3. Temporale Datenmodelle

Es sind viele Varianten von temporalen Datenmodellen vorgeschlagen wurden. Diese unterscheiden sich hinsichtlich

- des unterliegenden Datenmodells, z.B. Relationen-/ER-/OO-Modell
- der für **Zeitstempel** (Zeitmarkierungen) verwendeten temporalen Datentypen, meistens Zeitpunkte, Intervalle oder temporale Elemente
- der verwendeten Zeitdimensionen: z.B. Gültigkeitszeit
- der Gegenstände der Zeitstempel, also der zeitmarkierten Einheiten (evtl. aggregierten Einheiten) gemäß Datenmodell, z.B. Tupel (s.u.)

Bei den Zeitstempeln (und auch bei deren Gegenständen) kann ein temporales Datenmodell bis zu drei Repräsentationen betrachten:

- semantisch (Konzept): z.B. Mengen von Zeitpunkten
- logisch (Präsentation, Ein-/Ausgabe): z.B. Listen maximaler Intervalle
- intern (Speicherung): z.B. einzelne beliebige Intervalle

Semantische und logische "Temporalisierung" des ER-Modells

Hier betrachten wir nur Gültigkeitszeit. Jedes Entity und jedes Relationship wird mit einer Lebensdauer versehen, jedes Attribut mit einer zeitabhängigen Wertzuordnung. Auch jede (Entity, Attribut/Wert)-Kombination kann quasi mit einer "Lebensdauer" versehen werden. *Formal:*

Temporales ER-Modell:

[In einem Zustand der temporalen DB gilt:]

- Entity $e \mapsto T(e) \subseteq^t \mathcal{T}$ Lebensdauer von e
- Attribut $e.A \mapsto (T(e) \to \text{range}(A))$ zeitabhängige Wertzuordnung
  Schlüsselattribute müssen zeitinvariant sein (konstante Funktion).
- Attributwert $(e.A{=}v) \mapsto T(e.A{=}v) := e.A^{-1}(v) \subseteq^t \mathcal{T}$
- Relationship $r = (e_1, ..., e_n) \mapsto T(r) \subseteq^t \mathcal{T}$ Lebensdauer von r
  Dafür gilt folgende (inhärente) IB: $T(r) \subseteq T(e_1) \cap ... \cap T(e_n)$

$\mathcal{T}$ sei die Menge aller Zeitpunkte. Die Notation $E \subseteq^t \mathcal{T}$ meint, dass E ein temporales Element über $\mathcal{T}$ sein muss (nicht eine beliebige Teilmenge). "range(A)" bezeichnet den Wertedatentyp eines Attributs A. Lies "$x \mapsto$ ..." als "einem x wird ... zugeordnet".

Semantische und logische "Temporalisierung" des ER-Modells (Forts.)

Zeit(punkt)abhängige Abbildungen eignen sich nur zum Verständnis der Semantik.

Selbst die (durch endliche Listen von Intervallen darstellbaren) temporalen Elemente sind erfahrungsgemäß oft etwas gewöhnungsbedürftig, weil sie

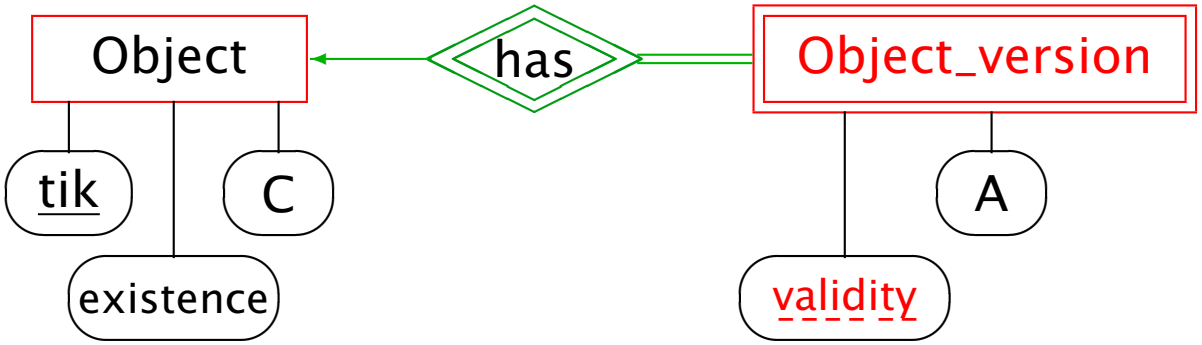
- das Verschwinden und Wiedererscheinen von Entities erlauben
- und mehrfache Gültigkeiten gleicher Werte zusammenfassen.

Zudem sind solche Zeitstempel an Attributen nicht (einfach-) relational umsetzbar.

Alternativ wird deshalb gern die Modellierung von aufeinanderfolgenden **temporalen Versionen der Entities** (Objekte) verwendet, die jeweils nur ein Intervall als Zeitstempel tragen. Diese lässt sich sogar im klassischen ER-Modell spezifizieren (s. u.); dabei wird *angenommen:*

- dass ungültig gewordene Objekte nicht wieder gültig werden (ggf. sind neue Objekte zu erzeugen),
- und dass immer dann eine neue Version eines Objekts gebildet wird, wenn sich der Wert irgendeines seiner Attribute ändert.
  (vgl. 'Bob'-Tupel im Beispiel aus Kap. 1)

Semantische und logische "Temporalisierung" des ER-Modells (Forts.)



- Object — Name eines Entitytyps
- tik — zeitinvarianter Schlüssel ("time invariant key", evtl. interne ID)
- existence — Lebenszeit-Intervall
- C — weitere zeitinvariante Attribute
- validity — Gültigkeits-Intervall für eine Attribute-Ausprägung

  Man beachte folgende IBen: Zu einem Objekt o und einer zugehörigen Objektversion ov muss $ov.validity \subseteq o.existence$ gelten. Zu verschiedenen Objektversionen eines Objekts o müssen die validity-Intervalle paarweise disjunkt sein und das o.existence-Intervall vollständig überdecken.
- A — zeitvariante Attribute

  Attribute brauchen dann keine Gültigkeitsangaben mehr.

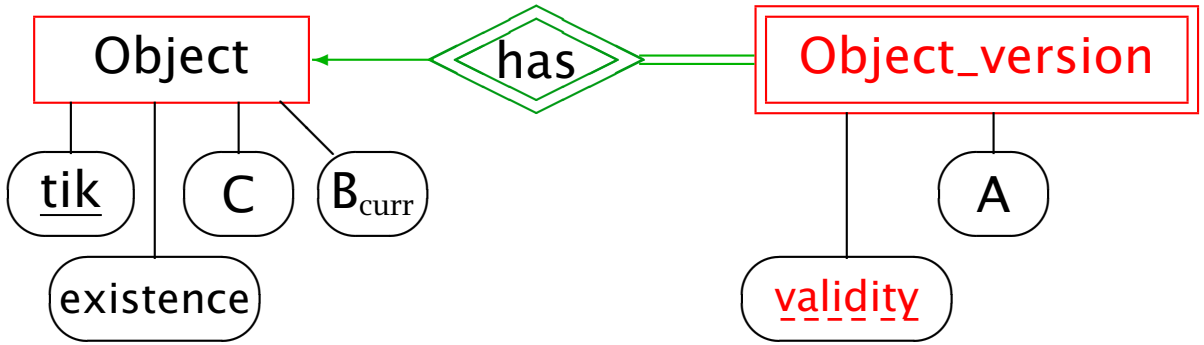
Semantische und logische "Temporalisierung" des ER-Modells (Forts.)

Praktische Varianten von Objektversionen:

(i) Partielle Temporalisierung:

Man verzichtet auf die temporale Buchführung einiger, eigentlich zeitvarianter Attribute und speichert nur deren aktuelle Werte.

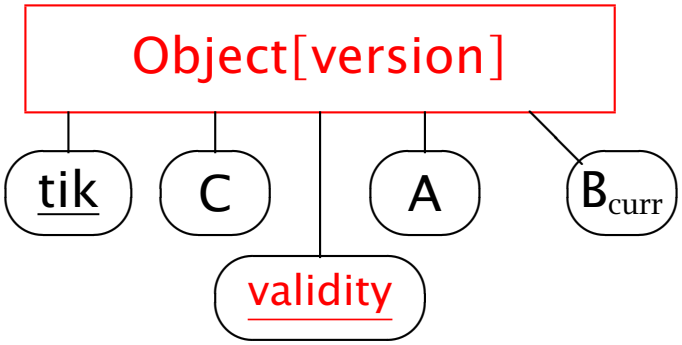
— Das passt ins obige Modell, indem einige Attribute (B) nicht an Object_version, sondern an Object gehängt werden, dort aber normal änderbar sind. (C – zeitinvariante, B_{curr} – aktuelle Werte)



Semantische und logische "Temporalisierung" des ER-Modells (Forts.)

(ii) Implizite Objektbildung:

Man verzichtet auf die Unterscheidung von Objekt und Objektversion (und von existence und validity) verzichtet und modelliert nur:



- Der Zusammenhang der Versionen eines Anwendungsobjekts ergibt sich dann nur noch über den gleichen tik.
- Zeitinvariante Attribute (tik, C) werden redundant gespeichert.
- Einige zeitvariante Attribute (A) bleiben "versionsbildend", d.h. bei Änderung wird eine neue Version erzeugt (validity aktualisieren!).
- Von anderen (B) werden nur aktuelle (letzte) Werte zur selben Version gespeichert.

Semantische und logische "Temporalisierung" des ER-Modells (Forts.)

Für **temporale Relationshiptypen** wird mit obiger IB angenommen, dass ein zu einem Zeitpunkt t gültiges Relationship immer nur zum Zeitpunkt t existente Entities verbindet.

(Daneben könnte es auch zeitübergreifende bzw. zeitunabhängige Relationships geben.)

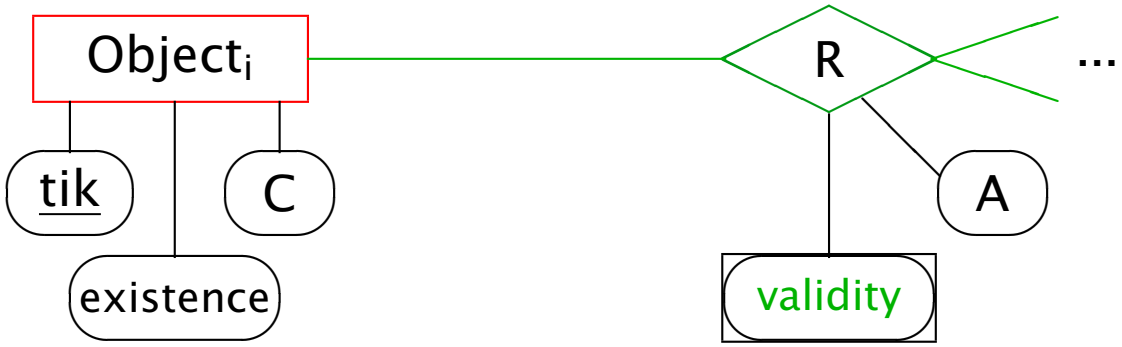
Temporale Relationships lassen sich im Umfeld temporaler Objektversionen (Object has Object_version) wie folgt modellieren:

a) Objekt-Relationships: Temporale Relationships verbinden Objekte.

Gemeint ist, dass ein zu einem Zeitpunkt t gültiges Relationship zwischen Objekten *implizit* die zum Zeitpunkt t gültigen Objektversionen verbindet. Somit wird für einen Relationshiptyp R eine (unabhängige) validity-Angabe benötigt.

Semantische und logische "Temporalisierung" des ER-Modells (Forts.)

a) Objekt-Relationships: (Forts.)



Damit sich Objektkombinationen wiederholen und zeitvariante Relationshipattribute (A) gespeichert werden können, muss validity eigentlich ein künstliches Entity mit singulärem Attribut sein.

Ein R-Relationship r ist dann durch eine Kombination von Objekten o_i der beteiligten Entitytypen und ein Gültigkeitsintervall v bestimmt: $r = (o_1, ..., o_i, ..., o_n, v)$.

Die obige IB gilt dann gdw. für alle solche r und alle i gilt:

$$v \subseteq o_i.\text{existence}$$

Semantische und logische "Temporalisierung" des ER-Modells (Forts.)

b) Versions-Relationships:

Temporale Relationships verbinden Objektversionen, nicht Objekte.

Bei Versionswechseln sind dann weiterbestehende Relationships auf die neue Version zu kopieren, was bezogen auf Objekte *redundant* erscheint.

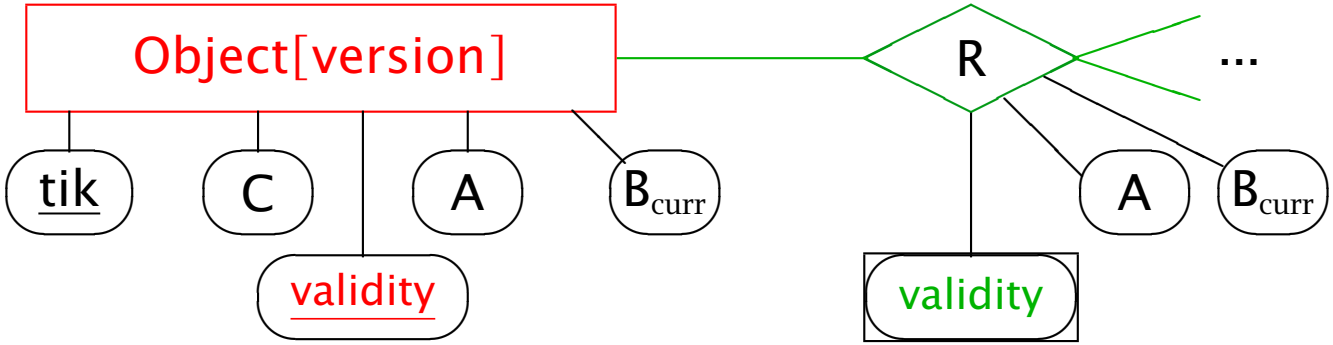
Ein R-Relationship r ist durch eine Kombination von Objektversionen ov_i und ein Gültigkeitsintervall v bestimmt. Die obige IB lautet hier: $v \subseteq ov_i.validity$ (für alle i)

Man könnte alternativ von jedem Objekt auch dann eine neue Version bilden, wenn sich die Beteiligung an einem Relationship ändert. Damit zöge jede Versionsbildung eines Objekts auch Versionsbildungen der verbundenen Objekte nach sich, so dass die validity-Intervalle von über Relationships verbundenen Objektversionen übereinstimmen. Statt IB würde hier für ein Relationship $r = (ov_1, ..., ov_n)$ immer gelten: $ov_i.validity = ov_j.validity$ (für alle i, j). Immerhin bräuchten Relationships so keine Gültigkeitsangaben mehr. — *Zu aufwändig!*

Semantische und logische "Temporalisierung" des ER-Modells (Forts.)

Praktische Kombination:

- implizite Objektbildung (durch Versionen)
- einschl. partieller Temporalisierung
- zusammen mit Versions-Relationships



IB für ein Relationship r : $r.validity \subseteq r.Object_i.validity$

Beispiel:

Im folgenden ER-Schema einer (überwiegend) temporalen Datenbank, nämlich für den Lehrveranstaltungs-/Modulkatalog kann man (teilweise) die obige "praktische Kombination" von Modellierungsvarianten wiedererkennen. Die Datenbank ist klassisch relational implementiert, leider ohne temporale Software.

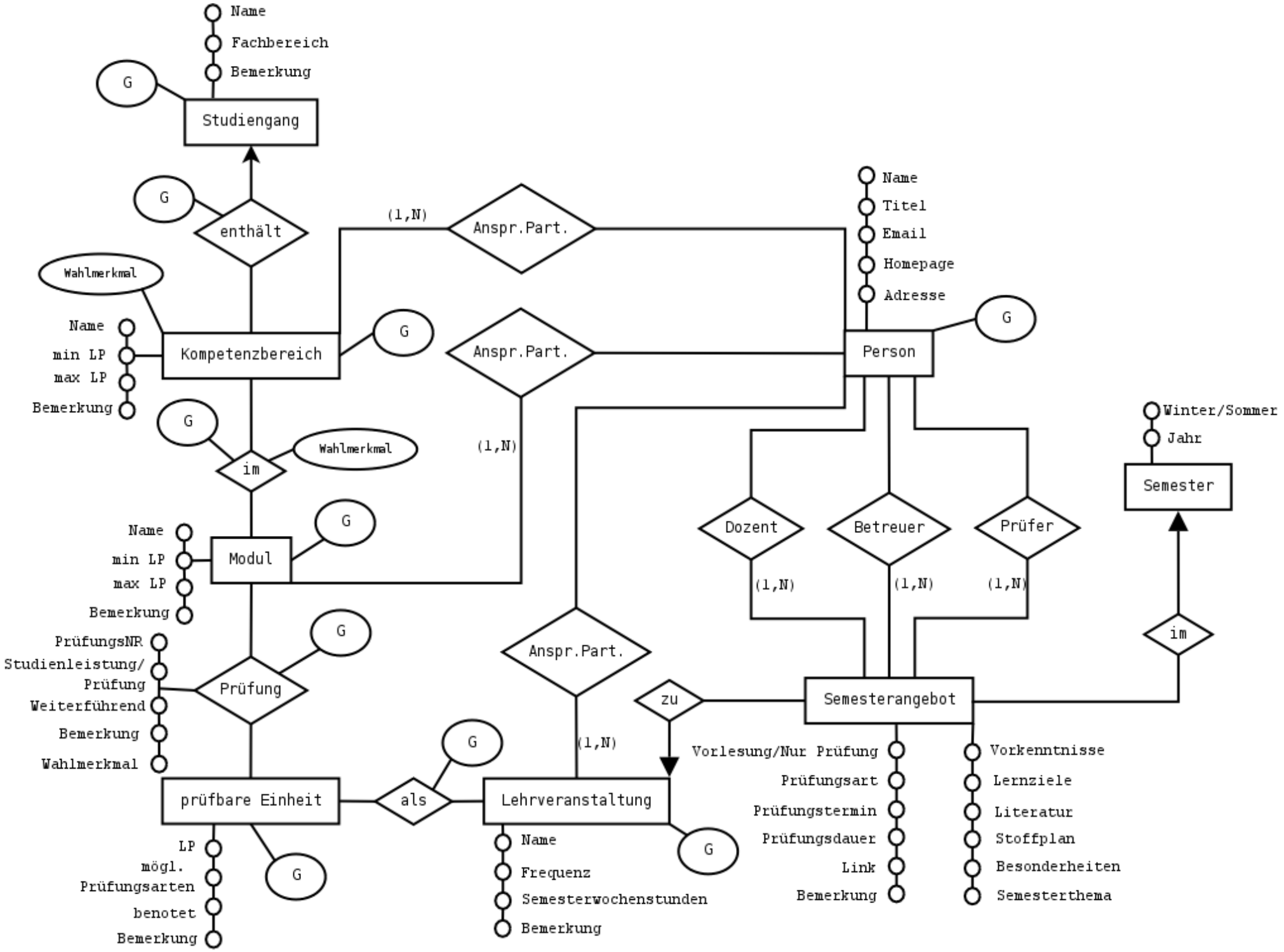
Die Gültigkeitsintervalle heißen hier "G" und beziehen sich auf Semester (als Kalendergranularität).

Zu allen Entities gibt es allerdings bisher nur ID-Attribute als Versionsschlüssel und keine zeitinvarianten Attribute.

Es gibt versionsbildende Attribute, um prüfungsrelevante Informationen temporal zu dokumentieren, z.B. Wahlmerkmal, LP, benotet, SWS. Andere Attribute werden nur aktuell gehalten, z.B. Bemerkungen, Frequenz; analog die nicht-temporalen Relationships. Da neue Versionen manuell angelegt werden müssen, sind versionsbildende Attribute auf das nötigste beschränkt.

Die IB zwischen temporalen Entities und temporalen Relationships wird durch passive oder aktive Trigger gesichert. Z.B. werden bei den meisten Updates einer Entity-Gültigkeit zugehörige Relationship-Gültigkeiten ggf. verkürzt.

Das auch temporale Entity Semesterangebot bildet einen Spezialfall: es hat als Gültigkeit das eine, zugeordnete Semester (also quasi nur einen Zeitpunkt), und es ist eigentlich ein schwaches Entity bzgl. Lehrveranstaltung; die Gültigkeit eines schwachen Entities muss immer in der des identifizierenden Entities enthalten sein. Andererseits bleiben hier Updates von Lehrveranstaltung.G eingeschränkt. Die Gültigkeit der Relationships Dozent,... ergibt sich implizit als das gleiche Semester.


Name
Fachbereich
Bemerkung
G
Studiengang
G
enthält
(1,N)
Anspr.Part.
Name
Titel
Email
Homepage
Adresse
Wahlmerkmal
Name
min LP
max LP
Bemerkung
Kompetenzbereich
G
Anspr.Part.
Person
G
G
im
Wahlmerkmal
(1,N)
Winter/Sommer
Jahr
Semester
Name
min LP
max LP
Bemerkung
Modul
G
Dozent
Betreuer
Prüfer
(1,N)
(1,N)
(1,N)
im
PrüfungsNR
Studienleistung/
Prüfung
Weiterführend
Bemerkung
Wahlmerkmal
Prüfung
G
Anspr.Part.
zu
Semesterangebot
(1,N)
G
Vorlesung/Nur Prüfung
Prüfungsart
Prüfungstermin
Prüfungsdauer
Link
Bemerkung
Vorkenntnisse
Lernziele
Literatur
Stoffplan
Besonderheiten
Semesterthema
prüfbare Einheit
als
Lehrveranstaltung
LP
mögl.
Prüfungsarten
benotet
Bemerkung
G
Name
Frequenz
Semesterwochenstunden
Bemerkung
G

"Temporalisierung" des Relationen-Modells

Temporales Relationenmodell:

- **attribute time stamping**:

  im Tupel zum Entity e (bei unverändertem Relationenschema):

  – Tupelkomponente zu Schlüsselattribut mit Wert k : $(k, T(e))$,
  also Schlüsselwert + Existenzzeit von e

  – Tupelkomponente zu Nicht-Schlüssel-Attribut A:

  $$\{ (v, T(e.A{=}v)) \mid v \text{ Attributwert während } T(e) \}$$

  also Attributwert + Gültigkeitszeit

Beachte: Zu speichern wären so nicht mehr nur atomare Werte (bekannt als 1NF = 1.Normalform), sondern jeweils ein Attributwert zusammen mit einem temporalen Element, also einer Vereinigung von Intervallen. Dies erfordert allerdings geschachtelte Relationen (wie in objekt-relationalen DBMS).

EMP	Name	Salary	Title
	Harry	25000, [01.01.2001-15.10.2003) ∪ [01.10.2004-01.01.2007)	Junior Res.Ass., [01.01.2001-01.01.2003) Senior Res.Ass., [01.01.2003-01.10.2003) ∪ [01.10.2004-01.01.2007)
		40000, [15.10.2003-01.10.2004)	Repl. Professor, [01.10.2003-01.10.2004)

"Temporalisierung" des Relationen-Modells (Forts.)

- oder **tuple time stamping**:
  mit Relationenschemata der Form: R(tik, C, ..., A, ..., Start, Stop)
  - tik zeitinvarianter Schlüssel
  - C, ... zeitinvariante Attribute
  - A, ... zeitvariante Attribute: sollten möglichst synchronen Änderungen unterliegen, sich also nicht unabhängig voneinander ändern (= temporale Normalform[1]) [2]
  - Start, Stop Anfangs-/Endezeitpunkt eines maximalen(!), rechtsoffenen Zeitintervalls, in dem die jeweilige (tik, C, ..., A, ...)-Wertekombination gültig ist[3]

z. B.

EMP	Name	Salary	Title	Start	Stop
	Harry	25000	Junior Res.Ass.	01.01.2001	01.01.2003
	Harry	25000	Senior Res.Ass.	01.01.2003	01.10.2003
	Harry	25000	Repl. Professor	01.10.2003	15.10.2003
	Harry	40000	Repl. Professor	15.10.2003	01.10.2004
	Harry	25000	Senior Res.Ass.	01.10.2004	01.01.2007

[1] ggf. sind Relationenschemata erst passend zu zerlegen, im Bsp.: (Name, Salary, Start, Stop) und (Name, Title, Start, Stop)
[2] und evtl. auch hier einige nur-aktuelle Attribute B,...
[3] alternativ: nur die Startwerte, da letzter Stopwert normalerweise unbestimmt "gilt bis in die Zukunft"

"Temporalisierung" des Relationen-Modells (Forts.)

Inhärente Integritätsbedingungen für temporale Relationen mit *tuple time-stamping*:

(a) Eindeutige temporale Identifikation von Tupeln durch den tik ("time invariant key") + Zeitstempel; d.h. zu jeder (tik, Zeitpunkt t)-Kombination darf es in jeder aktuellen Ausprägung von R nur ein Tupel (tik, ..., Start, Stop) mit $t \in$ [Start, Stop) geben.

In diesem Sinne kann tik als `PRIMARY KEY` einer temporalen Relation angegeben werden.

Ohne Bezug auf beliebige Zeitpunkte kann man Bedingung (a) äquivalent durch folgende Bedingungen (a1,a2) ausdrücken, die nur die expliziten Zeitstempel = [Start, Stop)-Intervalle und etwaige Überlappungen solcher Intervalle betrachten.

(a1) Temporale Widerspruchsfreiheit: Zeitstempel von Tupeln mit gleichem tik, aber verschiedenen zeitvarianten Attributen dürfen sich nicht überlappen.

(a2) Temporale Redundanzfreiheit: Zeitstempel von Tupeln mit gleichem tik und gleichen zeitvarianten Attributen dürfen sich nicht überlappen.

"Temporalisierung" des Relationen-Modells (Forts.)

Weitere inhärente IBen:

(b) Temporale Maximalität: Zeitstempel von Tupeln mit gleichem tik und gleichen zeitvarianten Attributen dürfen nicht benachbart sein; solche müssen also zu einem Intervall verschmolzen worden sein (Angabe: COALESCED).

(c) Temporale Vollständigkeit: Zwischen den Zeitstempeln von Tupeln mit gleichem tik darf es keine Lücke geben (Angabe: COMPLETE).

(d) Temporale Fremdschlüssel: Wie üblich angegebene FOREIGN KEY-Beziehungen ($R.A \rightarrow S.tik'$) müssen zu jedem Zeitpunkt jedes Quelltupels gelten. D.h. der Zeitstempel eines R-Quelltupels muss in der Vereinigung der Zeitstempel von passenden S-Zieltupeln ($R.A = S.tik'$) als Teilintervall enthalten sein.