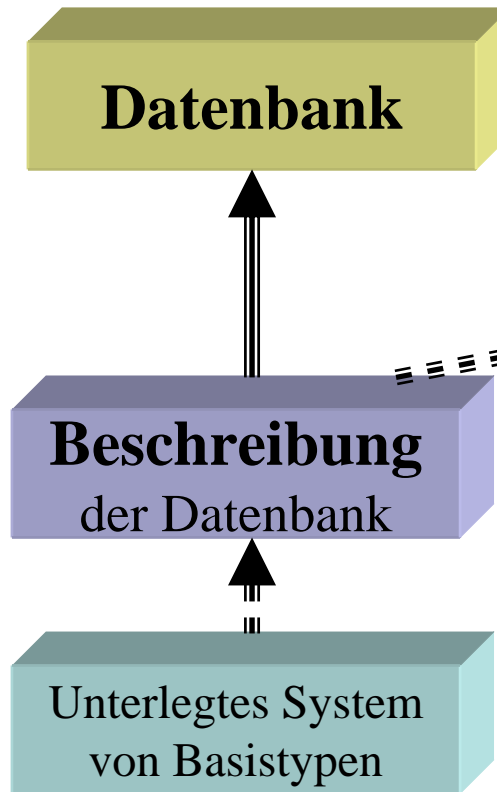


5. Datenbank-Modellierung

3 ★ Schichten
einer Datenbank

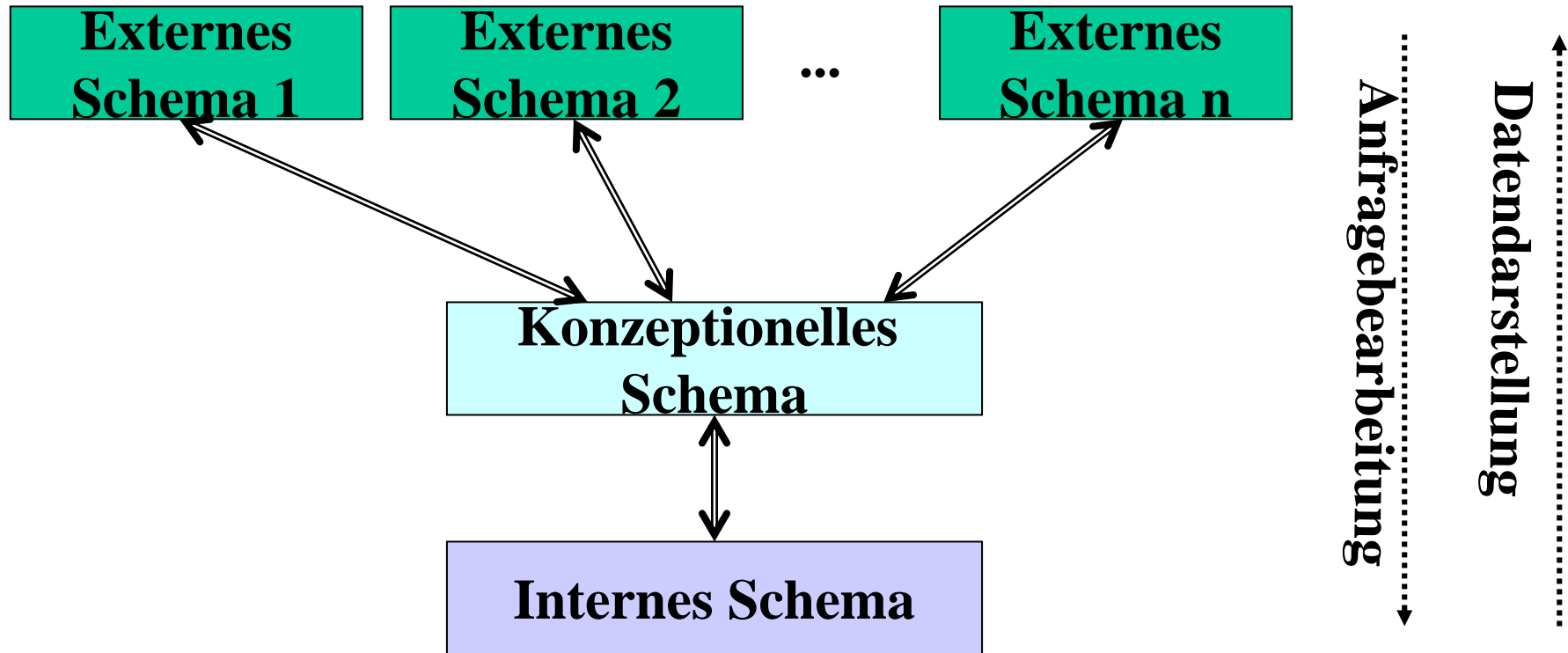
**Data
Dictionary**



Konzeptionelle Schicht:
(E)ER-Modell
Logische (Implem.-)Schicht:
relationales,
hierarchisches
Netzwerk-Modell
Implementationsschicht:
Physische Modelle

integer, real, float, decimal,
Boolean, date, time

3-Ebenen-Architektur für Schemata von Datenbankanwendungen



**Modellierungsobligationen: a) konzeptionelles Schema
b) Sichten**

Problem: **Redundanz** von Daten in einer Datenbank

Daten sind **mehrfach** in einer Datenbank **vorhanden**

⇒ Mehrfachveränderungen

z.B. Tabelle mit Schülern(amen), deren Nr., Adressen, Klasse etc. und

Tabelle mit Klasse, Nr., Schülername, Fach, Noten

wird nun Name geändert in einer Tabelle, dann ?

Daten können **erschlossen** werden

bzw. sind erschließbar auf unterschiedliche Art (versch. Berechnung)

z.B. Adresse mit PLZ etc. in DB, sowie Stadt-Straßen-PLZ-Verzeichnis

z.B. Tabellen mit *Auto-*, *Personen-*, *Halterdaten* und

Blitzerdaten mit Assoziation zu *Auto* und

Zahlungsaufforderung an *Person*

aber Vorteile bei Zugriff, Verständnis (Wiederholung), Systemproblemen

z.B. über das Netz

parallelen Zugriffen mehrerer Benutzer

Performanz

Idee zur Behebung: Daten nur an Ursprung, ansonsten Sicht

oder Master-Slave-Beziehung

Funktionale Zusammenhänge der Daten

Problem: Existenz **inkonsistenter Daten**

z.B. **Modifikation** in einer Tabelle ohne Modifikation in einer anderen

z.B. Daten zu Räumen, ihrer Ausstattung, die dafür notwendige Installation, den verantwortlichen Betreuern usw.
erfordern bei Modifikation einer Eigenschaft Modifikation einer Reihe von Daten

z.B. bei Bearbeitung von **Makrodaten**

Institut	Math	Inf	ZTG	insges.
WS98/99-Vorlesungen	25	20	32	44
Summer School 99	0	0	0	9

Probleme mit Überdeckungen (Inklusion/Exklusion) und Vollständigkeit

Probleme mit kontinuierlichen Daten (insbesondere Monatstatistiken)

z.B. **abgeleitete Daten**

Kontostand als Resultat einer Reihe von Modifikationsoperationen
bzw. als Zusammenfassung
mit Speicherung (und Weitergabe, ..., Datum für Bilanzen)

Wird eine Modifikation infragegestellt, dann ist auch Kontostand falsch und damit alle darauf basierenden Daten.

Problem: **Anomalien** von Operationen

Tabelle LIEFERUNG

<u>Lieferant</u>	<u>Lief-Adr.</u>	<u>Artikel</u>	<u>Anzahl</u>	<u>Kunde</u>	<u>Preis</u>	<u>Liefertag</u>

Update-Anomalie: Änderung von *Lief-Adr.* ist keine Operation für ein Datum

Insert-Anomalie: Lieferant kann nicht erfaßt werden, solange er nichts liefert
NULL ist nicht zugelassen für *Artikel*, *Kunde* und *Liefertag* (Schlüssel)

Delete-Anomalie: Werden alle Lieferungen eines Lieferanten gelöscht, dann hat er auch keine Adresse.

Warum trotzdem auf *Lieferschein*?

Potentielle Ineffizienz; Sicht benötigt
Praxis: Leben mit Inkonsistenzen oder
mit expliziter Pflege

Lösung: *Lieferschein* ist eine Sicht auf ein anomalien-freies Schema

Problem: **Instabilitäten** von Schemata bei Veränderungen

Änderungen von Schemata können zu **neuen Anomalien** führen

Änderungen der **Anwendung** führen zu neuen Sichten
,Kleckerburg‘-Syndrom

Performanz kann Veränderung notwendig machen
Inkonsistenz von logischem und physischem Schema bzgl. konzeptionellem

Daten werden in unterschiedlicher **Granularität** mit unterschiedlichem **Datentyp**
gefordert

Daten erfordern bei Modifikation das **Offenhalten** sehr unterschiedlicher
Tabellen, Blöcke, ..., wodurch lange Transaktionen verursacht werden

Erweiterungen von Schemata führen zu **Altlasten**, Alttabellen etc., deren
Management verschieden von dem der Neudaten ist

Problem: „Eines schickt sich nicht für alle ..“

Daten von **unterschiedlicher Abstraktion**

verschiedene Arten von Benutzern

z.B. Leiter, Gruppenleiter, Mitarbeiter, Bearbeiter, Mitglied eines Gremiums

Daten von **unterschiedlicher Genauigkeit**

pH-Wert mit einer Dezimalstelle, Durchschnitt mit 3

Daten von **unterschiedlicher Bedeutung**

z.B. 12. 1. 99 (12. Tag 1999, 7. Arbeitstag 1999, 31. Finanztage 1999)

Bearbeitung der Daten in verschiedenen **Rollen, Rechten**

in einem Schritt Modifikation, aber später verboten

Datenschutz (personensensible Daten (z.B. Krebsregister))

Daten in unterschiedlicher **Darstellung**

alphanumerisch, verschiedene Genauigkeiten, verschiedene Integration

Daten mit unterschiedlichen **Funktionalität**

numerische Daten, Skalarwerte, mit Nullwert bzw. Defaultwert, mit Ordnung,

mit Aufzählung; relative Daten (z.B. *Verbrauch*); Nominaldaten (z.B. *Rang*)

Daten mit unterschiedlichen **Exportformaten**

Überlagerung verschiedener Sichten (Formulartechnik) mit potentiell unterschiedlichen
Geschäftsprozessen

z.B. Dienstreiseformular der BTU

Probleme:

1. Redundanz
2. Risiko inkonsistenter Daten
3. Anomalien (insert-Anomalie: fd's verletzt bei insert,
delete-Anomalie: letzte Daten zu Beziehungen verloren,
update-Anomalie: statt Ein-Tupel-Operation viele Tupel)
4. Schemastabilität bei Änderungen
5. Abstraktionsniveau, Bedeutungen

Lösung:

1. **Normalisierung** (Aufbau von Typen, die jeweils einen Fakt darstellen)
2. Explizite **Reparaturmechanismen**
 - Erzwingung von Integritätsbedingungen
 - Zurückweisen der Operationenfolge (Transaktion)
 - Kaskadierende Operationen
 - Aufweichung der Integritätsbedingungen
 - Nullwerte, Defaultwerte

dazu erforderlich: vollständiges Wissen über

alle expliziten und **impliziten** Bedingungen (Ableitungskalkül)

Hintergrund hinter obigen Ideen

Erkennung schlechter Schemata

verbotene Teilstrukturen: Strukturen, die gewissen Eigenschaften nicht
genügen

ineffiziente Typen: schlechte Performanz ableitbar aus Struktur, Funktionalität
und verwendeten Implementationsmethoden (konzeptionelles Tuning)

Speicherung großer Tabellen nicht mehr im Zusammenhang möglich

Verfahren zur **Behebung schlechter Eigenschaften**

Heuristische Verfahren: Separation von **semantisch sinnvollen Einheiten**

z.B. *Lieferant* und *Lief-Adr.*

Modellierung mit **ER-Diagrammen**

Tuning des physischen Schemas evt. nach manuellem
Eingriff mit Monitoring

Zusammenlegen von Operationen zu konsistenter Einheit

Transaktion

Formale Verfahren: **Normalisierungsalgorithmen**

Algorithmen zur Generierung günstiger Schemata
mit expliziten Pflegemechanismen

Weitere Integritätsbedingungen:

1. UNION-Constraint

Senatoren = StudSenator + MitarbSenator + SonstMSenator + ProfSenator

damit auch IsA-Typen: partiell oder total bzw.

disjunkt oder nicht disjunkt

2. Funktionale Abhängigkeiten

Flug = ({Flug#, ChefPilot#, Von, Nach, Flugsteig, AbflZeit, Tag}, {{Flug#, Tag}}, Σ)

Attribute (Komponenten) X bedingen die Werte anderer Attribute Y

$X \rightarrow Y$

C(Flug): Tupel mit gleichen Werten über X müssen auch gleiche Werte über Y haben

$t, t' \in C(\text{Flug}): t[X] = t'[X] \rightarrow t[Y] = t'[Y]$

$\{\text{ChefPilot\#, Tag, AbflZeit}\} \rightarrow \{\text{Flug\#}\}$

$\{\text{Tag, AbflZeit, Flugsteig}\} \rightarrow \{\text{Flug\#}\}$

$\{\text{Flug\#}\} \rightarrow \{\text{Von, Nach, AbflZeit}\}$

Anmerkung: (0,1)-Kardinalität ist eine funktionale Abhängigkeit

Kalküle für Integritätsbedingungen

Funktionale Abhängigkeiten

$$\begin{array}{l} \text{Axiom:} \\ \text{Ableitungsregel:} \end{array} \quad \frac{X \cup Y \rightarrow Y \quad X \rightarrow Y, Y \rightarrow Z}{X \rightarrow Z}$$

Einführung einer Folgerungsbeziehung: $\Sigma \models \alpha$

Gilt Σ in der Relation R dann gilt auch α ..

Einführung einer Ableitungsbeziehung: $\Sigma \vdash \alpha$

Ableitung von α aus Σ mit den Axiomen und Ableitungsregeln.

Dieser Kalkül ist **korrekt** und **vollständig**, d.h. $\Sigma \models \alpha$ gdw. $\Sigma \vdash \alpha$.

Aus den fd's $\{\text{AbflZeit, Tag, ChefPilot\#}\} \rightarrow \{\text{Flug\#}\}, \{\text{Flug\#}\} \rightarrow \{\text{Von, Nach}\}$
folgt die fd $\{\text{AbflZeit, Tag, ChefPilot\#}\} \rightarrow \{\text{Von, Nach}\}$

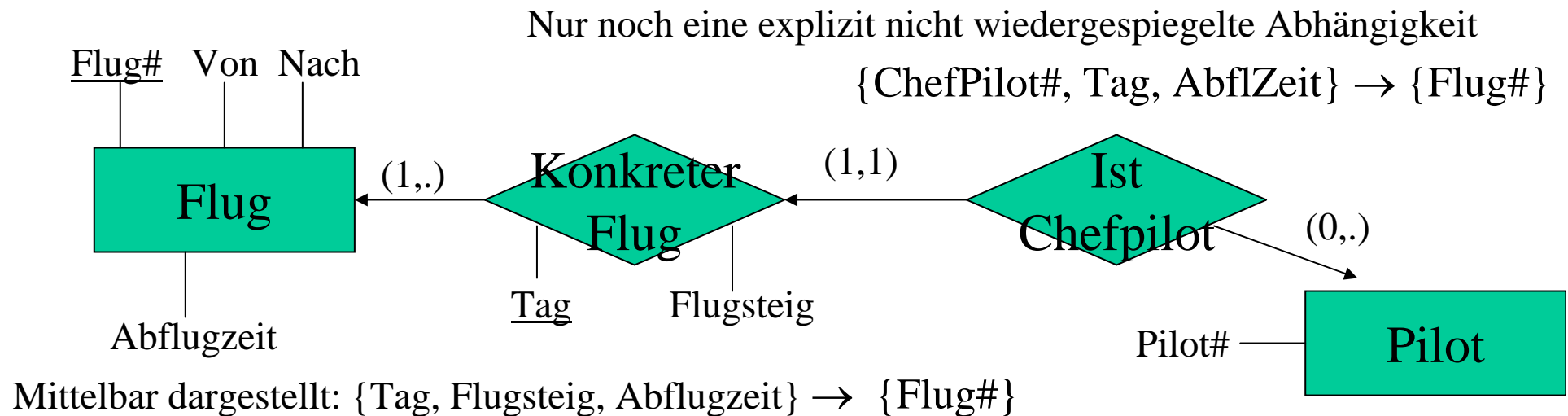
Eine Menge von fd's ist **redundant**, wenn aus einer echten Teilmenge bereits diese Menge folgt.

Zwei Mengen von fd's sind **äquivalent**, falls aus ihnen die gleichen fd's folgen.

Damit kann das Entwurfsziel wie folgt zusammengefaßt werden:

1. Herausfinden einer möglichst einfach (einfach pflegbaren), äquivalenten Menge von Abhängigkeiten.
2. Restrukturierung der Anwendung (mit Neustrukturierung von Komponenten)

Wesentlicher Zugang: (Vertikale) **Dekomposition** von Typen durch Projektion auf einzelne Komponenten



Theoretischer und algorithmischer Hintergrund

1. Spezielle Mengen

Hülle einer Menge Σ von fd's $\Sigma^+ = \{ \alpha \mid \Sigma \models \alpha \}$

Damit entsteht auch das Membership-Problem.

Einfache Lösung für Menge Σ von fd's: $X \rightarrow Y$

Rekursion: $X_{i+1} = \bigcup \{ A \in R \mid (Z \rightarrow A) \in \Sigma \wedge (Z \subseteq X_i) \}$

2. Normalformen

1. Normalform: nur atomare Attribute

2. Normalform: X minimaler Schlüssel, aber Nichtschlüsselattribut A
abhängig von echter Teilmenge von X sowie in 1. Normalform

3. Normalform: keine transitiven Abhängigkeiten $K \rightarrow X \rightarrow \{A\}$ für
Nichtschlüsselattribut A und Nichtschlüssel X

Boyce-Codd-Normalform: alle fd's sind Schlüssel-fd's

3. Wünschenswerte Eigenschaften für (Projektion, Verbund)-Dekomposition

Verbundtreue: alle Daten sind durch Verbund def. und nur solche

Abhängigkeitstreue: alle fd's lassen sich lokal auf neuen Typen prüfen

Vollständigkeit: alle fd's lassen sich aus lokalen fd's und Dekompos. erhalten

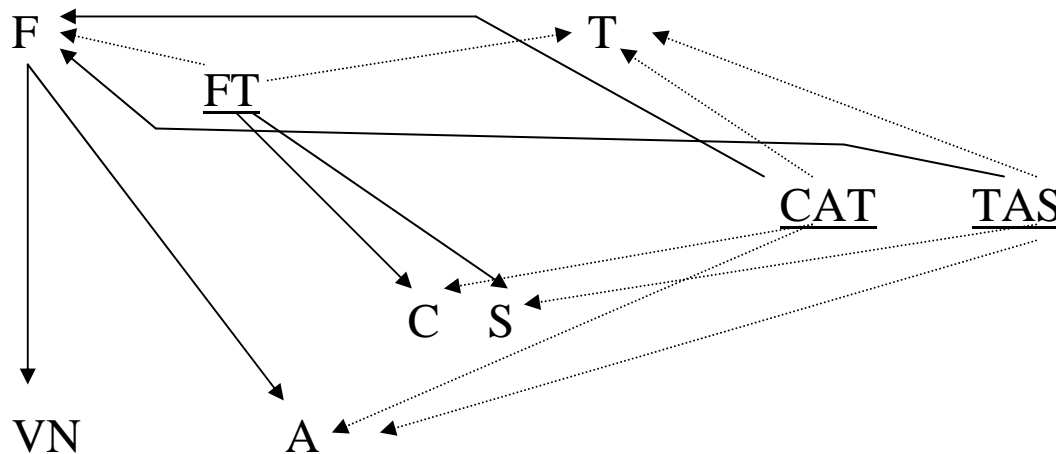
Algorithmen zur Konstruktion einer Normalform

negativ: I.a. Verbundtreue + Vollständigkeit + Abhängigkeitstreue + BCNF nicht möglich
muß vielleicht auch nicht immer gelten (Universalrelation ?, Verbund als Rekonstruktionsoperation)
evt. exponentiell

1. Dekompositionsalgorithmus

1. Berechnung aller minimalen Schlüssel
2. Bildung eines gerichteten Graphen mit der Menge der Schlüssel als Wurzel und (linksminimalen) fd's als Kanten (mit Hülle möglichst), Reduktion
3. Schrittweise Bildung eines neuen Typs für Senken und assoziierte linke Seite bzw. transitive Abhängigkeiten mit Reduktion des Graphen

Bsp: $CAT \rightarrow F$, $F \rightarrow VNA$, $TAS \rightarrow F$ $\{F,T\}, \{T,A,S\}, \{C,A,T\}$ sind minimale Schl.
 $FT \rightarrow CS$ damit $FT \rightarrow CSAVNFT$



Mögliche Dekomposition:

FVNA
FTSC

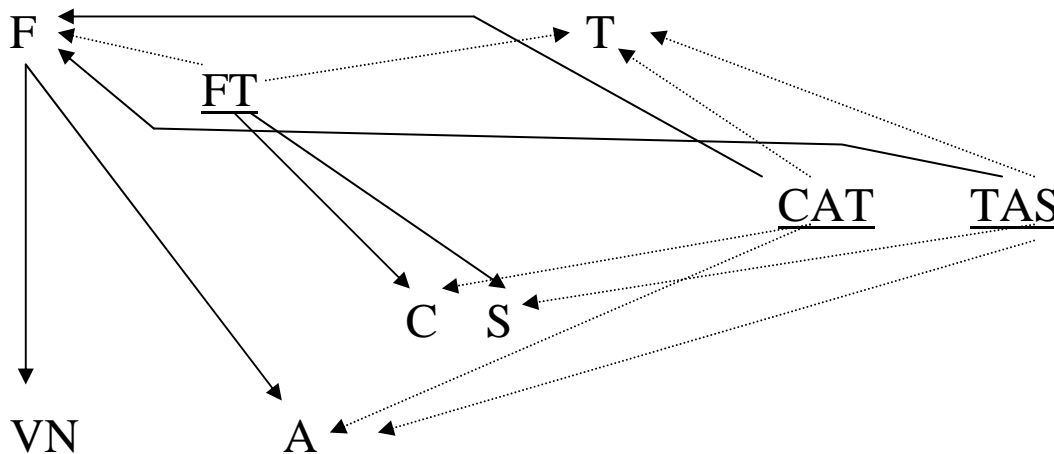
„Vergessene“ fd: $CAT \rightarrow F$
 $TAS \rightarrow F$

weder minimal,
noch abhängigkeittreu
(schlauer: FVN, FTSC, CATF, TASF)

2. Synthesealgorithmen

1. Entfernung überflüssiger fd's aus der gegebenen fd-Menge
Entfernung überflüssiger Attribute
2. Zusammenfassung der fd's zu Klassen mit gleichen (bzw. äquivalente) linken Seiten
Pro Klasse wird ein Schema angelegt
3. Hat kein Schema einen minimalen Schlüssel des ursprünglichen, dann wird ein Schema mit einem minimalen Schlüssel zusätzlich hinzugefügt.

Bsp: $CAT \rightarrow F$, $F \rightarrow VNA$, $TAS \rightarrow F$ $\{F,T\}, \{T,A,S\}, \{C,A,T\}$ sind minimale Schl.
 $FT \rightarrow CS$



Damit Schema:

FVNA, FTSC, CATF, TASF

In 3. Normalform, verbundtreu,
abhängigkeitstreu, vollständig,
aber nicht BCNF

(3. Schema, $F \rightarrow A$, CAT, TF)

(4. Schema, $F \rightarrow A$, TF, TAS)

3. Mehrwertige Abhängigkeiten $X \twoheadrightarrow Y, X \twoheadrightarrow Y' \mid Z$ $Y' = Y \setminus X, Z = \text{comp}(R) \setminus X \setminus Y'$

Unterschiedliche äquivalente Definitionen

a. Komponenten lassen sich strukturieren in X, Y, Z , wobei Y und Z unabhängig voneinander sind bei Berücksichtigung von X

b. Die Existenz von Tupel bedingt die Existenz weiterer Tupel.

$$\forall t, t' \in C(R) (t[X] = t'[Y'] \Rightarrow (\exists t'' \in C(R) : t''[X \cup Y'] = t[X \cup Y'] \wedge t''[X \cup Z] = t'[X \cup Z]))$$

c. Die Werte zu Y lassen sich nur aus X -Werten berechnen.

$$\forall x \in C(R)[X] \forall z \in C(R)[Z] : (\sigma_{X=x \wedge Z=z}(C(R)))[Y] = (\sigma_{X=x}(C(R)))[Y] \quad \Pi_V(C) =: C[V]$$

Bsp.: ISBN \twoheadrightarrow Autoren, Verlag, Titel ISBN \twoheadrightarrow Version, Jahr
 ISBN \twoheadrightarrow Stichwort

Buchexemplar						
ISBN	X			Buchversion		Stichwort
	Autoren	Verlag	Titel	Version	Jahr	
3-92.	Heuer, Saake	Thomson	Datenbanken	1.	1995	Datenbanken
				2.	1998	DB-Sprachen

d. Zerlegungseigenschaft

$$C(R) = ((C(R))[X \cup Y']) * ((C(R))[X \cup Z])$$

Klasse läßt sich durch ihre Projektionen verbundtreu darstellen
 pro X-Wert bilden die Projektionen auf Y' bzw. Z das kartesische Produkt

Axiomatisierung von mehrwertigen und funktionalen Abhängigkeiten

Axiome

$$X \cup Y \rightarrow Y$$

Regeln

$$\frac{X \rightarrow Y, Y \rightarrow Z}{X \rightarrow Z}$$

$$X \rightarrow Z$$

$$\frac{X \rightarrow \rightarrow Y, Y \rightarrow \rightarrow Z}{X \rightarrow \rightarrow Z \setminus Y}$$

$$X \rightarrow \rightarrow Y$$

$$X \rightarrow \rightarrow Y$$

$$\frac{X \rightarrow \rightarrow \text{comp}(R) \setminus X \setminus Y}{X \rightarrow Y}$$

$$X \rightarrow Y$$

$$X \rightarrow \rightarrow Y$$

$$\frac{W \subseteq Y, Y \cap Z = \emptyset \quad X \rightarrow \rightarrow Y, Z \rightarrow W}{X \rightarrow W}$$

$$X \rightarrow W$$

$$\frac{X \rightarrow \rightarrow Y, X \cup Y \rightarrow Z}{X \rightarrow Z \setminus Y}$$

$$X \rightarrow Z \setminus Y$$

vollständig und korrekt

Weitere Regeln z.B.

$$\frac{X \rightarrow \rightarrow Y}{X \cup W \cup Z \rightarrow \rightarrow Y \cup Z}$$

Verallgemeinerung auf n Komponenten (hierarchische, Verbundabhängigkeiten)

4. Inklusions- und Exklusionsabhängigkeiten

Inklusionsabhängigkeit $R_1[X] \subseteq R_2[Y]$ für Folgen X,Y von
Komponenten der Typen R_1, R_2
gilt in Klassen, falls $C(R_1)[X] \subseteq C(R_2)[Y]$

Exklusionsabhängigkeit $R_1[X] \parallel R_2[Y]$ für Folgen X,Y von
Komponenten der Typen R_1, R_2
gilt in Klassen, falls $C(R_1)[X] \cap C(R_2)[Y] = \emptyset$

Axiomatisierbar sind Inklusions- und Exklusionsabhängigkeiten zusammen,
nicht aber z.B. funktionale Abhängigkeiten und Inklusionsabhängigkeiten..

Inklusionsabhängigkeiten können über Werte leicht gepflegt werden.
Exklusionsabhängigkeiten erfordern für die Pflege extra Programme (Trigger
bzw. stored procedures)

Vorsicht: Dekomposition erzeugt für Verbundtreue auch
paarige Inklusionsabhängigkeiten!

5. Lokale Integritätsbedingungen (definiert für einen Typ)

Gegeben seien Typ $R = (\text{compon}(R), \text{keys}(R), \Sigma)$
und die Menge aller möglichen Instanzen mit der Struktur von R
 $\text{Inst}(R)$.

Funktion $\sigma : \text{Inst}(R) \rightarrow \{\text{false}, \text{true}\}$ heißt
lokale Integritätsbedingung.

Klasse $C(R)$ erfüllt σ , falls für $C(R)$ $\sigma(R) = \text{true}$ gilt.

6. Graphbedingungen

Gegeben seien Typenmenge M zusammenhängender Typen des Schemas S
und die Menge aller möglichen Instanzen $\text{Inst}(M)$ über M .

Funktion $\sigma : \text{Inst}(M) \rightarrow \{\text{false}, \text{true}\}$ heißt
Graphbedingung.

Klassenmenge $C(M)$ erfüllt σ , falls für $C(M)$ $\sigma(R) = \text{true}$ gilt.

Spezielle Form: **Pfadbedingungen, globale Integritätsbedingungen.**

Prinzipien der Datenbankmodellierung

Entwicklung am **Beispiel**

einfachere Darstellung und Erkennung der Semantik

Erprobung der Funktionalität

Demos

Entwicklung auf unterschiedlicher **Abstraktionsstufe**

1. Allgemeine Idee (Lastenheft); 2. Geschäftsprozesse (Pflichtenheft, Kosten);

3. Aktionen; 4. Konzeptionelles Modell; 5. Implementationsmodell

Codesign von Struktur, Funktionalität und Interaktion

Interaktion impliziert Funktionalität und Sichten, die benötigt werden

Integration der Modellierung in den Datenbank-**Lebenszyklus**

jede Veränderung der Anwendung zieht Veränderung des Modelles nach sich

Unterscheidung von **Phasen** des Betriebs

Vorbereitungsphase, Einfahrphase, Normalbetrieb, Wartung, Archivierung,

Migration bzw. Zurückfahren

daraus resultierend unterschiedliche Eigenschaften der Daten,

verschiedene Sichten, unterschiedliche Unterstützung,

omaisichere Interaktion

Statische Modellierung ist nur ein Teil der Modellierung

Ziel der Modellierung: System mit optimalen Verhalten

damit: **Modellierung des Verhaltens**

mit

Modellierung der **Funktionen**

Aufgaben, Geschäftsprozesse, Handlungen, Prozesse,
Programme

Modellierung der **Interaktion**

Motive, Story, Szenarien, Drehbuch, Inszenierung

Modellierung der **dynamischen Integritätsbedingungen**

Transitionsbedingungen

temporale Bedingungen

Aktionslogiken

Modellierung der Benutzung, Benutzer(innen) (**Akteure**)

mit ihren Rollen, Rechten, Obligationen, Aufgaben

und Einschränkungen, sowie der Organisationsstruktur

Daneben: Anwendungen sollten selbst vorher analysiert werden auf

Erfordernisse für Reorganisation

Das ‚PantaRhei‘ Modell (*WWW-Workflows*)

Agenten: Name

Aktivität: Kopf

Deklaration

Prozedur

Postcondition

Beschreibung

MaxTime

FormEigensch

Kostenfunktion

Rolle: Zuweisung einer Aktivität

Steuerfluß

Sequenz

Alternative

Schleife

Parallelität

Transaktionsunterstützung

TA-Steueranweisung

Vitalität

Kompensation

Ausführungsmodell

AndJoin, OrJoin,

XorJoin

Workflow: Kopf Name

Argumente (i, o, i/o)

Deklaration

Prozeßspez

Besitzer

Gegenstand

Beschreibung

MaxExecTime

FailureAktion

Daten

Typ, Wertebereich

Körper

Wer

Welche Aktion

Wann

Welche Daten

Interprozeß-Kommunikation

Sende Form an Server

Remote-Ausführung

Modellierung von Dialogen

Software-Ergonomie: **menschengerechte** und **aufgabengerechte** Gestaltung durch

Arbeitsstrukturierung (Aufgabenverteilung, -modell, -angemessenheit)

Anwendungssoftware-**Anpassung** (Funktionen, Leistung, Flexibilität)

E/A-Strukturierung (keine Wiederholung, keine Nach- und Vorarbeiten)

Arbeitsoberfläche (Gestaltungsregelwerke, style guides, look-and-feel)

unter **Berücksichtigung**

von **Benutzergruppen** (Anfänger, Gelegenheitsbenutzer, Experten)(know the user)

ihren **Kompetenzen** (Handlungskompetenz, Lernen, Aufgabenbereiche)

Handlungsflexibilität (Aufgabenmodifikation)

Aufgabenangemessenheit (prinzipielle Durchführbarkeit, Effizienz, Qualität)

Interaktion (Zwischenablage, dynamischer Datenaustausch, Objektverbindung u. -einbettung)

mit den Zielen

benutzeradäquate Bedienung (besser oo anstatt funktionsorientiert,

Direktmanipulation(pick, drag&drop))

wohlstrukturiert (Hierarchie, Fenster, wenig e Merkschritte)

Dialoggestaltung

Primärdialog

(synchronisierter) Sekundärdialog

Methoden:

- 1. Fenstergestaltung** (Fensterrahmen, Titelbalken, Menübalken, Statusbalken, Arbeitsbereich, Rollbalken, Fensterteiler, Kommandobereich, Piktogramme)
mit unterschiedlichen Fenstertypen (Anwendungsfenster (Titel, Menü, Arbeitsbereich), Unterfenster (Hierarchie der Anwendung, evt. auch Gesamtfunktionalität), Dialogfenster (Arbeitsbegleitung, Sekundärdialog, i.a. kein Menü, Ordnung), Mitteilungsfenster (Fehlermeldungen, Spezialdialog))
- 2. Dialogmodi** (Vollständige Beendigung (systemmodal, anwendungsmodal), parallele Bearbeitung (anwendungssemimodal), hierarchisch oder nicht-modal)
- 3. Objektorientierte Anwendungsbedienung**
je nach Typ des Objektes (pull-down-Menü, pop-up-Menü, ...)
- 4. MDI** (multiple document interface)
- 5. Menüs** (Aktionsmenü (Funktion auslösen, Objektauswahl),
Eigenschaftsmenü (Parameter für Verhalten, Darstellung)
mit Symbolbalken, Kommandoauswahl, Schlüssel)
- 6. Kommandos** mit Parametern, Optionen (!leicht lesbar?)

Bewertungskriterien

VDI 5005, DIN 66234

Kompetenzförderung (konsistente handlungsunterstützende Benutzungsoperationen)
(einheitlich, übersichtlich, nur anwendbare Fkt., Undo/Redo, inkrementelle Rückmeldungen)

Handlungsflexibilität (alternative Benutzungsoperationen, Makros, parallel, nicht-modal)

Aufgabenangemessenheit (Effizienz der Benutzungsoperationen, Minimierung)

Selbstbeschreibungsfähigkeit (Erschließung, Benutzungskontext, - zusammenhang)

Steuerbarkeit (beeinflussbarer Ablauf, Auswahl der Wege, mehrstufiges Redo/Undo)

Erwartungskonformität (Einheitlichkeit, Änderungsmitteilungen, Bestätigung)

Fehlerrobustheit (minimale Korrektur, kein Systemhalt, Alternativen, Fehlerort)

Ableitung der Dialogstruktur aus dem Fachkonzept

_____ Klasse auf Dialogobjekt, Komponenten je nach Wertebereich, -typ, -eigenschaften

Berücksichtigung von style guides und Konventionen

Generalisierung/Spezialisierung (Eigenständigkeit erhalten, Identifikation)

Komponenten-Aggregation (erhalten von Existenzabhängigkeiten)

Beziehungen je nach Richtung, Komplexität

E/A-Gestaltung

1. Berücksichtigung der **menschlichen Informationsverarbeitung**

Visuell (foveales versus peripheres Sehen; Blickfixation, Orientierung, Blinkrate)

Aufmerksamkeitssteuerung (kontrollierte, verteilte Aufmerks.), Gedächtnis, Regeln

2. **Interaktionselemente** Basiselemente(E/A-Feld, Knopf, Liste, Graphik)

Erweiterungselemente (Roll- und Trennbalken)

Gestaltungselemente (Umrandung, Überschrift, Führungstext, Spaltenüberschriften)

Gruppierung der Interaktionselemente (räumliche Nähe, Anordnung, Polarität, Farbe, Helligkeit)

Prinzip der guten Gestalt , (Figur-Grund-)Unterscheidung, Gliederung

3. **Verwendung von Farben (, Audio, ...)** Figur-Grund-Unterscheidung und Gruppierung

Suchen, Auffinden, Identifizieren, Zuordnen

Erkennen und Erinnern

Farbhelligkeit, Farbunterschiede, Kontrast, Konsistenz

4. **Formulare und Tabellen, Graphiken, Erklärungshintergrund**

∑ **Gestaltungskriterien: Kompetenzförderlichkeit, Handlungsflexibilität,
Aufgabenangemessenheit**