihrer Gültigkeitszeit aufweisen muß. Zusammenfassend ergeben sich drei wesentliche Zeitpunkte der Geschichtsschreibung:

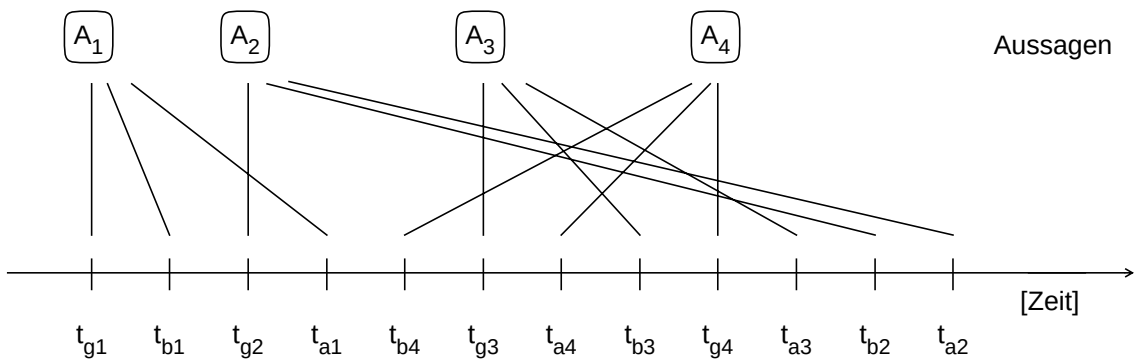
(1) **Gültigkeitszeit** t_g ist der Zeitpunkt, an dem ein Sachverhalt zu wirken beginnt, bzw. ab dem eine Aussage "wahr" wird oder an dem ein Meßwert ermittelt wird oder ...

(2) **Bestimmungszeit** t_b ist der Zeitpunkt, an dem ein Sachverhalt festgestellt oder beobachtet wird, bzw. an dem festgestellt wird, daß eine Aussage gilt oder gelten wird oder an dem ein Meßfühler den Meßwert weitermeldet oder ...

(3) **Aufzeichnungszeit** t_a ist der Zeitpunkt, an dem die Aussage aufgezeichnet wird.



a) prinzipielles Zusammenwirken von t_a, t_b und t_g



b) dynamische Zusammenwirken von t_a, t_b und t_g bei der Geschichtsentstehung

Bild 6: Zusammenwirken von t_a, t_b und t_g

Es ist klar, daß diese Zeitpunkte nicht vollkommen unabhängig voneinander sind. So muß immer $t_b \leq t_a$ gelten (eine Aussage kann erst aufgezeichnet werden, nachdem sie beobachtet wurde). t_g ist von den beiden anderen Zeiten unabhängig, da zukünftige oder rückwirkende Aussagen möglich sind. Weitergehende Zusammenhänge sind (für die allgemeine Geschichtsbetrachtung) nicht gegeben. Bild 6 /Hä84/ zeigt exemplarisch das Zusammenwirken von t_a, t_b und t_g.

Weiterhin ergibt sich, daß neben dem Anfügen an eine Geschichtsausprägung auch das Einfügen ermöglicht werden muß, da die Aussagen nach ihrer Gültigkeitszeit geordnet sind, aber die Aussagen nicht in dieser Reihenfolge aufgezeichnet werden müssen. Desweiteren muß auch ein Korrekturoperator geschaffen werden, da bei der Übernahme von Informationen aus der Miniwelt in Aussagen Fehler auftreten können.

Insgesamt ergeben sich damit vier grundlegende Operationen in Bezug auf Geschichtsausprägungen:

- das Erzeugen von Geschichtsausprägungen
- das Anfügen von Aussagen an Geschichtsausprägungen
- das Einfügen von Aussagen in Geschichtsausprägungen und

- das Korrigieren von Aussagen in Geschichtsausprägungen.

Durch die Hinzunahme von t_b und/oder t_a eröffnet sich ein neues Spektrum von **Korrektur**möglichkeiten. Analog der Bildung des Geschichtsbegriffs als zeitlich geordnete Folge von Aussagen, kann der Begriff der **Korrekturgeschichte** als zeitlich geordnete Folge von Korrekturständen gebildet werden. Da sich alle Korrekturstände zu einer Aussage auf die gleiche Gültigkeitszeit beziehen, muß eine weitere "Zeit" zur Bildung der Geschichte herangezogen werden. Hierzu bietet sich die Aufzeichnungszeit, also der Zeitpunkt an dem der Korrekturstand in die Korrekturgeschichte eingetragen wurde, an. Sei A eine Aussage, die zur Zeit t_b beobachtet, zur Zeit t_a aufgezeichnet und zur Zeit t_g gültig ist. Wird zur Zeit t_b' festgestellt, daß die Aussage A falsch ist und durch die Aussage A' ersetzt werden muß, so kann eine neue Aussage in die DB eingebracht werden, die den neuen Erkenntnisstand widerspiegelt. Für die neue Aussage A', der hier als Korrekturstand bezeichnet wird, gilt die gleiche Gültigkeitszeit (t_g'=t_g $\rightarrow$ Korrektur). Die Beobachtungszeiten t_b' und t_b sind unabhängig voneinander, da die Beobachtungen zu verschiedenen Zeiten von unterschiedlichen Beobachtern gemacht werden können. Es kann also nicht vorausgesetzt werden, daß die neueste Beobachtung auch die "richtigere" Beobachtung ist. Die Aufzeichnungszeit t_a' muß nach t_a liegen. Durch die Aufzeichnungszeiten t_a und t_a' wird eine Korrekturgeschichte gebildet, die quasi orthogonal zu der "normalen" Geschichte liegt. Bei der Auswertung entlang t_a für eine bestimmte Aussage werden die verschiedenen Korrekturstände der Aussage gezeigt.

Bei der Ergänzung wird eine neue Aussage A' zwischen zwei bestehende Aussagen (A'', A) eingefügt ($t_g \neq t_g$' , t_g'' $\neq t_g$') . Dabei muß, analog der obigen Überlegungen, t_a' nach t_a und t_a'' liegen, da die beiden umgebenden Aussagen A und A'' schon aufgezeichnet sein müssen, wenn die Aussage A' ergänzt werden soll.

Kurz zusammengefaßt ergeben sich folgende Beziehungen:

Korrektur: A' $\neq$ A und t_g' = t_g, sowie t_a' > t_a

Ergänzung: A $\neq$ A' $\neq$ A'' und t_g < t_g' < t_g'', sowie t_a' > t_a , t_a' > t_a''

Bild 7 zeigt Korrektur und Ergänzung bei zustandserhaltender Geschichte.

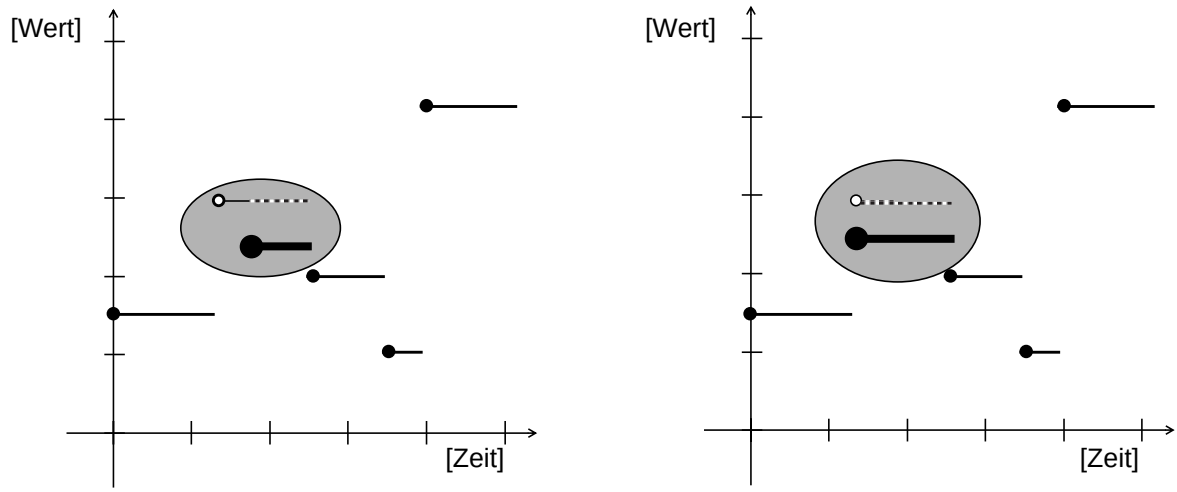


Bild 7: Ergänzung und Korrektur bei zustandserhaltender Geschichte

Insgesamt gesehen gibt es eine Integritätsbedingung, die für alle Anwendungen gültig ist. Es darf zu einem Objekt zu einer Zeit (d.h. in der Gesamtheit von t_g, t_b und t_a) jeweils nur eine Aussage existieren. Die Formulierung weiterer Integritätsbedingungen, auch beschränkt auf die verschiedenen Geschichtsarten, ist nicht möglich, ohne die Menge der Anwendungen einzuschränken.

2. Integration des Geschichtsmodells in das Relationenmodell

2.1 Welche Objekte und Operationen müssen integriert werden ?

Bevor die Integration des Geschichtsmodells in das Relationenmodell vorgestellt wird, soll die übliche Abbildung des E/R-Modells in das Relationenmodell skizziert werden. Entity-Typen und Relationship-Typen werden durch Relationen, Entity- und Relationship-Ausprägungen werden durch Tupel repräsentiert. Analog der in Kapitel 1 geschilderten Erweiterung des E/R-Modells wird auch das Relationenmodell erweitert: die Darstellungsmächtigkeit der Tupel muß erhöht werden. Ein Tupel des Relationenmodells kann nur den Informationsgehalt einer Aussage einer Geschichtsausprägung aufnehmen. Im erweiterten Relationenmodell soll ein Tupel hingegen den vollen Informationsgehalt einer Geschichtsausprägung umfassen (und damit natürlich auch Attribute zur Speicherung der Gültigkeitszeit, der Aufzeichnungszeit usw.). Um der Begriffsvielfalt Einhalt zu gebieten, wird im folgenden ein solchermaßen erweitertes Tupel ebenfalls als Geschichtsausprägung bezeichnet. Bild 8 zeigt, daß der Aspekt der Zeit nur direkt über die Geschichtsausprägung in das erweiterte Relationenmodell einfließt. Da aber jede Änderung einer Geschichtsausprägung bzw. das Erzeugen oder Löschen einer Geschichtsausprägung, durch die Belegung von Attributen mit Werten realisi-

ert wird und damit natürlich auch die Relationen geändert werden, erhalten diese Objekte indirekt einen Zeitbezug.

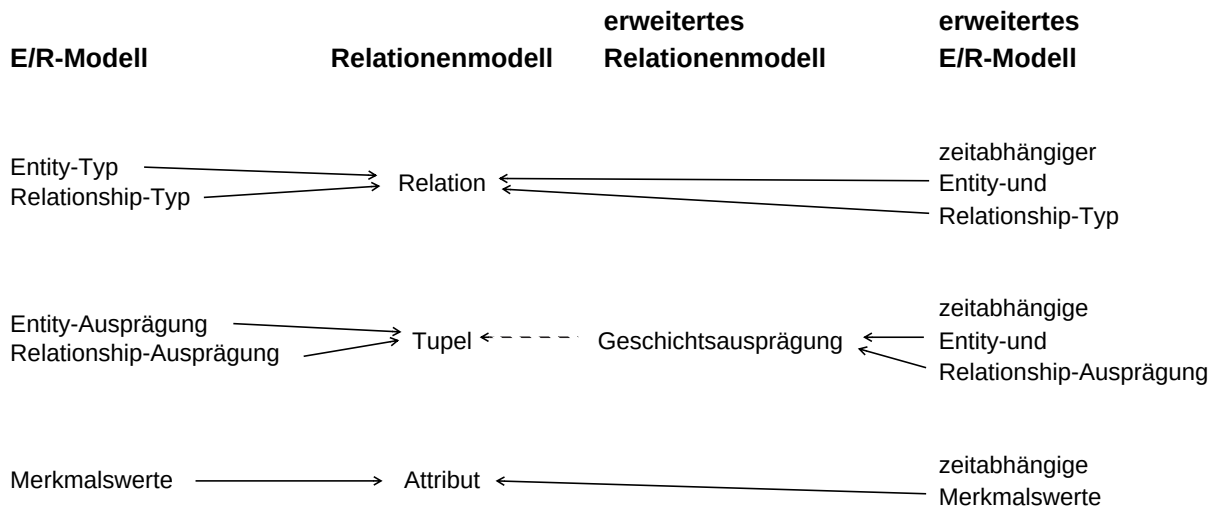


Bild 8: Abbildung des (erweiterten) E/R-Modells in das (erweiterte) Relationenmodell

Die Operationen des Relationenmodells können in natürlicher Weise auf Geschichtsausprägungen umgestellt werden. Die klassischen Mengenoperationen (Vereinigung, Schnitt, Differenz, symmetrische Differenz) erfahren keinerlei Veränderungen, da sie unabhängig vom Aufbau der "Tupel" sind. Bei den speziellen Relationenoperationen (Projektion, Division, Verbund, Restriktion) kann die Division analog den Mengenoperationen behandelt werden. Auch die Projektion bereitet keine Probleme. Die Selektion (Restriktion) wird in Abschnitt 2.2, der Verbund in Abschnitt 2.3 behandelt.

Zum besseren Verständnis sollen die Operationen im nachfolgenden kurz aus einer durch SQL /AC75/ gekennzeichneten Sichtweise betrachtet werden. Ausgehend von den SQL-Operationen zur Verwaltung "zeitloser" Tupel (Erzeugen, Ändern, Löschen, Darstellen) müssen diese Operationen für Geschichtsausprägungen adaptiert werden. Die Erzeugung einer Geschichtsausprägung entspricht der Erzeugung der ersten Aussage dieser Geschichtsausprägung und ist somit weitgehend identisch mit der Erzeugung eines herkömmlichen Tupels. Das Löschen einer Geschichtsausprägung ist nur durch die physische Beschränktheit des Adreßraumes zu rechtfertigen, da diese Operation der Intension des Geschichtsmodells "die Geschichte der Entities aufzuzeichnen" widerspricht. Der "Tod" eines Entities wird durch eine entsprechende Markierung der Geschichtsausprägung symbolisiert. Das Ändern einer Geschichtsausprägung muß, wie in Abschnitt 1.3 schon motiviert wurde, differenzierter betrachtet werden. Eine Änderung einer Geschichtsausprägung bedeutet entweder das An- oder Einfügen oder die Änderung einer Aussage innerhalb einer Geschichtsausprägung.

Bei der Darstellung von Geschichtsausprägungen, die abhängig von der zu repräsentierenden Geschichtsart ist, kann die Technik des "procedural attachment", die aus dem Bereich der Wissensrepräsentation bekannt ist /Ni82/, eingesetzt werden. Das Verfahren basiert auf der Idee, daß anstatt der eigentlichen Fakten eine Methode abgespeichert werden kann, mit deren Hilfe die benötigten Fakten deduziert werden können. Der

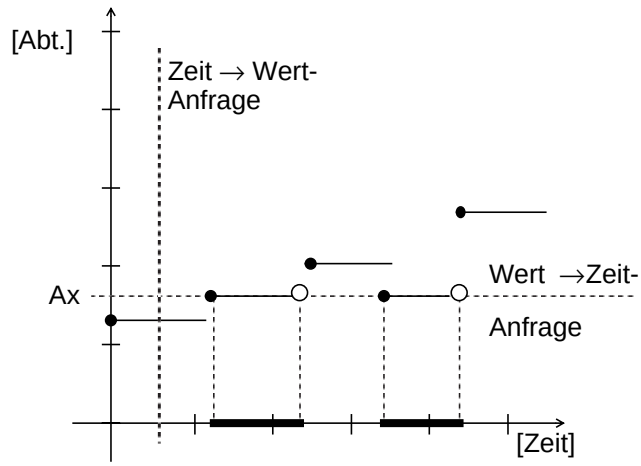


Bild 9: abhängige Zeitintervalle

Vorteil dieser Technik zeigt sich deutlich bei den Geschichtsausprägungen: anstatt alle möglichen Aussagen einer Geschichtsausprägung aufzuzeichnen, werden nur die charakteristischen Aussagen und eine Auswertefunktion gespeichert, die automatisch bei Bedarf eine Aussage zu dem gewünschten Zeitpunkt errechnet.

2.2 Welche Anfragearten sind zu unterscheiden ?

Zur Untergliederung zeitbehafteter Anfragen bieten sich zwei Kriterien an: zum einen die Art des Zeitbezugs und zum anderen die Fragerichtung.

Der Zeitbezug kann punktuell oder intervallartig sein. Die Fragerichtung gibt an, ob Aussagen zu einem bestimmten Zeitbezug gewünscht werden, oder ob zu Aussagen der Zeitbezug bestimmt werden soll. Damit ergibt sich der in Bild 10 gezeigte Klassifikationsbaum. Durch die Punktanfragen wird ein Schnappschuß der

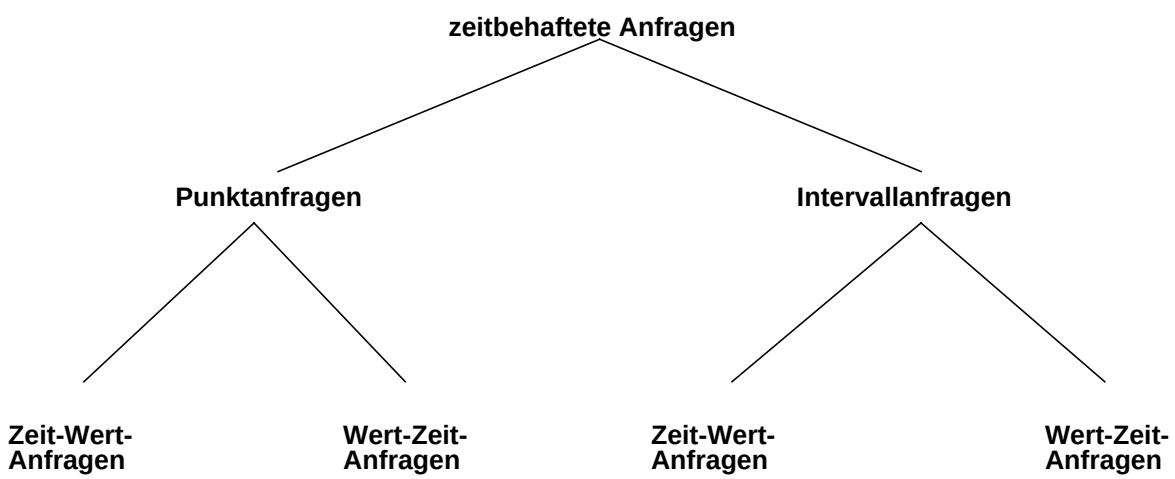


Bild 10: Klassifikation zeitbehafteter Anfragen

Miniwelt gebildet (Zeit →Wert) oder es wird gefragt, wann ein gegebener Schnappschuß aufgenommen wurde (Wert → Zeit). Das Ergebnis einer Zeit-Wert-Punktanfrage, die oft auch als "AS_OF-Query" bezeichnet wird, besteht somit aus einer Menge von unabhängigen Aussagen, d.h. die Aussagen gehören unterschiedlichen Geschichtsausprägungen an. Dies folgt aus der Injektivität des Geschichtsbildes: es kann zu einem Zeitpunkt zu einem Entity nur eine korrekte Aussage geben. Im umgekehrten Fall der Wert-Zeit-Punktanfragen, entsteht eine Menge von **abhängigen Zeitpunkten**. Abhängig deshalb, weil mehrere Zeitpunkte

aus einer Geschichtsausprägung stammen können (die gleiche Aussage kann zu mehreren Zeitpunkten gültig sein). Bei den Intervallanfragen wird der Zeitbezug durch Zeitintervalle beschrieben. Auch hier kann zwischen Zeit-Wert- und Wert-Zeit-Anfragen unterschieden werden. Die Zeit-Wert-Intervallanfragen werden auch als "WALK_THRU_TIME-Queries" bezeichnet. Analog zu den abhängigen Zeitpunkten entstehen **abhängige Zeitintervalle**. Die Unterscheidung zwischen abhängigen und unabhängigen Zeitpunkten bzw. Zeitintervallen wird bei der Verwendung von sogenannten "Built-in"-Funktionen deutlich. Die Anfrage "Wieviele Angestellte, die zur Zeit noch angestellt sind, haben jemals in ihrer Laufbahn in Abteilung Ax gearbeitet ?", zeigt dies deutlich. Ohne Beachtung der abhängigen Zeitintervalle würden Duplikate gezählt werden, da ein Angestellter mehrmals in dieser Abteilung gearbeitet haben kann. Bild 9 stellt diesen Sachverhalt graphisch dar. Auch bei der Bearbeitung der Verbund-Operation müssen diese Abhängigkeiten beachtet werden. Dazu mehr im nächsten Abschnitt.

2.3 Was ist ein "zeitbehafteter Verbund" ?

Dieser Abschnitt beschäftigt sich mit den Problemen bei der Bearbeitung eines Verbundes zwischen zeitbehafteten Entities. Der Sonderfall, daß ein beteiligter Entity-Typ nicht zeitbehaftet ist, wird auf den allgemeinen Fall zurückgeführt, indem angenommen wird, daß alle Ausprägungen dieses Typs seit "Urbeginn" gültig sind. Bei einem **zeitbehafteten Verbund** (also einem Verbund, an dem wenigstens ein zeitbehafteter Entity-Typ beteiligt ist) müssen zwei Fälle unterschieden werden:

(1) Das Verbundattribut ist die Gültigkeitszeit, z.B.:
Welcher Angestellte wohnte in Karlsruhe, als das Etat des Projektes P1 50.000 DM betrug ?
Bemerkung: Die Geschichte der Angestellten ist zunächst unabhängig von der Geschichte der Projekte, da nicht gefordert ist (im Gegensatz zu Bild 2), daß die Angestellten bei den entsprechenden Projekten mitgearbeitet haben. Die Verbindung zwischen beiden Geschichten wird alleine durch die gemeinsame Zeit, in der sich beide Geschichten entwickelt haben, gebildet.

(2) Das Verbundattribut ist ein beliebiges anderes Attribut, z.B.:
In welchen Abteilungen war Fritz bisher tätig ?
Bemerkung: Natürlich muß auch bei dieser Art des Verbundes der Zeitbezug beachtet werden, da ja zwei Geschichten verbunden werden. Die Verbindung zwischen den Geschichten wird aber im Gegensatz zum ersten Fall nicht über die Zeit, sondern über andere Attribute gebildet.

Die Vorgehensweise im ersten Fall ist offensichtlich: Zunächst wird die Gültigkeitszeit der zweiten Geschichte bestimmt und anschließend überprüft, ob in dieser Zeit die Auswahlbedingung (also ob der Angestellte in Karlsruhe gewohnt hat) erfüllt ist. Im zweiten Fall kann von folgendem Denkmodell ausgegangen werden:

Die Verbundelemente sind Geschichtsausprägungen, d.h. es werden die Geschichten der einzelnen Objekte miteinander verbunden (Zur Verdeutlichungist in Bild 11 das Attribut VALID zweifach dargestellt, obwohl es eigentlich nur einmal in den neu gebildeten Gschichtsausprägungen vorhanden ist). Innerhalb einer Geschichtsausprägung kann sich der Wert des Verbundattributes mehrfach ändern. Dies hat zur Folge, daß zwei Geschichtsausprägungen "zeitweise" zueinander passen, "zeitweise" nicht. Damit entstehen aus zwei Geschichtsausprägungen, die in ihrer Entwicklung ein oder mehrmals zueinander passen, entsprechend

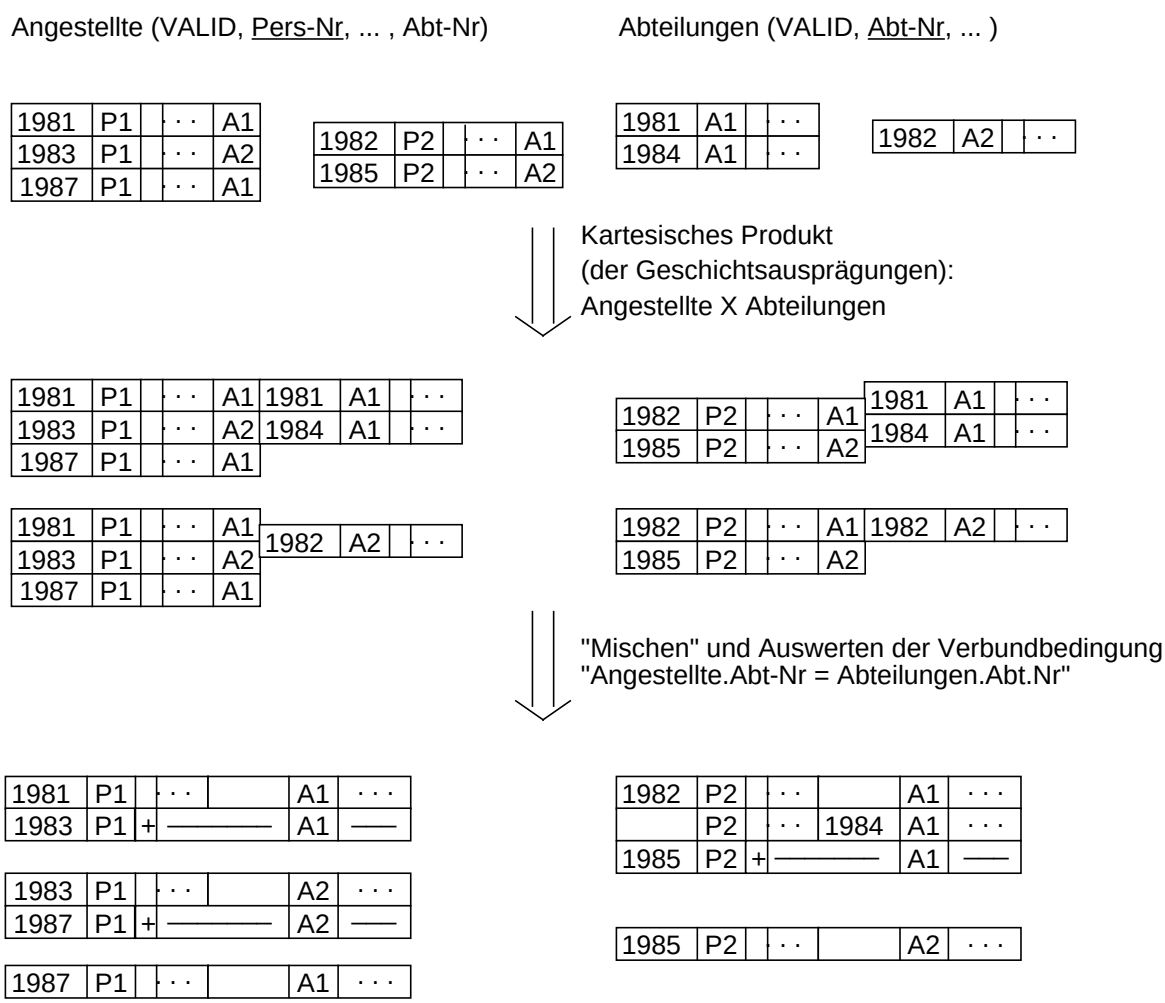


Bild 11: Beispiel eines zeitbehafteten Verbundes

eine oder mehrere Geschichtsausprägungen des neuen Typs. Die neuen Geschichtsausprägungen enthalten alle Aussagen der Ausgangsgeschichten, in denen die "Geschichten zueinander paßten".

Bild 11 zeigt einen zeitbehafteten Verbund zwischen zwei Angestellten (P1 und P2) und zwei Abteilungen (A1 und A2). In einem ersten Schritt wird das kartesische Produkt der beiden Relationen gebildet. Bei einem nicht-zeitbehafteten Verbund werden jetzt alle Tupel eliminiert, die die Verbundbedingung nicht erfüllen. Im vorliegenden Fall müssen dazu erst gemeinsame Zeitintervalle gebildet werden, in denen jetzt jeweils einzeln die Gültigkeit der Verbundbedingung überprüft werden muß. Ist der Test erfolgreich, wird das Zeitintervall zu einer Aussage reduziert und in die Ergebnis-Geschichtsausprägung übernommen. Ist im nächsten gemeinsamen Zeitintervall die Verbundbedingung ebenfalls erfüllt, wird dieses Zeitintervall ebenfalls als Aussage in die Ergebnis-Geschichtsausprägung übernommen. Ansonsten muß ein Platzhalter, der den Tod des Objekts symbolisiert, an die Ergebnis-Geschichtsausprägung angefügt werden.

Diese Verarbeitungsstrategie kann dahingehend optimiert werden, daß das kartesische Produkt nicht explizit berechnet werden muß. Die Bildung der gemeinsamen Zeitintervalle bei der Bearbeitung des Verbundes bereitet bei der ereignisorientierten und der zustandserhaltenden Geschichte keine Probleme. Bei der

zustandsverändernden und ableitbaren Geschichte sind hierzu u. U. sehr komplexe mathematische Funktionen notwendig. Diese Funktionen, die von der zugrundegelegten Anwendungssemantik abhängen, müssen vom Benutzer dem DBS zur Verfügung gestellt werden. Dabei ist es z.B. denkbar, daß die kontinuierlichen Funktionen in Treppenfunktionen mit äquidistanter Schrittweite zerlegt werden.

3. Ein SQL-basierter Sprachvorschlag für temporale DBS

Bisher wurden die logischen Aspekte von Zeitverwaltungen untersucht. Dabei wurde gezeigt, wie durch das Konzept der Geschichtsausprägungen eine konsistente Erweiterung des E/R-Modells auf das Relationenmodell fortgesetzt werden konnte. Im folgenden soll die Realisierung von Zeitverwaltungen untersucht werden. Dazu werden zunächst einige existierende Systeme und die dort verwendeten Konzepte mit dem vorliegenden Vorschlag in Beziehung gesetzt, bevor die Realisierung desselben vorgestellt wird.

3.1 Abgrenzung zu verwandten Arbeiten

Bei den verwandten Arbeiten sind zwei Gruppen zu unterscheiden, von denen jeweils zwei Vertreter kurz vorgestellt und diskutiert werden. Die erste Gruppe wird durch die NDBS-Prototypen AIM (Advanced Information Management System) und DASDBS (Darmstädter DBS) gebildet. Bei der zweiten Gruppe handelt es sich um Systeme, die als Erweiterung des Relationenmodells konzipiert sind. Hier werden TODM (Temporally Oriented Data Model) und TQuel (Temporal Query Language) besprochen.

Die Grundlage des AIM-Prototyps /Da84, Lu85/ bildet das NF^2-Datenmodell (Non-First-Normal-Form) / PA86, Sch85/. In die Anfragesprache, die SQL-ähnlich ist, wurden Sprachelemente zur Formulierung zeitbehafteter Anfragen integriert. Das Zeitmodell basiert auf der Aufzeichnungszeit, d.h., bei jeder Änderungsoperation wird eine neue Version eines NF^2-Tupels erzeugt und mit einem Zeitstempel versehen. Zur Speicherplatzersparnis werden einfache Rückwärtsdifferenzen /Da84/ eingesetzt. Durch die ausschließliche Verwendung der Aufzeichnungszeit werden wesentlich weniger Funktionen zur Wartung der Strukturen benötigt, da nur das Anfügen, nicht aber das Einfügen oder die Korrektur benötigt werden. Die Sprachmittel erlauben es, daß auf diesen Ketten einfache AS_OF- oder WALK_THRU_TIME-Anfragen formuliert werden können. Die Verwendbarkeit des Modells leidet allerdings stark unter der Tatsache, daß die Gültigkeitszeit nicht unterstützt wird: alle zeitbezogenen Anfragen benutzen die Aufzeichnungszeit, die prinzipiell unabhängig zur Gültigkeitszeit ist (beispielsweise bei rückwirkenden Gehaltserhöhungen).

DASDBS /De85/ basiert ebenfalls auf dem NF^2-Datenmodell. Im Gegensatz zum AIM-Prototyp soll DASDBS aber nur den neutralen Kern eines sonst anwendungsorientierten DBS realisieren. Aus diesem Grunde wird auch nur ein Versionen-Manager zur Verfügung gestellt, dessen Funktionen nicht in der Anfragesprache repräsentiert sind. Ebenso wie beim AIM-Prototyp werden als Grundlage die Aufzeichnungszeit verwendet und die einzelnen Versionen als Differenzen abgespeichert. Zusätzlich kann zur Leistungssteigerung die Kettenlänge der Rückwärtsdifferenzen durch die Abspeicherung kompletter Versionen begrenzt werden. Aber auch hier ist vorgesehen, daß "neue" Versionen immer am Anfang der Kette angefügt werden, was die

Verwendbarkeit des Versionen-Managers wiederum stark eingeschränkt. Aufbauend auf diesem Versionenverwalter sollen anwendungsorientierte Zeitverwaltungen implementiert werden.

Die beiden Prototyp-Implementierungen zeigen, daß sehr viel Sorgfalt auf die Aspekte der Speicherungsstrukturen gelegt wurde, daß aber nur wenige Aspekte eines Zeitmodells an der Benutzerschnittstelle unterstützt werden. Im Gegensatz dazu sind die Prototypen der folgenden Gruppe zu sehen. Hier werden jeweils die Gültigkeitszeit, die Aufzeichnungszeit und eventuell weitere "benutzerdefinierte Zeiten" unterstützt. Die Aspekte der Abspeicherung werden allerdings nicht untersucht.

TODM bzw. TOSQL (Temporally Oriented SQL) /Ar86a, Ar86b/ stellen eine Erweiterung der Sprache SQL dar. Die SELECT-Anweisung wurde um einige Prädikate erweitert, die es erlauben, eine Anfrage mit einem Zeitbezug zu versehen, der alle vier in Kapitel 2.2 vorgestellten Möglichkeiten umfaßt. Es werden die Gültigkeits- und die Aufzeichnungszeit, jeweils für Tupel, verwaltet. Die Ausdrucksmächtigkeit wird allerdings zum einen dadurch eingeschränkt, daß es nicht möglich ist, die Prädikate zur Beschreibung des Zeitbezugs ineinander zu verschachteln und zum anderen durch das Fehlen von Funktionen zur Bestimmung impliziter Zeitpunkte (etwa TIME_OF (event)). Weiterhin wird durch das Fehlen einer Verbund-Operation die Verwendbarkeit des Modells stark eingeschränkt.

TQuel /SA86, Sn86a, Sn86b, Sn87/ stellt eine Erweiterung der Sprache Quel dar. Das Datenmodell wurde wiederum um zwei Attributtypen erweitert, die die Gültigkeitszeit und die Aufzeichnungszeit repräsentieren. Attribute dieser Typen werden implizit für jede zeitbehaftete Relation vereinbart. Die Gültigkeitszeit einer Aussage wird entweder als Intervall (FROM, TO) bei der Intervallgeschichte (in unserer Terminologie: zustandserhaltende Geschichte) oder als Zeitpunkt (AT) bei der ereignisorientierten Geschichte ausgedrückt. Durch die explizite Verwendung der Intervalle pro Tupel ergeben sich schwerwiegende Probleme, wenn Änderungen von mehreren Attributen eines Tupels mit unterschiedlichen Gültigkeitsintervallen auftreten.

Die beiden letztgenannten Systeme sind als Zusatzebene auf relationen DBS implementiert. Bei der Abbildung der Geschichtsausprägungen auf das unterliegende Relationenmodell wird in beiden Fällen der gleiche Weg beschritten: Die Gültigkeitszeit wird zum Primärschlüssel der "zeitlosen" Relation hinzugenommen. Dieses Vorgehen hat einige gravierende Nachteile. Aus der logischen Sicht wird eine Geschichtsausprägung, die eine einzelne Entity-Ausprägung beschreibt, in mehrere Tupel zerteilt, die voneinander unabhängig in der Relation abgespeichert werden. Das Wissen, daß diese Tupel ein Entity beschreiben muß nun bei jeder Anfrage in die Anfrage integriert werden, da es auf der Datenstrukturebene nicht mehr vorhanden ist. Diese Tatsache macht die zeitbehafteten Anfragen auf dem zugrundeliegenden relationalen DBS komplex und schwerfällig. Betrachtet man den Suchraum einer Anfrage, die sich nur auf die aktuellen Daten bezieht, so müssen im vorliegenden Fall trotzdem die historischen Daten durchsucht werden, da sie alle in einer Relation abgespeichert sind. Auch ein Zugriffspfad kann nicht effektiv eingesetzt werden, da er auch automatisch die historische Information mit einbezieht. Setzt man die (realistische) Annahme voraus, daß der Anteil der historischen Daten wesentlich höher ist als der Anteil der aktuellen Daten, so wird der Zugriff auf die aktuellen Daten deutlich behindert. Zusätzlich ist ein immenses Datenvolumen zu erwarten, da die Verwendung von Differenztechniken wesentlich erschwert oder sogar unmöglich wird, wenn man unterstellt, daß das relationale DBS einen einheitlichen Aufbau der Tupel einer Relation erwartet.

Der vorliegende Vorschlag versucht einige Unzulänglichkeiten der vorgestellten Ansätze durch die Verwendung der Gültigkeits- und der Aufzeichnungszeit sowie durch das Konzept der Geschichtsausprägungen zu vermeiden. Dabei wird die den Benutzer primär interessierende Gültigkeitszeit in den Mittelpunkt gestellt. Die

damit verbundene erhöhte Komplexität der Operationen zur Wartung der Geschichtsausprägungen wird durch die wesentlich höhere Flexibilität des Zeitmodells mehr als ausgeglichen. Anfragen, die sich auf die Aufzeichnungszeit beziehen, wie sie üblicherweise für Revisionsaufgaben benötigt werden, erfordern einen Mehraufwand, da die Aussagen innerhalb der Geschichtsausprägungen umgeordnet werden müssen. In diesen Sonderfällen erscheint dies aber tolerierbar.

Das Konzept der Geschichtsausprägungen erlaubt die natürliche Abbildung der realen Welt in die Miniwelt des DBS, da analog dem "zeitlosen" Ansatz eine 1:1-Beziehungen zwischen den Objekten gewahrt bleibt: jedes Entity der realen Welt wird durch genau eine Geschichtsausprägung in der Miniwelt repräsentiert. Damit prädestiniert sich die Geschichtsausprägung auch als Einheit der Verarbeitung, was letztendlich eine einfache und natürliche Formulierung von zeitbehafteten Anfragen ermöglicht. Dies zeigt sich insbesondere im Falle des zeitbehafteten Verbundes, dessen Semantik in einfacher Weise durch die Geschichtsausprägungen definiert werden kann. Der in Abschnitt 3.4 vorgestellte Sprachvorschlag demonstriert diesen Sachverhalt.

Die Verwendung der Geschichtsausprägungen als Einheit der Verarbeitung bietet neben der Simplifizierung der Anfragen, da der Aufbau der Geschichtsausprägungen dem DBS bekannt ist und somit nicht mehr in die Anfrage integriert werden muß, auch die Möglichkeit zu mannigfaltigen Optimierungen innerhalb des DBS. So können beispielsweise aktuelle von historischen Daten getrennt werden. Dies erlaubt den Einsatz von getrennten Zugriffspfaden oder auch den Einsatz spezieller Speicherungsstrukturen. Auch der Einsatz von Differenztechniken wird möglich, die dem zu erwartenden großen Datenvolumen entgegenwirken.

3.2 Wie können die Geschichtsarten repräsentiert werden ?

Der Sprachvorschlag basiert auf einer Erweiterung von SQL, die im wesentlichen auf die Erweiterung des in Kapitel 2 vorgestellten erweiterten Relationenmodell zurückzuführen ist: als "Einheit der Verarbeitung" muß das "Tupel" durch die Geschichtsausprägung ersetzt werden. Jede Geschichtsausprägung besteht aus einer zeitlich geordneten Folge von Aussagen. Jede Aussage entspricht einem herkömmlichen Tupel, erweitert um ein Attribut VALID zur Darstellung der Gültigkeitszeit, ein Attribut ALIVE zur Anzeige, ob der Inhalt noch gültig ist, und einem Attribut TS zur Speicherung der Aufzeichnungszeit. Auf den Geschichtsausprägungen sind folgende Operationen definiert:

- CREATE zur Erzeugung einer Geschichtsausprägung (Abspeichern der ersten Aussage der Geschichtsausprägung).
- UPDATE zum Ändern von Attributwerten von Aussagen in Geschichtsausprägungen. Diese Operation kann beispielsweise zur Korrektur falscher Aussagen verwendet werden.
- APPEND zum Anfügen von Aussagen an Geschichtsausprägungen. Das "normale" Fortschreiben der Geschichte.
- INSERT zum Einfügen von Aussagen zwischen bestehende Aussagen von Geschichtsausprägungen. Damit kann der Ergänzungsoperator aus Abschnitt 1.3 realisiert werden.
- DISCARD zum Löschen von Aussagen oder ganzen Geschichtsausprägungen. Dieser Operator ist auf die physische Begrenztheit des Adreßraumes zurückzuführen.
- SELECT zum selektieren von Aussagen und Geschichtsausprägungen. Dieser Operator wird im folgenden Abschnitt erläutert.

3.3 Wie können die Anfragearten unterstützt werden ?

Zur Unterstützung der vier Anfragearten werden vier Prädikate (AT, WHEN, DURING, WHILE) eingeführt. Durch diese Konstrukte wird der Zeitbezug einer Anfrage, genauer: der Zeitbezug einer an diese Konstrukte gebundenen (zeitlosen) booleschen Bedingung ausgedrückt. AT und DURING beschreiben dabei den Zeitbezug direkt mittels Zeitkonstanten, während WHEN und WHILE den Zeitbezug indirekt durch die Gültigkeit einer weiteren (zeitlosen) booleschen Bedingung beschreiben. Im folgenden soll nun die Semantik der Prädikate schrittweise erläutert und verfeinert werden. Zur Veranschaulichung werden einfache Anfragen in SQL formuliert. Grundlage bildet das in Kapitel 3.5 definierte Schema. Die Anfragen sind aber auch mit den Angaben aus Bild 1 und 2 verständlich. Zunächst wird zur einfacheren Formulierung der Prädikate eine spezielle Notation eingeführt:

- t_1, t_2, t_3, ... seien Zeitpunkte (einer bestimmten Granularität);
- I_1, I_2, I_3, ... seien Intervalle der Form $[I_{1b},I_{1e}]$, $[I_{2b},I_{2e}]$, $[I_{3b},I_{3e}]$, ... ;
- *I* sei eine Menge von Intervallen, *T* eine Menge von Zeitpunkten;
- B_i seien boolesche Bedingungen ohne Zeitbezug.

Damit lassen sich folgende Typen von zeitbehafteten Ausdrücken formulieren:

- **(B) AT t** : liefert alle Aussagen, für die B zum Zeitpunkt t zu TRUE evaluiert wird, z.B.:
  SELECT * FROM angestellte WHERE (name = 'Hans') AT 01.06.1984
  liefert alle Angestellten mit Namen Hans, die am 1. Juni 1984 eingestellt waren.
- **(B) SOMETIMES DURING I** liefert (Teile von) Geschichtsausprägungen, die im Intervall I mindestens zu einem Zeitpunkt die Bedingung B erfüllen, z.B.:
  SELECT * FROM angestellte WHERE (wohnort = 'Hamburg') SOMETIMES DURING [198*]
  liefert alle Angestellten, die in den achtziger Jahren irgendwann einmal in Hamburg gewohnt haben.
- **(B) ALWAYS DURING I** : analog zur obigen Formulierung mit dem Unterschied, daß im Intervall I "zu jedem Zeitpunkt" die Bedingung B erfüllt sein muß. Ersetzt man in der obigen Anfrage SOMETIMES durch ALWAYS, so erhält man alle Angestellten, die in den achtziger Jahren durchgehend in Hamburg gewohnt haben.

Die Prädikate beziehen sich zunächst auf einen Zeitpunkt bzw. auf ein Zeitintervall. Da aber die Zeitpunkte und Zeitintervalle auch implizit bestimmt werden können und dabei Mengen von Zeitpunkten bzw. Mengen von Zeitintervallen entstehen können, müssen die Definitionen erweitert werden:

- **(B) AT *T*** : liefert alle Aussagen, für die (B) AT t für ein $t \in T$ gilt.
- **(B) SOMETIMES / ALWAYS DURING *I*** : liefert alle Aussagen, für die (B) SOMETIMES / ALWAYS DURING I für ein / für alle $I \in I$ gilt.

Bei den bisherigen Prädikaten wurden die Zeitpunkte bzw. die Zeitintervalle explizit angegeben. Um jedoch eine höhere Ausdrucksmächtigkeit zu erreichen, muß es möglich sein, diese auch implizit anzugeben bzw. durch andere Prädikate zu bestimmen. WHEN und WHILE erlauben eine implizite Festlegung von Zeitpunkten bzw. Zeitintervallen. Die Prädikate können folgendermaßen erläutert werden:

- **WHEN (B)** : liefert alle Zeitpunkte, an denen die boolesche Bedingung B erfüllt ist, z.B.:
  SELECT VALID FROM angestellte WHEN (wohnort = 'Kaiserslautern')
  liefert den Termin, an dem die Angestellten nach Kaiserslautern gezogen sind. Eine WHERE-Klausel kann entfallen, da keine weiteren Bedingungen an die Angestellten gestellt werden. Der Zusatz WHERE wohnort = 'Kaiserslautern' würde zusätzlich fordern, daß die Angestellten zur Zeit (AT NOW) noch in Kaiserslautern wohnen.
- **WHILE (B)** : liefert alle Zeitintervalle, in denen B erfüllt ist,z.B.:
  SELECT name, VALID FROM angestellte WHILE (gehalt > 1500)
  liefert alle Angestellten mit den Intervallen, in denen sie mehr als 1500 verdient haben.

Die Beziehungen zwischen WHEN und AT bzw. zwischen WHILE und DURING lassen sich folgendermaßen ausdrücken:

- WHEN (B) = $T \Leftrightarrow$ (B) AT T = TRUE
- WHILE (B) = $I \Leftrightarrow$ (B) ALWAYS DURING (I) = TRUE

Mit diesen Prädikaten (AT, DURING, WHEN, WHILE), die durchweg auf Mengen von Zeitpunkten bzw. auf Mengen von Zeitintervallen definiert sind, lassen sich mächtige Selektionsbedingungen formulieren. Zur Verknüpfung von Ergebnissen der einzelnen Prädikate werden noch einige Operationen benötigt, die natürlich ebenfalls auf Mengen definiert sein müssen. Zunächst soll jedoch zur Verdeutlichung der Semantik eine einfachere Form vorgestellt werden. OVERLAPS bildet das Schnittintervall zweier Intervalle, EXPAND bildet die Vereinigung zweier Intervalle, die dicht aneinanderliegen oder sich überschneiden, während UNION als Erweiterung von EXPAND gesehen werden kann, da das Ergebnis neben den durch EXPAND gebildeten Intervallen auch die Intervalle enthält, die an keiner EXPAND-Operation beteiligt waren (Die Operationen lassen sich auch auf Mengen von Zeitpunkten anwenden, was hier aber nicht explizit gezeigt werden soll). Weitere Operationen auf Intervallen, allerdings ohne direkten Bezug zum Aspekt der Zeit, finden sich in /LJ88/.

OVERLAPS (I_1,I_2) := $[MAX(I_{1b},I_{2b}), MIN(I_{1e},I_{2e})]$, falls $I_{2b} \le I_{1e}$ und $I_{1b} \le I_{2e}$
[0,0] sonst

EXPAND (I_1,I_2) := $[MIN(I_{1b},I_{2b}), MAX(I_{1e},I_{2e})]$, falls $I_{2b} \le I_{1e}$ und $I_{1b} \le I_{2e}$
[0,0] sonst

UNION (I_1,I_2) := EXPAND(I_1,I_2) falls $I_{2b} \le I_{1e}$ und $I_{1b} \le I_{2e}$
$\{I_1,I_2\}$ sonst

Die Erweiterung dieser Operatoren auf Mengen von Zeitpunkten bzw. Mengen von Intervallen ergibt:

OVERLAPS (I_1, I_2) := $\cup$ OVERLAPS (I_1, I_2) mit $I_1 \in I_1$, $I_2 \in I_2$

EXPAND (I_1, I_2) := $\cup$ EXPAND (I_1, I_2) mit $I_1, I_2 \in I_1 \cup I_2 \cup$ EXPAND (I_1, I_2),
solange bis EXPAND (I_1, I_2) konstant bleibt.

UNION (I_1, I_2) := EXPAND $(I_1, I_2) \cup \{I_i\} \cup \{I_j\}$, mit $I_i, I_j \in I_1 \cup I_2$ und für alle I_i, I_j gilt:
EXPAND $(I_i, I_j) = [0, 0]$

Bild 12 zeigt graphisch an einem Beispiel die Wirkungsweise der Operationen auf. Bisher wurde gezeigt, wie die Prädikate und Operatoren auf Mengen von Zeitpunkten bzw. Mengen von Zeitintervallen arbeiten. Jetzt soll gezeigt werden, wie zusammengesetzte boolesche Bedingungen aufgelöst werden können:

(B_1 and/or B_2) AT $T \Leftrightarrow$ (B_1) AT T and/or (B_2) AT T

(B_1 and/or B_2) SOMETIMES/ALWAYS DURING $I \Leftrightarrow$ (B_1) SOMETIMES/ALWAYS DURING I and/or (B_2) SOMETIMES/ALWAYS DURING I

WHEN (B_1 and/or B_2) $\Leftrightarrow$ OVERLAPS/UNION (WHEN (B_1), WHEN (B_2))

WHILE (B_1 and/or B_2) $\Leftrightarrow$ OVERLAPS/UNION (WHILE(B_1), WHILE(B_2))

Neben diesen Berechnungsformeln sind noch folgende Funktionen notwendig:

BEGIN_OF (*I*) = $\{I_b\}$ mit $I \in I$

END_OF (*I*) := $\{I_e\}$ mit $I \in I$

DURATION (*I*) := $\sum I_e - \sum I_b$ mit $I \in I$.

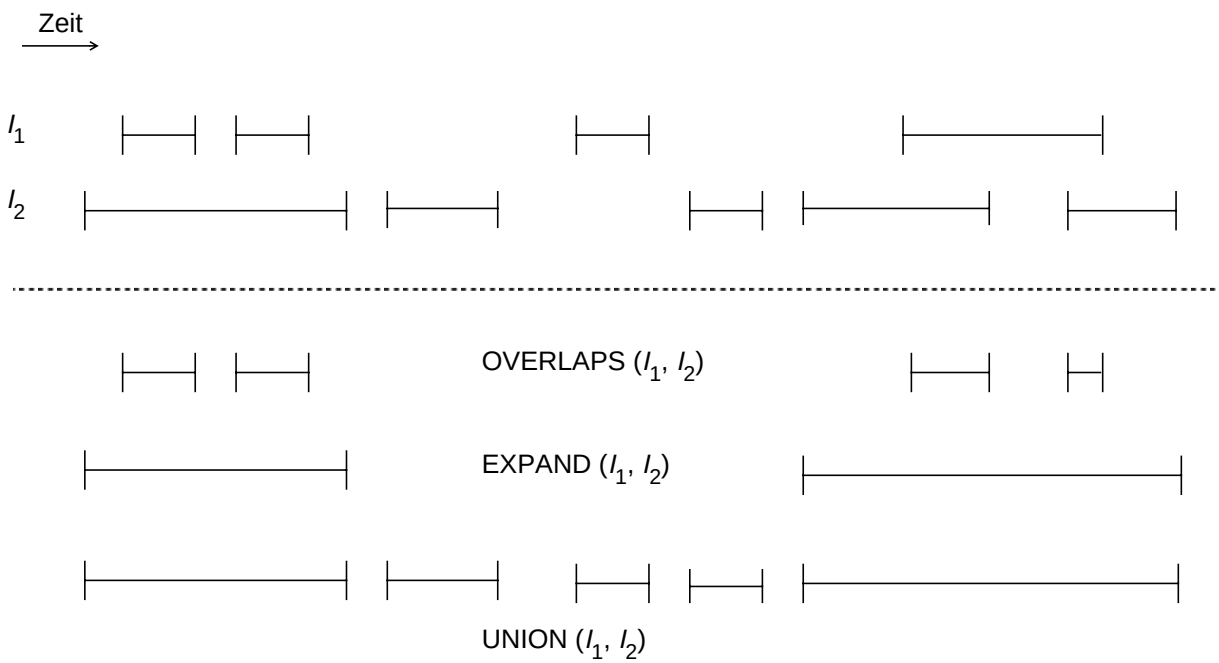


Bild 12: Beispiele für OVERLAPS, EXPAND und UNION

3.4 Der Sprachvorschlag

Nachdem die einzelnen Prädikate und Operatoren vorgestellt wurden, sollen die Zusammenhänge geschildert werden. Dazu wird eine ziemlich abstrakte Syntax der SELECT-Anweisung gegeben. Anschließend sind noch grobe Skizzen der Änderungsanweisungen und Anweisungen zur Integritätserhaltung aufgezeigt. (Worte in Blockbuchstaben stellen Schlüsselworte der Syntax dar, "/" trennt Alternativen, "[" und "]" schließen optionale Klauseln ein.)

<statement> ::= <query_statement> / <upd_statement> / <integrity_stat>

<query_statement> ::= SELECT <projection>
FROM <objects>
WHERE [<qualification>] [<time_spec>]

<projection> ::= * / <attr1> ... <attrn> / VALID / TS / ALIVE <time_selection>

<time_selection> ::= ALL / ONLY / DURING (<t>,<t>)

<objects> ::= <relation_spec> [, <relation_spec>]

<qualification> ::= <bool_expr> <time_expr>
[<operator> <bool_expr> <time_expr>]

<bool_expr> ::= vergleicht Attributwerte mit Literalen oder Attributwerten
(Verbund); boolesche Operatoren und Klammerung sind möglich.

<time_expr> ::= AT (<t>) / WHEN (<bool_expr>) /
<some_always> <interval_spec>

<some_always> ::= SOMETIMES / ALWAYS

<interval_spec> ::= DURING (<t>, <t>) / WHILE (<bool_expr>)

<operator> ::= AND / OR / OVERLAPS / EXPAND / UNION

<time_spec> ::= AS_OF <t>

<t> ::= Zeitpunkt, z.B.: 1987/12/22 oder 1987/12/22/08:47:25 oder auch:
NOW / BIRTH / DEATH

<upd_statement> ::= <create> / <append> / <insert> / <update> / <discard>

<create> ::= CREATE <VALID, data> FROM <objects>

<append> ::= APPEND <VALID, data> FROM <objects>

<insert> ::= INSERT <VALID, data> FROM <objects>

<update> ::= UPDATE <VALID, data> FROM <objects>

<discard> ::= DISCARD <objects> WHERE <qualification>

<integrity_stat> ::= ASSERT ON <objects> [BY <upd_statement>] [<action>] <qualification>

3.5 Ein Beispiel

Das Beispiel formalisiert die Miniwelt, die in Bild 2 dargestellt ist. Die Entity-Typen "angestellte" und "projekte" werden als Relationen gleichen Namens beschrieben. Der Relationsship-Typ wird als Relation "arbeitet_mit" dargestellt. Die Primärschlüssel werden durch Unterstreichung gekennzeichnet.

```
CREATE T_RELATION     angestellte
    valid             : TIME ( YEAR),
    alive             : BOOLEAN,
    name              : CHAR(30),
    wohnort           : CHAR(30),
    gehalt            : REAL;

USE  "zustandserhaltende Geschichte" FOR (wohnort, gehalt).

CREATE T_RELATION     projekte
    valid             : TIME ( YEAR),
    alive             : BOOLEAN,
    pro_nr            : CHAR(3),
    etat              : REAL,

USE  "zustandserhaltende Geschichte" FOR (etat).

CREATE T_RELATION     arbeitet_mit
    valid             : TIME ( YEAR),
    alive             : BOOLEAN,
    funktion          : CHAR(30),
    teilbereich       : CHAR(10),
    mitarbeiter       : CHAR(30) /* Angestellten-Name */,
    projekt_nr        : CHAR(3) /* Projekt-Bezeichnung */;

USE  "zustandserhaltende Geschichte" FOR (funktion, teilbereich, mitarbeiter, projekte).
```

Aufbauend auf diesem Schema und den schon in Bild 2a gezeigten Daten sollen einige Beispielanfragen erläutert werden. Zunächst sollen vier Fragen korrespondierend zu der Klassifikation aus Abschnitt 2.2 gezeigt werden:

(3) Ausgehen von einem bestimmten Zeitpunkt soll der Wert eines Attributes bestimmt werden:
Welches Gehalt bezog Hans 1981 ?

```
SELECT      gehalt
FROM        angestellte
WHERE       (name = "Hans") AT 1981
```

Ausgabe: 1000

Durch Modifikation der Projektionsliste zu

```
SELECT      VALID, name, gehalt
```
erhält man die

Ausgabe: *, 1980, Hans, 1000
+, 1982, Hans, ----

"*" bezeichnet die Geburt, "+" den Tod des Objektes, das durch die Projektionsliste gebildet und durch die Auswahlbedingung identifiziert wird. Oder anders ausgedrückt: alle Aussagen zwischen einem "*" und dem darauffolgenden "*" oder "+" bilden eine Geschichtsausprägung, deren Typ durch die Projektionsliste beschrieben wird.

(4) Ausgehend von einem bestimmten Wert sollen die Zeitpunkte ermittelt werden, an denen der Wert gültig war:
Wann wurde der Etat des Projektes P1 auf 50.000 gesetzt ?

```
SELECT      VALID
FROM        projekte
WHERE       (name = "P1") WHEN (etat = 50.000)
```

Ausgabe: 1980
1984

(5) Bei der Zeit-Wert-Intervallanfrage sollen alle Werte innerhalb eines bestimmten Zeitintervalls ermittelt werden:
In welchen Orten war Hans bisher wohnhaft ?

```
SELECT      wohnort
FROM        angestellte
WHERE       (name = "Hans") DURING (*, +)
```

Ausgabe: Kaiserslautern

(6) Bei der Wert-Zeit-Intervallanfrage werden die Gültigkeitsintervalle zu einem bestimmten Wert ermittelt:
In welchen Zeiträumen lag der Etat von Projekt P1 über 200.000 ?

```
SELECT      VALID, etat
FROM        projekte
WHERE       (name = "P1") WHILE (etat < 200.000)
```

Ausgabe: *, 1980, 50.000
+, 1982, -------
*, 1984, 50.000
1986, 150.000

Wird der Etat nicht selektiert, ergibt sich folgende

Ausgabe: *, 1980
+, 1982
*, 1984

(7) Bei den bisherigen Anfragen bildete jeweils eine Relation die Grundlage. Bei der folgenden Anfrage sollen alle drei Relationen einbezogen werden. Bei der Anfrage handelt es sich um eine Zeit-Wert-Punktanfrage:
Welche Angestellten arbeiteten in welcher Funktion in Projekt P1?

```
SELECT    name, funktion
FROM      angestellte, arbeitet_mit, projekte
WHERE     (name = mitarbeiter and pro_nr = projekt_nr) AT NOW and
          (bezeichnung = "P1") DURING (*,+)
```

Ausgabe: Hans Progr.
Hans Leiter
Fritz Progr.

Der Zusatz AT NOW kann im Prinzip entfallen. Besitzt die Anfrage keinen expliziten Zeitbezug wird immer die aktuellste Sicht zugrunde gelegt.

(8) Bei dieser Anfrage soll gezeigt werden, daß auch zeitliche Abhängigkeiten zur Qualifikation des Ergebnisses herangezogen werden können:
Welche Angestellten wurden während ihrer Mitarbeit in Projekt P1 in den Teilbereich DV versetzt, als Fritz in P1 mitarbeitete ?

```
SELECT    name
FROM      angestellte, arbeitet_mit, projekte
WHERE     (name = mitarbeiter and pro_nr = projekt_nr)
WHEN      (teilbereich = "DV") OVERLAPS  (WHILE (name = "fritz" and
          mitarbeit = "P1"))
```

Ausgabe: Hans

4. Zusammenfassung und Ausblick

Nach einer Präzisierung des Begriffs "Geschichte", die zu einer Klassifikation mit vier verschiedenen Geschichtsarten führt, wird die Geschichtsentstehung untersucht. Dabei wird besonders die Bedeutung der Gültigkeitszeit gegenüber der Aufzeichnungszeit herausgearbeitet. In einem weiteren Schritt wird eine Klassifikation zeitbehafteter Anfragen entworfen.

Anschließend wird gezeigt, wie das E/R-Modell zur Darstellung zeitbehafteter Informationen erweitert werden kann. Die so gebildeten zeitbehafteten Entities und Relationships werden mit Hilfe des Konzepts der Geschichtsausprägungen in das erweiterte Relationenmodell integriert. Nachdem die Semantik eines "zeitbehafteten Verbundes" erklärt wurde, wird eine Erweiterung von SQL vorgestellt, die auf der Basis von Geschichtsausprägungen und einigen speziellen temporalen Prädikaten die Formulierung auch komplexer zeitbehafteter Anfragen erlaubt. Die Mächtigkeit der Sprache wird abschließend an einem Beispiel demonstriert.

Das Modell wird zur Zeit als Zusatzebene auf dem Non-Standard-DBS PRIMA /Hä88/ implementiert. Dabei liegt der Schwerpunkt auf der Verwendung des Molekül-Konzeptes zur Realisierung der Geschichtsausprägungen und der Technik der Anfragemodifikation zur Übersetzung der zeitbehafteten Anfragen in "zeitlose" MQL-Anfragen. Weiterhin wird die Tauglichkeit der herkömmlichen Zugriffspfadstrukturen (wie z.B. B*-Baum, R-Baum, Grid-File), die von PRIMA bereitgestellt werden, für die Aufgaben der Zeitverwaltung untersucht. Erste Überlegungen zur Modellierung der Geschichtsausprägungen mit MAD, in der bereits Differenztechniken zur Speicherplatzersparnis berücksichtigt werden, sind in /Kä88/ niederlegt.

Parallel dazu finden Überlegungen zur Erweiterung der Mächtigkeit des Geschichtsmodells statt. In einem zweiten Schritt soll MQL so erweitert werden, daß zeitbehaftete Anfragen auf Molekülen ermöglicht werden. Dazu sind noch einige interessante Probleme zu lösen, die insbesondere den Aufbau und die Qualifikation zeitbehafteter Moleküle mit Hilfe der nunmehr zeitbehafteten Referenzen betreffen.

Danksagung:

Ich möchte mich bei Herrn Prof. Dr. Härder für die Anregung bedanken, mich mit dieser Thematik zu befassen. Herrn Mitschang möchte ich für das sorgfältige Lesen und die hilfreichen Diskussionen bei der Entstehung dieses Manuskriptes danken.

5. Literatur

AC75 : Astrahan, M.-M., Chamberlin, D.-D.: Implementation of a structured English query language, CACM, Vol. 18, No. 10, Oct. 1975, pp 580-588

Ar86a : Ariav, G.: Preserving the Time Dimension in Information Systems, Ph. D. thesis, University of Pennsylvania, 1986

Ar86b : Ariav, G.: A Temporally Oriented Data Modell, ACM TODS, Vol. 11, No. 4, Dec. 1986, pp 499-527

BD83 Bolour, A. ,Dekeyser, L.-J.: Abstractions in Temporal Information, Information Systems, Vol. 8, No. 1, 1983, pp 41-49

Bo82 : Bolour, A., Anderson, T.-L., Dekeyser, L.-J., Wong, H.-K.-T.: The Role of Time in Information Processing: A Survey, ACM SIGMOD Record, Vol. 12, No. 3, 1982

Br72 : Bruce, B.: A Model for Temporal Reference and its Application in a Question Answering Program, AI, Vol. 3, No. 1, 1972 aus: Klopprogge, M. : Gegenstands- und Beziehungsgeschichten: Ein Konzept zur Beschreibung und Verwaltung zeitveränderlicher Information in DB, Dissertation, Universität Karlsruhe, 1983

Bu77 : Bubenko, J.-A.: The Temporal Dimension in Information Modeling, in: Nijssen, G.-M.: Architecture and Models in Data Base Management Systems, North Holland, 1977

Ch76 : Chen, P.-P.: The Entity-Relationship Model - Toward a Unified View of Data, ACM TODS, Vol. 1, No. 1, 1976, pp 9-36

Da84 : Dadam, P., Lum, V., Werner, H.-D.: Integration of TIME Versions into a Relational Database System, Proceedings of the 10th Int. Conf. on VLDB, Singapore, 1984, pp 509-522

De85 : Deppisch, U., Obermeit, V., Paul, H.-B., Schek, H.-L., Scholl, M., Weikum, G.: Ein Subsystem zur stabilen Speicherung versionenbehafteter, hierarchisch strukturierter Tupel, in: Proc. der GI-Fachtagung Datenbank-Systeme für Büro, Technik und Wissenschaft, IFB Nr. 94, Springer-Verlag, Karlsruhe, 1985, S. 421-440

Fe85 : Ferg, S.: Modelling the Time Dimension in an Entity-Relationship Diagram, in: Proc. of the 4th Int. Conf. on Entity-Relationship Approach, IEEE, Chicago, Illinois, Oct. 1985, pp 280-286

Hä84 : Härder, Th.: Überlegungen zur Modellierung und Integration der Zeit in temporalen Datenbanksystemen, Forschungsbericht Nr. 19/84, SFB 124, Uni Kaiserslautern, 1984

Hä87 : Härder, Th., Meyer-Wegener, K., Mitschang, B., Sikeler, A.: PRIMA A DBMS Prototype Supporting Engineering Applications, Proc. of the 13th Int. Conf. on VLDB, Brighton, Great Britain, 1987, pp 433-442

Hä88 : Härder, Th. (ed.): The PRIMA Project - Design and Implementation of a Non-Standard Database System, SFB-Bericht Nr. 26/88, SFB 124, Uni Kaiserslautern, 1988

He75 : Held, G.-D., Stonebraker, M.-R., Wong, E.: INGRES - A relational data base system, in: Proc. of the 1975 National Computer Conf., vol. 44, AFIPS Press, Reston, Va., 1975, pp 409-416

Kä88: Käfer, W.: Modellierungen von "Geschichtsausprägungen" mit dem MAD-Modelll, Arbeitsbericht, Universität Kaiserslautern, 1988

Kl81 : Klopprogge, M.-R.: TERM An Approach to Include the Time Dimension in the Entitiy-Relationship Model, in: Proc. of the 2nd Int. Conf. on Entity-Relationship Approach, 1981, pp 477-512

Lu84 : Lum, V., Dadam, P., Erbe, R., Guenauer, J., Pistor, P., Walch, G., Werner, H., Woodfill, J.: Designing DBMS Support for the Temporal Dimension, in: Proc. of the SIGMOD Int. Conf. on Management of Data, 1984, pp 115-130

LJ88 : Lorentzos, N.-A., Johnson, R.-G.: An Extension of the Relational Model to Support Generic Intervals, in: Proc. of Advances in Database Tchnology - EDBT '88, Int. Conf. on Extending Database Technology, Venice, Italy, March 1988, Springer-Verlag, pp 528-542

Mi87 : Mitschang, B.: MAD - ein Datenmodell für den Kern eines Non-Standard-Datenbanksystems, GI-Fachtagung "Datenbanksysteme für Büro, Technik und Wissenschaft", Darmstadt, in: IFB Bd. 136, Springer-Verlag, Berlin, Heidelberg, 1987, S. 180-195

MS83 : Müller, Th., Steinbauer, D.: Eine Sprachschnittstelle zur Versionenkontrolle in CAM Datenbanken, in: Schmidt, J.-W.: Sprachen für Datenbanken, IFB Bd. 72, Springer-Verlag, Berlin, Heidelberg, 1983, S. 76-95

Ni82 : Nilsson, N.-J.: Principals of Artificial Intelligence, Springer-Verlag, Berlin, Heidelberg, New York, 1982

OS82 : Overmyer, R., Stonebreaker, M.: Implementation of a Time Expert in a Data Base System, ACM SIGMOD RECORD, Vol. 12, No. 3, 1982, pp 51-60

PA86 : Pistor, P., Anderson, F.: Desigining a Generalized NF^2 Data Model with an SQL-type Language Interface, in: Proc. of the 12th Int. Conf. on VLDB, Kyoto, Japan, 1986, pp 278-288

SA86: Ahn, I., Snodgrass, R.: Performance Evaluation of a Temporal Database Management System, Proc. of the ACM SIGMOD Int. Conf. on Management of Data, Washington, 1986 (Record Vol. 15, No. 2, June 86)

Sch85: Schek, H.-J.: Towards a Basic Relational NF^2 Algebra Processor, in: Proc. of the Int. Conf. on Foundations of Data Organization, Kyoto, Japan, 1985, pp 173-182

Sn86a: Snodgrass, R.: Temporal Databases, IEEE COMPUTER, Vol. 19, No. 6, Sept. 1986, pp 35-42

Sn86b: Snodgrass, R.: Research Concerning Time in Databases: Project Summaries, ACM SIGMOD RECORD, Vol. 15, No. 4, 1986, pp 19-39

Sn87: Snodgrass, R.: The Temporal Query Language TQuel, ACM TODS, Vol. 12, No. 2, June 1987, pp 247-298