Die Bearbeitung aller Übungsaufgaben, die mit diesem Formblatt nachgewiesen wird, ist **Voraussetzung für die Zulassung zur Prüfung**.		**Abgabetermin:** **Mitte Juni** bzw. **vor Weihnachten**
Name (DRUCKSCHRIFT)	Vorname	Matrikel-Nr.
1.		
2.		
3.		
4.		
5.		
6.		
7.		
Übungsthema	1. Korrektur	2. Korrektur
1. Structured Analysis 0		
2. Structured Analysis 1		
3. Structured Analysis 2		
4. Datenschutz		
5. Synthetische Datenmodellierung		
6. Beziehungstypen, Kardinalitäten		
7. Vorläufiges Datenlexikon		
8. Normalisierung		
9. Endgültiges Datenlexikon		
10. UNF-Schlüsselvariation		
11. Analoge Datenmodelle		
12. Teilsichtenintegration [bis Ende Mai]		
13. SQL [ein Monat]		

1. Teil: Datenmodellierung
Bibliotheksverwaltungssystem: konzeptuelle Datenmodellierung (Informationsmodell).
Minimalforderung: Beschreibung der Entitätstypen
Benutzergruppen, Benutzer, Bücher (Titel, Exemplare),
Ausleihvorgänge mit Vormerkungen und Verlängerungen,
Statistik-Grunddaten (anonymisierte Ausleihvorgänge)

Zur DB-Tabelle „Statistik-Grunddaten / Anonymisierte Ausleihvorgänge“: Eine Bibliothek darf (personalisierte) Ausleihvorgänge aus Datenschutzgründen nicht beliebig lang über das Rückgabedatum hinaus speichern, braucht aber statistische Auswertungen, die ein paar Jahre in die Vergangenheit zurückreichen. Deshalb kopiert man in gewissen Zeitabständen (bspw. alle halbe Jahre) die älteren Datensätze (die älter als ein halbes Jahr sind) aus der DB-Tabelle Ausleihvorgänge in anonymisierter Form in die DB-Tabelle Anonymisierte Ausleihvorgänge. Die ursprünglichen Datensätze in der DB-Tabelle Ausleihvorgänge werden gelöscht.

2. Teil: SQL
SQL-Übungsaufgaben für das Fahrradfachgeschäft in Edwin Schicker,
Datenbanken und SQL, Stuttgart: Teubner 1996, Abschnitte 3.4, 4 und 6.
Eine Oracle-Demo-Version findet sich in Balzert, Softwaretechnik.

Diese beiden Anwendungsbeispiele sind als Anregung gedacht. Natürlich können Sie nach Absprache mit mir auch jedes andere geeignete Beispiel verwenden.
Auch ein durchgängiges Beispiel für beide Teile ist möglich.

Alle Übungsaufgaben, insbesondere Graphiken, sind ausführlich zu kommentieren.

1./2./3. Überblick über eine Bibliotheksverwaltung mit kommentierten Informationsflussplänen in SA (Structured Analysis)

Die Übungsaufgaben 1–3 dienen dazu, dass Sie sich in einen Anwendungsbereich tiefer einarbeiten, für den Sie später Datenmodelle entwickeln sollen. Das kann die vorgeschlagene Bibliotheksverwaltung sein oder eine Einkaufsverwaltung oder irgendein anderer geeigneter Anwendungsbereich. An Vorkenntnissen brauchen Sie nur Structured Analysis, die Sie aus Software Engineering kennen. Denken Sie an die damals gelernten Konventionen, insbesondere Konsistenz zwischen verschiedenen Abstraktionsebenen!
Bitte die drei Aufgaben nicht einzeln, sondern als "Paket" abgeben.

1. Abstraktionsebene 0 (Kontextdiagramm, Systemgrenzen, Systemumgebung):
   Bibliotheksverwaltung funktional undifferenziert (monolithisch)
   mit externen Schnittstellen und den zugehörigen Informationsflüssen
   (ohne Material- und Geldflüsse)

2. Abstraktionsebene 1:
   Analoga zu betrieblichen Hauptfunktionsbereichen als Teilfunktionen:
   Vertrieb, Einkauf, Rechnungswesen, Personalwirtschaft, Unternehmensführung etc.;
   Informationsflüsse zwischen Funktionen nicht direkt, sondern über Speicher
   (unternehmensweite Datenbank);
   alle Informationsflüsse und externen Schnittstellen aus Ebene 0 übernehmen

3. Abstraktionsebene 2:
   Verfeinerung des Funktionsbereichs Vertrieb (Ausleihverwaltung)
   in 5-9 Teilfunktionen; Schnittstellen aus Ebene 1 übernehmen;
   Funktionsnamen aus Organisationssicht: Substantiv + Verb (Akkusativ-Objekt + Prädikat)
   Genau auf Konsistenz zwischen den drei Abstraktionsebenen achten

4. Datenschutz: Für diese Übungsaufgabe müssen Sie sich in die Datenschutz-Grundverordnung DSGVO der EU einarbeiten. Dazu dient der Foliensatz „Datenschutz und Berufsethik“ auf meiner Homepage. Alternativ dazu können Sie auch die BfDI-Info 1 oder 6 verwenden.

Die Beantwortung der folgenden Fragen soll sich auf den Anwendungsbereich in der Übungsaufgabe 1 beziehen.

1. Welche personenbezogenen Daten verwaltet der Anwendungsbereich?
2. zeitliche Begrenzung der Datenspeicherung
3. Anonymisierung, Statistik-Intervalle etc.
4. verbotene Auswertungen, Missbrauchsmöglichkeiten
5. Zugriffsschutz
6. Unterscheidung privater vs. institutioneller Kunde (bzw. Bibliotheksbenutzer)

Die Übungsaufgaben 5–12 beziehen sich auf Datenmodellierung. Wählen Sie einen Anwendungsbereich als Grundlage für diese Übungsaufgaben.
Ausgangspunkt sollte eigentlich der Anwendungsbereich in Übungsaufgabe 3 sein, Sie können sich aber auch für einen anderen entscheiden.
Das kann bspw. die Bibliotheksverwaltung sein, die auf Seite 2 oben beschrieben ist - mit all den dort genannten Tabellen.
Sie können aber auch jeden beliebigen anderen Anwendungsbereich wählen (z. B. Videoverleih etc.), der 5-7 Tabellen umfasst.

Bei den Übungsaufgaben 5–7 geht es um synthetische Datenmodellierung, also intuitiv Tabellen finden und dann dazu Attribute festlegen.

5. Synthetische Datenmodellierung: Liste der Tabellen des ausgewählten Anwendungsbereichs mit (informationsrelevanten) Basisattributen und Schlüsselkandidaten.
   Implementationsrelevante abgeleitete Attribute nicht verwenden.
   Die Minimalforderung hinsichtlich der Entitätstypen beachten (Seite 2 oben).
   Spätestes und tatsächliches Rückgabedatum unterscheiden.
   Anonymisierungszeitpunkt der Ausleihvorgänge angeben.

Wenn Sie mit Beziehungen zwischen Tabellen schon vertraut sind, können Sie die Übungsaufgabe 6 gleich mit bearbeiten.

6. Beziehungstypen / Kardinalitäten zwischen den in Übungsaufgabe 5 gefundenen Tabellen:
   vorläufiges Datenstrukturdiagramm mit 1:n-, n:m-Beziehungen

Halten Sie sich dabei an folgende Konventionen:

– 1:n-Beziehungen werden mit einem Pfeil (oder auch Krähenfuß) in n-Richtung bezeichnet.

– An jeden Beziehungspfeil wird der entsprechende Fremdschlüssel geschrieben.
  Wenn ein Modellierungstool das nicht kann, ergänzen Sie die Angaben von Hand.

– Markieren Sie Primärschlüssel (PK) durch Unterstreichen, Fremdschlüssel (FK) durch ein Sternchen *

– Wenn Sie die Notationsweise „FK“ für Fremdschlüssel verwenden:
  Unterscheiden Sie verschiedene FK in ein und derselben Tabelle mit „FK1“, „FK2“!!!
  Die Nummer zählt verschiedene FK. Das ist wichtig für zusammengesetzte FK!
  Markieren Sie die Attribute eines zusammengesetzten FK als „FKn“ mit der gleichen Nummer n.

7. Vorläufiges konzeptuelles Datenlexikon (Liste aller Attribute Ihres Datenmodells):
   Attribute, die im Datenmodell zweimal vorkommen – als Primärschlüssel und als Fremdschlüssel (oder als Teile davon) –, werden nur einmal aufgeführt.
   Zweispaltige Tabelle:
   – ein sprechender Attributname
   – ein aussagekräftiger semantischer Kommentar, wenn erforderlich
   Dient als Stoffsammlung zur Normalisierung (UNF für Aufgabe 8),
   deshalb müssen die Attributnamen im gesamten Datenmodell eindeutig sein.
   (ausführlich in Aufgabe 9).

8. Analytische Datenmodellierung – Normalisierung bis zur 3NF:
   Als UNF dient die Attribut-Liste in ÜA 7 mit einem geeigneten UNF-Primärschlüssel.
   (Jeder Primärschlüssel einer Tabelle aus 5./6. ist als UNF-Primärschlüssel geeignet.)
   Ziel ist ein Modell, das dem Modell in ÜA 6 sehr ähnlich ist.
   Zu zeichnen sind Datenstrukturdiagramme mit Fremdschlüsseln für jeden einzelnen Normalisierungsschritt und ggf. eine manuelle Optimierung.
   Dazu können Sie eine Excel-Tabelle verwenden.

– Sie müssen Ihr Modell aus Aufgabe 6 daneben legen und schrittweise darauf lossteuern!

– Es ist grundsätzlich jeder Primärschlüssel, der bei den Tabellen in Aufgabe 5/6 erscheint, als UNF-Schlüssel geeignet. Jeder geeignete UNF-Schlüssel führt zur gleichen 3NF; 1NF und 2NF sind aber je nach UNF-Schlüssel unterschiedlich (vgl. Aufgabe 10).

– Es dürfen keine Attribute mit gleichem Bezeichner vorkommen. Die Attributnamen müssen für das gesamte Datenmodell (DB-weit) eindeutig sein.

– Es dürfen ggü. der UNF keine neuen Attribute dazukommen!

– Stellen Sie jeden einzelnen Schritt auf dem Weg zur 3NF dar; darüber hinaus ggf. einen zweiten Durchlauf durch den Kalkül und eine manuelle Optimierung.
  Es genügt, wenn Sie jeweils nur die Tabellen nennen, in denen es Veränderungen gibt.

– Geschachtelte Abhängigkeiten werden in mehreren Schritten (für jede Schachtelungsebene einer) ausgeklammert, parallele in einem.

– Es können immer nur solche neuen Tabellen entstehen, die mit der jeweiligen Ausgangstabelle direkt verbunden sind!

9. Datenlexikon und Tabellenmatrix (dazu gibt es keine Theorie-Folien)
Attribute, die im Datenmodell zweimal vorkommen – als Primärschlüssel und als Fremdschlüssel (oder als Teile davon) –, werden nur einmal aufgeführt.

9.1 Endgültiges konzeptuelles Datenlexikon in 3NF
Diese Teilaufgabe ist die Fortsetzung und Erweiterung von Aufgabe 7. Das ist rein handwerkliche Arbeit. Erstellen Sie eine Excel-Tabelle.
Für jedes Attribut: IT-Name (mit Tabellenkürzel; mnemotechnische Abkürzung, die man sich leicht merken kann), Langname, Kommentar,
Datentyp, Länge, Wertebereich, Voreinstellung, Eindeutigkeit
Für jedes Attribut: Ist es Primär-, Fremdschlüssel? Zu welcher Tabelle gehört es?

Bei der Korrektur achte ich besonders auf den Datentyp der Primärschlüssel. Es soll keine unmotivierten numerischen Primärschlüssel geben.

9.2 Tabellenmatrix
Diese Teilaufgabe ist spannender. Erstellen Sie wiederum eine Excel-Tabelle:
vertikal (Zeilen): alle Attribute des Datenlexikons (ohne Redundanzen)
„Ohne Redundanzen“ bedeutet, dass Attribute, die eine Primärschlüssel-Fremdschlüssel-Beziehung bilden, nur einmal genannt werden.
horizontal (Spalten): alle DB-Tabellen
Für jedes Attribut ist zu markieren, ob es in in der jeweiligen DB-Tabelle Primär-, Fremdschlüssel oder einfaches Attribut ist.

Mithilfe der Tabellenmatrix kann man erkennen,
– ob jede DB-Tabelle einen Primärschlüssel hat
– ob es zu jedem Fremdschlüssel einen Primärschlüssel gibt

Ich achte also bei der Korrektur auf zwei Dinge (die Sie selbst vorher überprüfen sollten):
– In jeder Spalte: Gibt es zu dieser DB-Tabelle einen Primärschlüssel?
– In jeder Zeile, wo ein Fremdschlüssel vorkommt: Gibt es dazu einen Primärschlüssel?

10. Normalisierungskalkül mit UNF-Schlüsselvariation (nächste Seite; keine Theorie-Folien):
    1. UNF-Schlüssel: Artikelnummer
    2. UNF-Schlüssel: Kundennummer
    3. UNF-Schlüssel: Auftragsnummer
    4. UNF-Schlüssel: Auftragsnummer + Artikelnummer

Unabhängig vom UNF-Schlüssel (vier Möglichkeiten sind angegeben) entsteht zwar immer die gleiche 3NF (vier Tabellen). Aber die Zwischenschritte 1NF und 2NF sind verschieden!

Zeichnen Sie für jede (!) der vier Normalisierungsvarianten das (gleiche!) 3NF-Datenstrukturdiagramm und vermerken Sie unter jeder Tabelle, in welchem Normalisierungsschritt sie entsteht.

Fassen Sie Ihre Ergebnisse in einer Excel-Tabelle zusammen:
– Spalten: UNF-Schlüsselvariationen
– Zeilen: DB-Tabellen
Geben Sie in den Tabellenzellen an, welche Tabelle in welchem Normalisierungsschritt entsteht.

Ziel ist, dass Sie mit UNF-Schlüsselvariation umzugehen lernen, d. h. also ausgehend von verschiedenen geeigneten UNF-Schlüsseln die – gleiche – 3NF schrittweise korrekt ableiten können.

Zu dieser Aufgabe ist noch Folgendes zu ergänzen:

– Die Attributnamen sind in dieser Aufgabe etwas antiquiert formuliert. Sie können bessere Bezeichner verwenden.

– Sie haben es mit einer Standard-Auftragsverwaltung zu tun. Also: zu einem Kunden kann es mehrere Aufträge geben, zu einem Auftrag mehrere Auftragspositionen; und ein Artikel kann in mehreren Auftragspositionen vorkommen.

– Das Attribut Kundenpriorität ist nur kundenabhängig und gilt für jeden Auftrag des Kunden.

– Das Attribut Sonderfertigung kennzeichnet Aufträge, in denen eine Sonderfertigung vorkommt.

Übungsaufgabe 10

Aufgabe: Erstellung eines logischen Datenmodells

Ausgangsstruktur des konventionellen Systems:

	Artikel-Nr.	Artikel-Bez.	Bestand	
Wurzel	ANR	ABESCHR	BEST	Stufe 0

Auftrags-Nr.	Sonderfertigung (JA/NEIN)	Kunden-Nummer	Kundenpriorität	Anzahl	
AUFNR	S	KNR	KPR	ANZ	Stufe 1

Ein Betrieb bietet Artikel an, die eine Artikelnr. (ANR) und eine Beschreibung besitzen. Es ist ein Bestand vorhanden und mehrere Aufträge sind eingegangen (Wiederholungsgruppe). Die Aufträge haben eine Nummer, eine Angabe, ob Sonderanfertigung vorliegt oder nicht, die Kundennummer des Auftrages (KNR kann redundant existieren), die Kundenpriorität, d.h. ob ein Kunde bevorzugt beliefert wird, und die Bestellmenge. Definieren Sie ein unnormalisiertes Datenmodell und leiten Sie davon ausgehend über die FNF und SNF die TNF her.

Kundenpriorität: nicht auftragsabhängig
Sonderfertigung: auftragsabhängig

11. Referenzmodelle und Modellanalogien
Entwerfen Sie und stellen Sie vergleichend gegenüber:
– 1. ein komplexes generisches Referenzmodell (Oberbegriffe)
– 2.–8. die (teil-)analogen Datenmodelle der folgenden Anwendungsbereiche

2. Ausleihverwaltung einer Bibliothek
– Ausleihaufträge haben immer nur eine Ausleihposition, so dass man die beiden Tabellen in einer zusammenfasst.
– Eine Ausleihposition bezieht sich immer auf ein ganz bestimmtes, individuell identifizierbares Buchexemplar mit einer Inventarisierungsnummer, auf einen phys. Vertragsgegenstand, auf ein konkretes Exemplar eines log. Vertragsgegenstands, eines bestimmten Buchtitels, der alle Buchexemplare mit dem gleichen Titel und Autor zusammenfasst.
– Eine Reservierung bezieht sich immer auf einen Buchtitel (log. Vertragsgegenstand) und nicht auf ein Buchexemplar (phys. Vertragsgegenstand). Alle Buchexemplare eines Buchtitels sind gleich, daher ist es egal, welches Exemplar ein Bibliotheksbenutzer bekommt.

Zum Vergleich: die Entitätstypen Mitgliedschaftsarten, Personen/Mitglieder, Leihvorgänge, Leihgeräte bei einem Verein (vgl. Datenmodelle auf den entsprechenden Vorlesungsfolien)
Zum Vergleich: die Entitätstypen Fahrzeugklassen, Fahrzeuge, Reservierung, Mitglieder, Mitgliedschaftsarten bei Carsharing

3. (Kunden-)Auftragsverwaltung / Rechnungsverwaltung
Bei einer Auftragsverwaltung unterscheiden Sie nicht zwischen phys. und log. Artikeln. Denken Sie etwa an einen Kassenbon in einem Supermarkt. Wenn Sie sich irgendwo eine Schachtel Schrauben einer bestimmten Größe kaufen, dann sehen alle diese Schachteln gleich aus, sind nicht durch einen individuellen Code unterscheidbar und sind nicht individuell identifizierbar. Damit liegen keine physischen Vertragsgegenstände vor, auch wenn sie eine Ware konkret in der Hand halten können.
Denkbar ist ein phys. Vertragsgegenstand beim Kauf eines bestimmten Gebrauchtwagens.

Zum Vergleich: die Entitätstypen Mitgliedergruppen, Mitglieder, Einzelrechnungen, Einzelrechnungspositionen bei Carsharing
Zum Vergleich: die Entitätstypen Personentypen, Personen, Mieter, Mietverträge, Wohnungen, Objekte einer Vermietergenossenschaft (Vorsicht: hier gibt es an einer wichtigen Stelle keine Analogie!)

4. Einkaufsverwaltung (Bestellwesen, Lieferantenverwaltung)
Zum Vergleich: die Entitätstypen Lieferanten, Bestellungen, Bestellpositionen, Artikel, Warengruppen bei einem Comicladen

5. Lieferanten-Angebotsverwaltung
Bei Lieferantenangeboten erfolgt keine Unterscheidung zwischen phys und log Erzeugnissen.

6. Interne Projektverwaltung
(Abteilung, Mitarbeiter, Mitarbeiter-Projekt-Zuordnung, Projekte, Projektgruppen/-typen)
Es gibt keine verschiedenen physischen Repräsentationen eines logischen Projekts. Log. und phys. Projekte fallen zusammen.

7. Externe Projektverwaltung
(Auftraggeber, Projekt, Projektposition, Dienstleistungsart)

8. zusätzlich der Anwendungsbereich, den Sie in Aufgabe 8 modelliert haben (falls er nicht schon unter 1–6 vorkommt)

Sie lösen diese Aufgabe am besten mit einer Excel-Tabelle:
– Zeilen: Anwendungsbereiche
– Spalten: hier schreiben Sie die analogen Entitätstypen der Anwendungsbereiche untereinander, dazwischen die Beziehungspfeile.

WICHTIG: Es braucht nicht zu jedem Entitätstyp eines Anwendungsbereichs in jedem anderen Anwendungsbereich einen analogen Entitätstyp zu geben!!!

12. Teilsichtennormalisierung und –integration (nächste Seite; keine Theorie-Folien)

12.1 Teilsichtennormalisierung ohne Normalisierungskalkül
– Erstellen Sie für jede der fünf Teilsichten ein kleines Datenmodell (Datenstrukturdiagramm)
– Lassen Sie hierzu nicht den gesamten Normalisierungskalkül ablaufen, sondern:
– Prüfen Sie die drei Normalisierungsbedingungen im Nachhinein.
– Vergessen Sie die Überschriften „Suchbegriff“ und „Ausgaben“ in der Vorlage.
– Wählen Sie sprechende Attributnamen nach Ihrem Geschmack.

1. Außendienst
– statt „Provision“ „Provisionssatz“
– ergibt zwei oder auch drei Tabellen

2. Auftragsverwaltung
– ergibt vier Tabellen

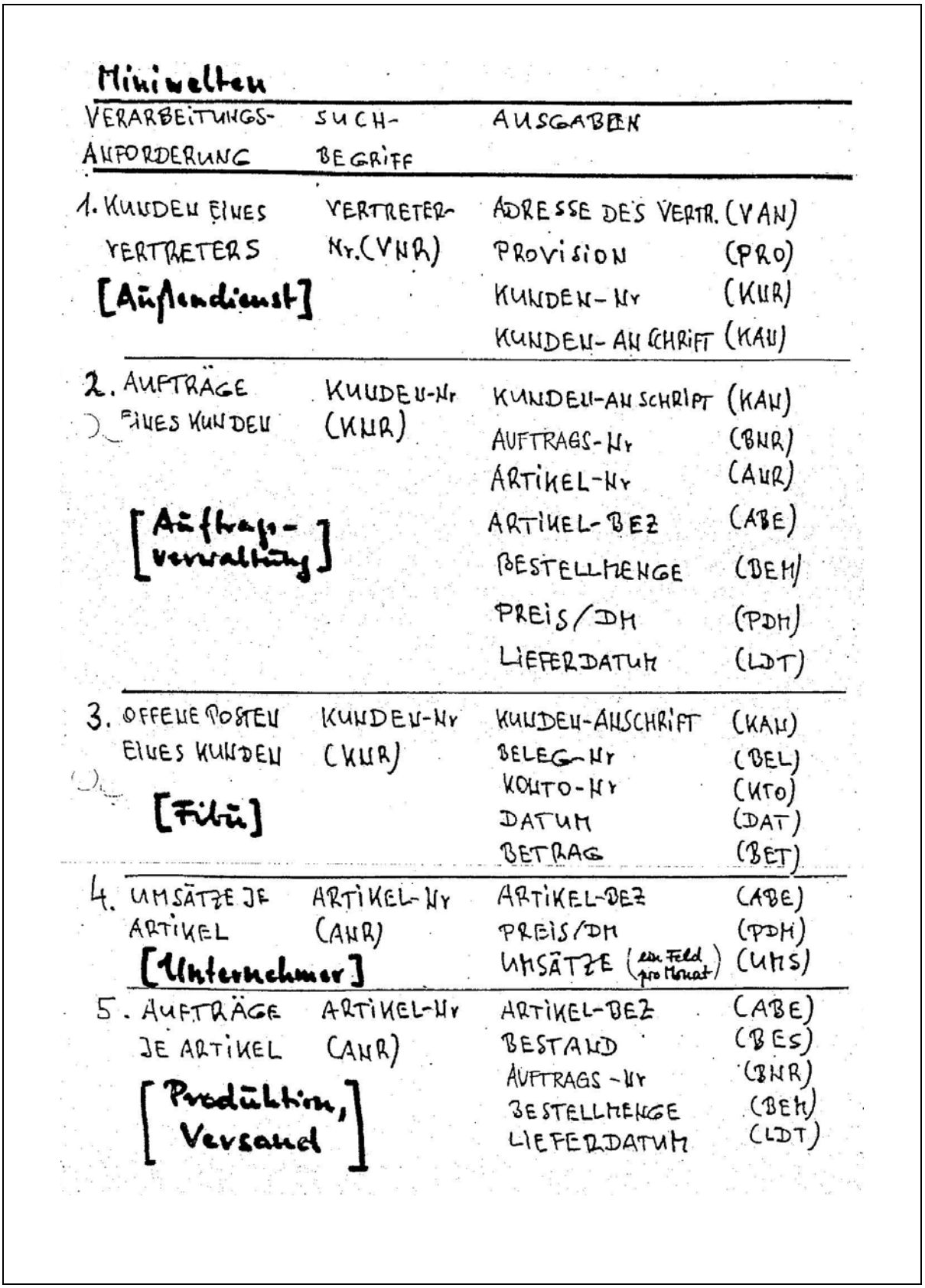
3. Fibu
– Beleg_Nr ist die Rechnungsnr einer unbezahlten Rechnung (eines offenen Postens)
– Konto_Nr ist das Fibu-Erlös-Konto, dem der offene Posten zugeordnet wird
– ergibt zwei oder auch drei Tabellen

4. Unternehmer-Statistik
– ergibt zwei Tabellen

5. Produktion
– ergibt drei Tabellen

12.2 Teilsichtenintegration: Überlagerung der Teilmodelle aus 12.1
Fassen Sie die fünf Teilmodelle zu einem Gesamtmodell zusammen. Das Ergebnis kennen Sie zum Großteil aus der Referenzmodellierung.
Teildatenmodelle dürfen Sie nur dann zu einem größeren Modell zusammensetzen, wenn sie mindestens eine Tabelle gemeinsam haben. Also keine Einführung von Fremdschlüsseln und Beziehungen durch die Hintertür erst bei der Modellintegration!

Übungsaufgabe 12



Minuwelten

VERARBEITUNGS-ANFORDERUNG	SUCH-BEGRIFF	AUSGABEN	
1. KUNDEN EINES VERTRETERS [Außendienst]	VERTRETER-Nr. (VNR)	ADRESSE DES VERTR.	(VAN)
		PROVISION	(PRO)
		KUNDEN-Nr	(KNR)
		KUNDEN-ANSCHRIFT	(KAN)
2. AUFTRÄGE EINES KUNDEN [Auftrags-verwaltung]	KUNDEN-Nr (KNR)	KUNDEN-ANSCHRIFT	(KAN)
		AUFTRAGS-Nr	(BNR)
		ARTIKEL-Nr	(ANR)
		ARTIKEL-BEZ	(ABE)
		BESTELLMENGE	(BEM)
		PREIS/DM	(PDM)
		LIEFERDATUM	(LDT)
3. OFFENE POSTEN EINES KUNDEN [Fibu]	KUNDEN-Nr (KNR)	KUNDEN-ANSCHRIFT	(KAN)
		BELEG-Nr	(BEL)
		KONTO-Nr	(KTO)
		DATUM	(DAT)
		BETRAG	(BET)
4. UMSÄTZE JE ARTIKEL [Unternehmer]	ARTIKEL-Nr (ANR)	ARTIKEL-BEZ	(ABE)
		PREIS/DM	(PDM)
		UMSÄTZE (ein Feld pro Monat)	(UMS)
5. AUFTRÄGE JE ARTIKEL [Produktion, Versand]	ARTIKEL-Nr (ANR)	ARTIKEL-BEZ	(ABE)
		BESTAND	(BES)
		AUFTRAGS-Nr	(BNR)
		BESTELLMENGE	(BEM)
		LIEFERDATUM	(LDT)

13. Für diese Übungsaufgabe müssen Sie sich in den SELECT-Befehl einarbeiten (Selbststudium).
Verwenden Sie als Anwendungsbereich (suchen Sie sich selbst aus) nur ein kleines Datenmodell mit maximal 5 Tabellen.

Abzugeben sind:

– das Datenmodell des verwendeten Anwendungsbereichs

– fünf bis sieben nicht zu einfache, sondern etwas spannendere, komplexere Problemstellungen (denken Sie sich selbst aus).
Es sollen WHERE, JOIN … ON, GROUP mit Aggregatfunktion, HAVING, ORDER, geschachtelter SELECT und paralleler Select (mit UNION, INTERSECT oder MINUS) mindestens einmal vorkommen.

– zu jeder Problemstellung eine Lösung mit einem SELECT-Befehl (ANSI-Standard) unter irgendeinem DBMS oder trocken

– zu jeder Aufgabe ein strukturierter Kommentar
(Bezug zwischen Semantik der Aufgabenstellung und Syntax der Lösung wie im Foliensatz „Ergänzende Folien zu Datenbanken“ beschrieben und an vielen Stellen in „Einführung und Aufgaben zu SQL“)

Die folgenden Aufgaben sind ergänzende Übungen zur Vorbereitung auf die Klausur.

Übungsaufgabe Datenmodellierung (Einzelhändler) (auf den folgenden drei Seiten)

Die Anleitung in dieser uralten Aufgabe ist nicht immer sauber und teilweise sogar schlampig und falsch. Aber genau deswegen belasse ich sie in meinem Übungsprogramm. Um mögliche Fehler zu zeigen!

Zu Nr. 1: „Datenelemente ... objektbezogen im Datenkatalog vorgruppieren“ meint ganz einfach, die Attribute der UNF semantisch vorzustrukturieren, so dass man sich beim Erkennen der Wiederholgruppen für die 1NF leichter tut.

Zu Nr. 2: Bestimmung eines geeigneten UNF-Primärschlüssels

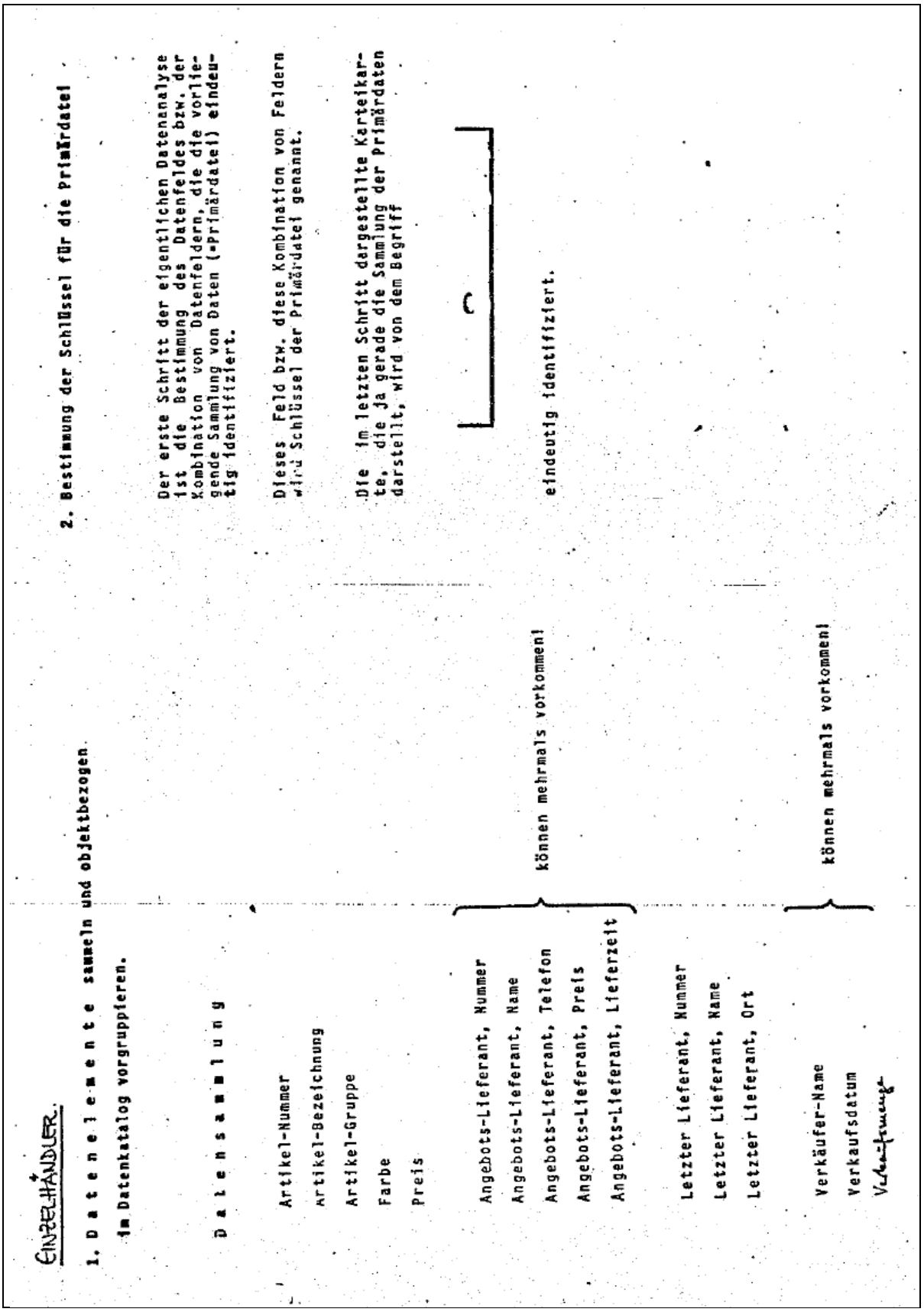
Zu Nr. 3: 1NF
– „Herauslösen von Mehrfachfeldern“ meint Ausklammern von Wiederholgruppen.
– Die Regel zur PK-Festlegung in den neuen Tabellen der 1NF ist in der Anleitung falsch. Siehe meine Folien zur 1NF (Daumenregel und Auftrags_ID)!
– Die in der Aufgabe verwendete Terminologie „Angebot“ muss man hinterfragen. Im Sinne unseres Referenzmodells handelt es sich um den Spezialfall von Angeboten mit nur einer einzigen Angebotsposition.

Zu Nr. 4: 2NF

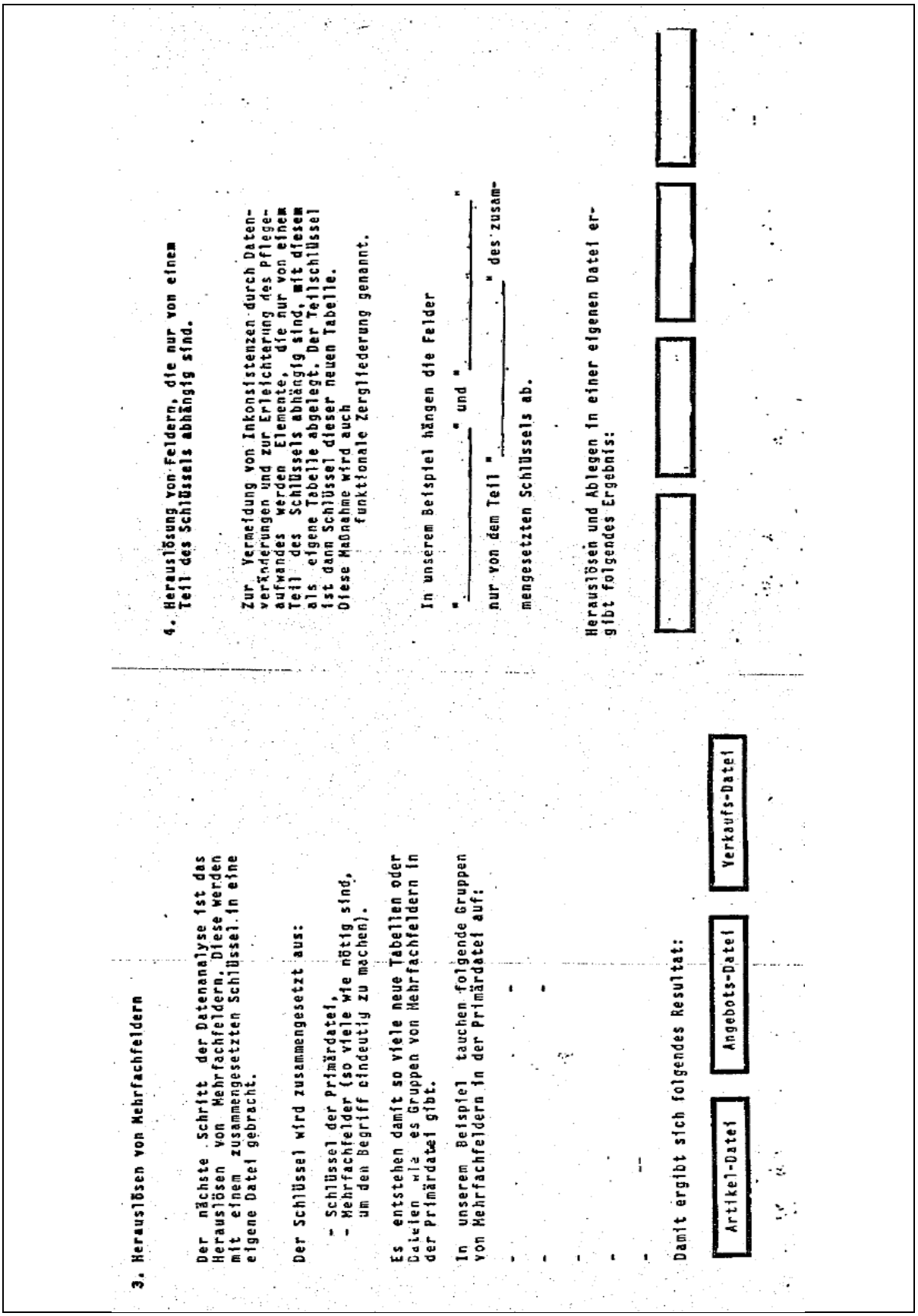
Zu Nr. 5: 3NF
Das Zusammenlegen der Angebots-Lieferanten-Tabelle und der Letzte-Lieferanten-Tabelle zu einer allg. Lieferanten-Tabelle darf bei der mathematisch exakten Vorgehensweise in unserem Kurs erst im Rahmen einer manuellen Optimierung erfolgen.
Beim Zusammenlegen muss nämlich in der Artikel-Tabelle ein Fremdschlüssel erhalten bleiben, der auf den letzten Lieferanten (und nicht Angebots-Lieferanten) verweist. Zu jedem Artikel gibt es ja genau einen letzten Lieferanten.

Zu Nr. 6: Auch der letzte Schritt der Anleitung ist nicht sauber beschrieben. Es werden nämlich neue Attribute eingefügt, was im Lauf des Kalküls verboten ist. Mathematisch exakt – wie wir das machen – hätte man bereits in der UNF folgende Attribute aufführen müssen:
– Art_Gruppen_ID und Art_Gruppen_Bezeichnung
– Art_Farb_ID und Art_Farb_Bezeichnung
– Verkäufer_ID und Verkäufer_Name
Die zugehörigen Tabellen würde man dann bereits in der 2NF (Verkäufer) und 3NF (Art_Farben, Art_Gruppen) erhalten.
Was man daraus lernen sollte: Wenn einem während des Normalisierungskalküls auffällt, dass man Attribute vergessen hat oder dass es nützlich wäre, weitere Attribute einzufügen, dann zurück zur UNF und das Ganze von vorne.

Übungsaufgabe Datenmodellierung (Einzelhändler)



Übungsaufgabe Datenmodellierung (Einzelhändler)



3. Herauslösen von Mehrfachfeldern

Der nächste Schritt der Datenanalyse ist das Herauslösen von Mehrfachfeldern. Diese werden mit einem zusammengesetzten Schlüssel in eine eigene Datei gebracht.

Der Schlüssel wird zusammengesetzt aus:

- Schlüssel der Primärdatei,
- Mehrfachfelder (so viele wie nötig sind, um den Begriff eindeutig zu machen).

Es entstehen damit so viele neue Tabellen oder Dateien wie es Gruppen von Mehrfachfeldern in der Primärdatei gibt.

In unserem Beispiel tauchen folgende Gruppen von Mehrfachfeldern in der Primärdatei auf:

-
-
-
-
-

Damit ergibt sich folgendes Resultat:

Artikel-Datei | Angebots-Datei | Verkaufs-Datei

4. Herauslösung von Feldern, die nur von einem Teil des Schlüssels abhängig sind.

Zur Vermeidung von Inkonsistenzen durch Datenveränderungen und zur Erleichterung des Pflegeaufwandes werden Elemente, die nur von einem Teil des Schlüssels abhängig sind, mit diesem als eigene Tabelle abgelegt. Der Teilschlüssel ist dann Schlüssel dieser neuen Tabelle.
Diese Maßnahme wird auch
funktionale Zergliederung genannt.

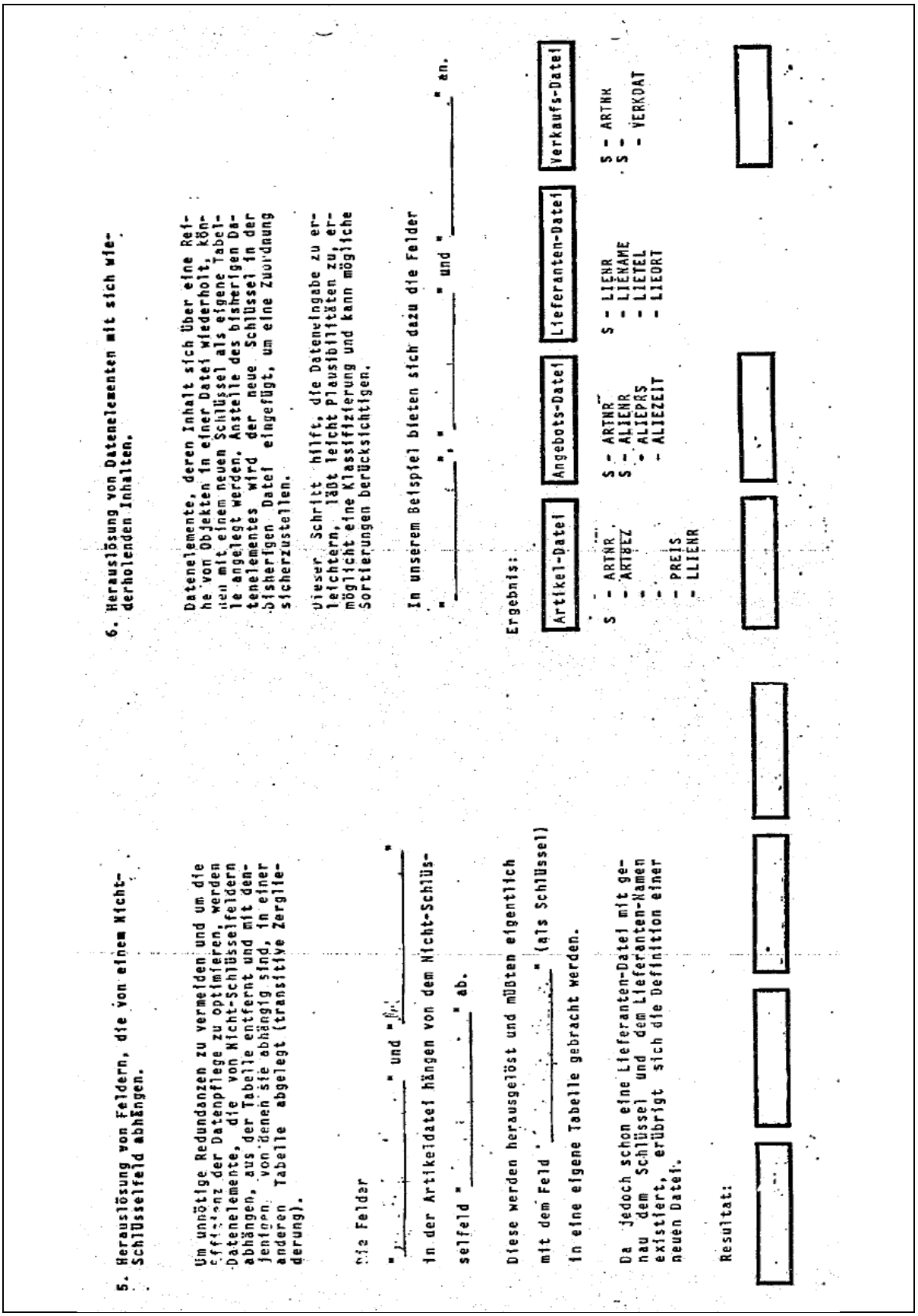
In unserem Beispiel hängen die Felder

"_____" und "_____"

nur von dem Teil "_____" des zusammengesetzten Schlüssels ab.

Herauslösen und Ablegen in einer eigenen Datei ergibt folgendes Ergebnis:

Übungsaufgabe Datenmodellierung (Einzelhändler)



5. Herauslösung von Feldern, die von einem Nicht-Schlüsselfeld abhängen.

Um unnötige Redundanzen zu vermeiden und um die Effizienz der Datenpflege zu optimieren, werden Datenelemente, die von Nicht-Schlüsselfeldern abhängen, aus der Tabelle entfernt und mit denjenigen, von denen sie abhängig sind, in einer anderen Tabelle abgelegt (transitive Zergliederung).

Die Felder

"_____" und "_____"

in der Artikeldatei hängen von dem Nicht-Schlüsselfeld "_____" ab.

Diese werden herausgelöst und müßten eigentlich mit dem Feld "_____" (als Schlüssel) in eine eigene Tabelle gebracht werden.

Da jedoch schon eine Lieferanten-Datei mit genau dem Schlüssel und dem Lieferanten-Namen existiert, erübrigt sich die Definition einer neuen Datei.

Resultat:

6. Herauslösung von Datenelementen mit sich wiederholenden Inhalten.

Datenelemente, deren Inhalt sich über eine Reihe von Objekten in einer Datei wiederholt, können mit einem neuen Schlüssel als eigene Tabelle angelegt werden. Anstelle des bisherigen Datenelementes wird der neue Schlüssel in der bisherigen Datei eingefügt, um eine Zuordnung sicherzustellen.

Dieser Schritt hilft, die Dateneingabe zu erleichtern, läßt leicht Plausibilitäten zu, ermöglicht eine Klassifizierung und kann mögliche Sortierungen berücksichtigen.

In unserem Beispiel bieten sich dazu die Felder

"_____", "_____" und "_____" an.

Ergebnis:

Artikel-Datei
- S - ARTNR
- ARTBEZ
- PREIS
- LLIENR

Angebots-Datei
- S - ARTNR
- S - ALIENR
- ALIEPRS
- ALIEZEIT

Lieferanten-Datei
- S - LIENR
- LIENAME
- LIETEL
- LIEORT

Verkaufs-Datei
- S - ARTNR
- S -
- VERKDAT

Klausuraufgabe Datenmodellierung (Fortbildungsinstitution)

Eine Fortbildungsinstitution will die Verwaltung von Prüfungsteilnehmern mit einer relationalen Datenbank abwickeln.
Es gelten die folgenden Beziehungen:

- Ein Teilnehmer muß mindestens eine, kann mehrere Prüfungen ablegen.
- Eine Firma kann mehrere Teilnehmer anmelden.
- Ein Dozent kann mehrere Prüfungen stellen.
- Eine Prüfung wird von einem Dozenten abgehalten.

a) Führen Sie ausgehend von folgenden Datenelementen (UNF) eine Datenanalyse bis zur 3NF mit dem Normalisierungskalkül durch.
Bezeichnen Sie bei den Tabellen jeder Normalform Primärschlüsse durch Unterstreichen und Fremdschlüssel durch *.
Finden Sie sinnvolle EDV-Namen für die Datenfelder in den 3NF-Tabellen.

Teilnehmer-Nummer
Teilnehmer-Name
Teilnehmer-Anrede
Teilnehmer-Privatanschrift

Prüfungs-Nummer
Prüfungs-Bezeichnung
Prüfungs-Ort
Prüfungs-Datum
Dozenten-Schlüssel
Dozenten-Name
Note
Platzziffer
Gesamtzahl der Prüfungsteilnehmer
} kann mehrmals vorkommen

Firmen-Schlüssel
Firmen-Bezeichnung
Firmen-Anschrift
Firmen-Telefon

b) Skizzieren Sie in Stichworten die Herleitung von 1NF (2 Zeilen), 2NF (1 Zeile) und 3NF (1 Zeile).

c) Stellen Sie die Abhängigkeiten der 3NF-Tabellen in einem Datenstrukturdiagramm als gerichteten Graph dar.

Klausuraufgabe SQL (Ausleihstatistik einer Bibliothek)
– Ein Ausleihvorgang in einer Bibliothek bezieht immer nur auf genau ein Buch.
– Der Stichtag muss gleich „heute“ sein.
Mit einem Stichtag in der Vergangenheit wird die Aufgabe viel schwieriger.

```
CREATE TABLE STUDENT
(S_MATRNR            INTEGER NOT NULL,      Primärschlüssel
 S_STUDNAME          CHAR(30),
 S_STUDPLZ           CHAR(4),
 S_STUDORT           CHAR(30));

CREATE TABLE BUCH
(B_SIGNATUR          CHAR(15),              Primärschlüssel
 B_FACHGEBIET        CHAR(02),
 B_AUTORNAME         CHAR(40),
 B_TITEL             CHAR(65),
 B_ERSCHEINUNGSORT   CHAR(30),
 B_ERSCHEINUNGSJAHR  INTEGER);

CREATE TABLE AUSLEIHVORGANG
(A_MATRNR            INTEGER NOT NULL,
 A_SIGNATUR          CHAR(15),              Primärschlüssel
 A_AUSLEIHDATUM      DATE,
 A_RÜCKGABEDATUM     DATE,
 A_MAHNZAHL          INTEGER);

CREATE TABLE STATISTIK
(STAT_SIGNATUR       CHAR(15),              Primärschlüssel
 STAT_AUSLEIHDATUM   DATE,                  Primärschlüssel
 STAT_STUDPLZ        CHAR(4));
```

Formulieren Sie - ausgehend von den vier oben definierten Basistabellen - SQL-Befehle für die folgenden Auswertungen, die zu einem beliebigen Stichtag angefertigt werden sollen.
Geben Sie an, ob Sie sich auf ANSI-SQL oder eine spezielle Implementation beziehen.

3.1 Anzahl der Ausleihvorgänge, die am Stichtag erfolgt sind.

3.2 Anzahl der Bücher der Bibliothek, die am Stichtag im Besitz von Studenten (= entliehen) sind.

3.3 Anzahl der Studenten, die am Stichtag von der Bibliothek mit mindestens einem Ausleihvorgang bedient worden sind.

3.4 Anzahl der Studenten, die am Stichtag mindestens ein Buch der Bibliothek im Besitz haben.

3.5 Anzahl der Studenten, die am Stichtag kein Buch der Bibliothek im Besitz haben.

3.6 Liste aller Studenten (A_MATRNR, S_STUDNAME, Anzahl), bei denen Mahnungen erfolgt sind und die die angemahnten Bücher noch nicht zurückgebracht haben, aufsteigend sortiert nach der Anzahl der Mahnschreiben je Student.
Hinweis: A_MAHNZAHL enthält die Anzahl der versandten Mahnschreiben je Ausleihvorgang.

Klausuraufgabe SQL (Kunden-Artikel-Statistik einer Auftragsverwaltung)

In einem Sammelabrechnungssystem werden einzelne Lieferpositionen tagesbezogen für die spätere monatliche Fakturierung gespeichert. Die Löschung der Positionszeilen erfolgt nicht schon bei der Rechnungsschreibung, sondern erst nach einer statistischen Auswertung am Kalenderjahresende.

Ein Ausschnitt aus einem solchen System sei durch folgende drei Basistabellen gegeben:

```
CREATE TABLE ARTIKEL
(A_ARTNR       INTEGER NOT NULL,
 A_ARTBEZEICH  CHAR(25));

CREATE TABLE KUNDE
(K_KDNR        INTEGER NOT NULL,
 K_KDNAME      CHAR(30),
 K_KDPLZ       INTEGER,
 K_KDORT       CHAR(30),
 K_KDSTRASSE   CHAR(30));

CREATE TABLE LIEFERPOSITION
(L_KDNR        INTEGER NOT NULL,
 L_ARTNR       INTEGER NOT NULL,
 L_DATUM       DATE,
 L_MENGE       INTEGER);
```

Für die Statistik des Jahres 1990 werden u.a. folgende Übersichten benötigt:

1. Liste aller 1990 verkauften Artikel (Nummer, Bezeichnung) mit der jeweiligen Gesamtumsatzmenge im Jahre 1990 sortiert nach der Artikelnummer

2. alle Artikel (Nummer, Bezeichnung), die 1990 nicht verkauft worden sind, sortiert nach der Artikelnummer

3. alle Kunden (Nummer), die 1990 nichts gekauft haben, sortiert nach der Kundennummer; Anzahl dieser Kunden

Formulieren Sie - ausgehend von den drei oben definierten Basistabellen - SQL-Befehle für diese Auswertungen.
Geben Sie an, ob Sie sich auf ANSI-SQL oder INGRES-SQL beziehen.

4. Wie muß die Lösung von 1 geändert werden, um die gleiche Liste absteigend sortiert nach der jeweiligen Gesamtumsatzmenge zu erhalten? (1 Zeile)

5. Wie muß die Lösung von 1 geändert werden, um die gleiche Liste beschränkt auf den einzigen Großkunden (KundenNr 10000) zu erhalten? (1 Zeile)

6. Welches SQL-Konzept, das in verschiedenen Datenbanksystemen in unterschiedlicher Syntax ausgedrückt wird, ermöglicht es, die Listen 1 und 2 in einem Arbeitsgang zu erzeugen? (1 Zeile)

Vorlesungsüberblick

01. Formalien
02. Grundbegriffe, Datenbank-Architektur
03. Datenschutz
04. Software Engineering: die beiden essentiellen Entwurfsebenen
05. Synthetische Datenmodellierung: Math. Modellierung von Mengen
    Math. Modellierung von Beziehungen zwischen Mengen
06. Analytische Datenmodellierung: Normalisierungskalkül
07. Referenzmodelle
08. Spezialfälle
09. SQL im Selbststudium

Entwurf einer einfachen Datenbank zur Einkaufszettel-Verwaltung

Entwurf einer einfachen Datenbank zur Einkaufszettelverwaltung

Was soll die Datenbank können?

1. Artikel (Waren) speichern
2. Geschäfte speichern
3. Speichern, in welchen Geschäften man welche Artikel bekommt
4. Sonderangebote speichern

5. Einkaufsliste anzeigen / drucken
6. Geschäftsliste anzeigen / drucken
7. Anzeigen / drucken,
welche Artikel man in einem bestimmten Geschäft bekommt
8. Anzeigen / drucken,
in welchen Geschäften man einen bestimmten Artikel bekommt

1. Was ist eine Datenbank?

Genauer sagt man Datenbank-System.

1. Eine Menge von Daten, z.B. Adressen, Artikel:
die eigentliche Datenbank

2. Ein Programm zur Verwaltung der Daten
das Datenbank-Verwaltungssystem, Database Management System DBMS

2. Wofür braucht man Datenbanken?

1. Daten speichern

2. Daten auswerten: Verknüpfungen, Benutzersichten (Views)

3. Verknüpfung einzelner Datenbereiche (Beziehungen zwischen Tabellen)
z.B. in welchen Geschäften bekommt man welche Artikel

4. Datenbankweite Änderungen / Berechnungen
z.B. Preiserhöhung um einen bestimmten Prozentsatz für alle Artikel

5. Vermeidung der Mehrfachspeicherung von Daten (Redundanz)
Wenn es in einem Geschäft mehrere Artikel gibt,
will man die Adresse des Geschäfts nur einmal speichern;
ebenso wenn man einen Artikel in mehreren Geschäften bekommt.

3. Wofür braucht man Datenbanken noch?

1. Speicherung großer Datenmengen

2. Zentrale Datenspeicherung in einem Unternehmen (ERP-Systeme)

3. Pflege von Benutzerberechtigungen
z.B. darf nicht jeder Rechnungen erstellen

4. Verfahren bei konkurrierenden Zugriffen
z.B. zwei Angestellte wollen bei einem Artikel
unterschiedliche Preisänderungen durchführen

3. Wofür braucht man Datenbanken noch?

1. Verfahren zur Erhaltung der Widerspruchsfreiheit (Konsistenz)
1.1 Verfahren zur Gruppierung von Aktionen (Transaktion)
z.B. die Preise aller Artikel um 1% erhöhen
1.2 Verfahren bei Systemabsturz, Stromausfall (Recovery, Rollback)

2. Schneller Zugriff auf Daten (Performance)
2.1 Direktzugriff über Schlüsselfelder
2.2 Permanente automatische Sortierungen

4. Welche Datenbank-Verwaltungssysteme gibt es?

DB2 (IBM)
Oracle
MS SQL Server
mySQL
Access (MS Office)
SQLite

Effektive Vorstufen von Datenbank-Verwaltungssystemen:
Tabellenkalkulationsprogramme oder Spreadsheet-Programme
Excel
OpenOfficeCalc

5. Namen von Tabellenzellen (Variablennamen)

Tabellenname!Spaltenbuchstabe und Zeilennummer, z.B. Artikel!D3
In echten Datenbanken kann man richtige Spaltennamen vergeben.

	A	B	C	D	E
1					
2					
3					
4					

	Artikel	Geschäfte	Tabelle3	Tabelle4	Tabelle5	

6. Programmieren in Excel

Wertzuweisung bei numerischen Variablen:
Rechenformeln: z.B. Preiserhöhung um 10 Cent
=D7+0,1

Wertzuweisung bei Textvariablen:
z.B. Verkettung von zwei Feldern (Vorname und Name)
=VERKETTEN(B4; C4) oder B4&C4 oder besser B4&“ “&C4

Bedingte Anweisungen
Vergleichsoperatoren (<, <=, == etc.)
Verknüpfung von Vergleichsbedingungen (UND, ODER)
=WENN(Bedingung; Ja-Wert; Nein-Wert)
=VERKETTEN(C2; (WENN(D2<>""; ", "; "")); D2)

7. Beispiele

Preiserhöhung um 10 Cent bei allen Artikeln → Tabelle
=Variablenname+0,1

Preiserhöhung um 10 Prozent bei allen Artikeln → Tabelle
=Variablenname*1,1

Verkettung von Geschäftsname, Adresse → Tabelle
=VERKETTEN(B5; C5) oder B5&C5

Schulaufgabenbewertung → Tabelle

Sortieren und Filtern
Pulldown-Menü Daten

8. Datenmodell der Einkaufszettel-Verwaltung

Primärschlüssel
Fremdschlüssel

9. Werte mit Hilfe von Nummern oder Schlüsseln (Kürzeln) suchen

Der INDEX-Befehl liefert den Wert in einer bestimmten Matrixzelle (keine Spaltennummer, wenn die Matrix nur e i n e Spalte hat).
=INDEX(Matrix; ZeilenNr in der Matrix[; SpaltenNr in der Matrix])

Matrix = Teil einer Tabelle: von Zelle links oben bis Zelle rechts unten mit impliziter Zellennummerierung

	A	B	C	D	E	F
1						
2		1		1; 1	1; 2	
3		2		2; 1	2; 2	
4		3		3; 1	3; 2	
5						

9. Werte mit Hilfe von Nummern oder Schlüsseln (Kürzeln) suchen

Die Bezeichnung eines Artikels aus der Artikel-Tabelle
in die Zuordnungs-Tabelle holen;
das geschieht über Fremdschlüssel Artikel_ID in der Zuordnungs-Tabelle.

1. Der INDEX-Befehl mit (Zeilen-)Nummern (numerischer Schlüssel)
(Spalte Zuordnung!B enthält den FS Artikel_ID in Gestalt von Nummern.)
=INDEX(Artikel!C2:Artikel!C5; [Zuordnung!]B3)

2. Der INDEX-Befehl mit Kürzeln (alphanumerischer Schlüssel)

Der VERGLEICH-Befehl liefert
eine Zeilennummer in einer Vergleichsmatrix.
Die Zeilennummer im INDEX-Befehl kann so ermittelt werden:
VERGLEICH(Vergleichswert; Vergleichsmatrix; erstes Vorkommen)

Der INDEX-Befehl mit Kürzeln (alphanumerischer Schlüssel)
(Spalte Zuordnung!C enthält den FS Artikel_ID in Gestalt von Kürzeln.)

=INDEX(Artikel!C2:Artikel!C5;
VERGLEICH([Zuordnung!]C3; Artikel!B2:Artikel!B5; 0))

	A	B	C	D	E
1		Artikel_ID	Artikel_Bez		
2					
3					
4		→			
5					

Artikel | Geschäfte | Tabelle3 | Tabelle4 | Tabelle5

Datenschutz (ohne Gewähr!) und Berufsethik der Informatik

1. Datenschutz (ohne Gewähr, teilweise nach BfDI-Info 6 DSGVO)

1.0 Grundlegende Definitionen

DSG regeln
1. Rechte der Betroffenen gegenüber den verantwortlichen Stellen
2. Rechtssituation der Verantwortlichen / Auftragsverarbeiter:
   Pflichten; Einschränkungen, Zulässigkeit der Datenverarbeitung

Art. 1 (1) DSGVO: „Diese Verordnung enthält Vorschriften zum Schutz natürlicher Personen bei der Verarbeitung personenbezogener Daten und zum freien Verkehr solcher Daten.“

Art. 4 DSGVO „1. „**personenbezogene Daten**“ alle Informationen, die sich auf eine identifizierte oder identifizierbare natürliche Person (im Folgenden „**betroffene Person**“) beziehen.“

1.0 Grundlegende Definitionen

Art. 4 DSGVO „1. als **identifizierbar** wird eine natürliche Person angesehen, die direkt oder indirekt, insbesondere mittels Zuordnung zu einer Kennung wie einem Namen, zu einer Kennnummer, zu Standortdaten, zu einer Online-Kennung oder zu einem oder mehreren besonderen Merkmalen, die Ausdruck der physischen, physiologischen, genetischen, psychischen, wirtschaftlichen, kulturellen oder sozialen Identität dieser natürlichen Person sind, identifiziert werden kann“

1.0 Grundlegende Definitionen (Rechtspraxis)

Bestimmbarkeit: Bestimmbar ist eine Person, wenn ihre Identität unmittelbar oder mittels Zusatzwissen festgestellt werden kann. Daten sind personenbezogen, wenn ein Personenbezug hergestellt werden kann.

Gilt auch bei engen finanziellen, persönlichen oder wirtschaftlichen Verflechtungen zwischen einer natürlichen und einer juristischen Person: In diesen Fällen werden gewerbliche und Wirtschaftsdaten zu personen-bezogenen; die beiden Arten sind nicht mehr trennbar.

Berufliche und geschäftliche Sphäre einbezogen!
Einzelfirma, Ein-Mann-GmbH, Einzelkaufmann
Art der Beteiligung an einer Gesellschaft
Art der Zugehörigkeit zu und Funktion in einem Unternehmen
→ EMail-Verschlüsselung!

Quellen: Landesbeauftragter NRW, Bundesbeauftragter

1.0 Grundlegende Definitionen

Besondere Kategorien personenbezogener Daten

Die Verarbeitung von personenbezogenen Daten, aus denen die rassische und ethnische Herkunft, politische Meinungen, religiöse oder weltanschauliche Überzeugungen oder die Gewerkschaftszugehörigkeit hervorgehen, sowie von genetischen Daten, biometrischen Daten zur eindeutigen Identifizierung, Daten über Gesundheit oder Sexualleben und sexuelle Ausrichtung ist grundsätzlich untersagt (Art. 9 (1) DSGVO) – es sei denn es liegen bestimmte ausdrücklich geregelte Ausnahmen vor (Art. 9 (2) DSGVO). Es muss eine Einwilligung ausdrücklich erfolgen.

Ausnahmeregelungen in § 22, 28, 29 BDSG

1.0 Grundlegende Definitionen

Art. 4 DSGVO: „7. „**Verantwortlicher**“ die natürliche oder juristische Person, Behörde, Einrichtung oder andere Stelle, die allein oder gemeinsam mit anderen über die Zwecke und Mittel der Verarbeitung von personenbezogenen Daten entscheidet“

§ 2 (4) BDSG: „**Nichtöffentliche Stellen** sind natürliche und juristische Personen, Gesellschaften und andere Personenvereinigungen des privaten Rechts“

Art. 4 DSGVO: „8. „**Auftragsverarbeiter**“ eine natürliche oder juristische Person, Behörde, Einrichtung oder andere Stelle, die personenbezogene Daten im Auftrag des Verantwortlichen verarbeitet“

1.0 Grundlegende Definitionen

Art. 4 DSGVO: „2. „**Verarbeitung**“ jeden mit oder ohne Hilfe automatisierter Verfahren ausgeführten Vorgang oder jede solche Vorgangsreihe im Zusammenhang mit personenbezogenen Daten wie das Erheben, das Erfassen, die Organisation, das Ordnen, die Speicherung, die Anpassung oder Veränderung, das Auslesen, das Abfragen, die Verwendung, die Offenlegung durch Übermittlung, Verbreitung oder eine andere Form der Bereitstellung, den Abgleich oder die Verknüpfung, die Einschränkung, das Löschen oder die Vernichtung“

Art. 4 DSGVO: „6. „**Dateisystem**“ jede strukturierte Sammlung personenbezogener Daten, die nach bestimmten Kriterien zugänglich sind, unabhängig davon, ob diese Sammlung zentral, dezentral oder nach funktionalen oder geografischen Gesichtspunkten geordnet geführt wird“ [d. h. digitale Daten, Karteien und Formulare; nicht: Aktenunterlagen]

1.0 Basic definitions – English

GDPR – General Data Protection Regulation – gdpr-info.eu
Natürliche, juristische Person: natural, legal person
Personenbezogene Daten: personal data
Betroffener: **data subject** – identified or identifiable natural person
Verantwortlicher: **controller**
Auftragsverarbeiter: **processor**
Verarbeitung: processing; automatisiert: by automated means
Dateisystem: filing system
Rechtmäßigkeit: lawfulness
Zweckbindung: purpose limitation
Richtigkeit: **accuracy**
Speicherbegrenzung: storage limitation
Integrität und Vertraulichkeit: integrity and confidentiality
TOM: technical and organisational measures

1.0 Basic definitions – English

Aufsichtsbehörde: supervisory authority
– federführende Aufsichtsbehörde: lead supervisory authority
Auskunftsrecht: right of access
Beschäftigungskontext: context of employment
Datenschutzbeauftragter: **data protection officer**
Datenschutz-Folgenabschätzung: **data protection impact assessment**
Datenschutzverletzung: **personal data breach**
Erheben: collect, obtain
Kohärenzverfahren: consistency mechanism
Meldung: notification
Rechenschaftspflicht: **accountability**
Rechtsbehelf: judicial remedy
Schadenersatz: compensation
Verzeichnis von Verarbeitungstätigkeiten: records of processing activities
Zustimmung: consent

1.1 Rechtsvorschriften

1.1.1 Besondere Geheimhaltungsvorschriften

Sozial(daten)geheimnis: I §35 SGB [Sozialgesetzbuch], X §§67-85 SGB

Heilberufe (ärztliche Schweigepflicht, §27 BayKrG [Bayer. Krankenhaus-gesetz]), Rechtsanwälte (§43a BRAO, §2 BORA),
Steuerberater (Steuergeheimnis: §§57, 62 StBerG, § 30 AO), Sozialberater
Allgemein §203 StGB [Strafgesetzbuch]

Akteneinsicht: §29 BayVwVfG [Bayer. Verwaltungsverfahrensgesetz]

Schulnoten: §62 BayEUG [Bay. G üb. d. Erziehungs- u. Unterrichtswesen]

Ausschluss der Öffentlichkeit in Gerichtsverhandlungen

Geschäfts-, Betriebsgeheimnisse: Unternehmen (StGB, UWG, nicht DS!)

1.1.1 Subsidiarität von Datenschutz-Gesetzen

BDSG ist **subsidiär** („zur Aushilfe dienend“), nachrangig
§1(2) BDSG: „**Andere Rechtsvorschriften** des Bundes über den Datenschutz **gehen den Vorschriften dieses Gesetzes vor** …
Die Verpflichtung zur Wahrung gesetzlicher Geheimhaltungspflichten … bleibt unberührt.“

1.1.2 Aktuelle Datenschutzgesetze

EU-Richtlinie 95/46/EG und altes BDSG zum 24.05.2018 aufgehoben

EU: Ab 25.05.2018: EU-Verordnung 2016/679 vom 14.04.2016
EU-Datenschutz-Grundverordnung DSGVO
Gesetzescharakter; rechtsverbindlich für alle Mitgliedstaaten
Anwendungsvorrang gegenüber nationalem Recht (D: §1(5) BDSG)
ca. 70 **Öffnungsklauseln** für nationales Recht
(schwierig auffindbar; enthalten das Wort „Mitgliedstaat“)
Daneben: EU-Richtlinie (Directive) 2016/680 vom 14.04.2016
Datenschutz bei justizieller und polizeilicher Zusammenarbeit

D: **Bundesdatenschutzgesetz BDSG-neu** = Art. 1 DSAnpUG-EU
Kommentare: Spiros Simitis; Herbert Auernhammer; Kurt Nagel
Bayerisches Datenschutzgesetz BayDSG

1.1.2 Datenschutz-Grundverordnung EU-DSGVO (2016/679)

Kap. I: Allgemeine Bestimmungen (Art. 1-4)
Kap. II: Grundsätze (Art. 5-11)
Kap. III: Rechte der betroffenen Person
Abschnitt 1: Transparenz und Modalitäten (Art. 12)
Abschnitt 2: Informationspflicht und Recht auf Auskunft (Art. 13-15)
Abschnitt 3: Berichtigung und Löschung (Art. 16-20)
Abschnitt 4: Widerspruchsrecht, automatis. Entscheidungsfindung (21f.)
Abschnitt 5: Beschränkungen (Art. 23)
Kap. IV: Verantwortlicher und Auftragsverarbeiter
Abschnitt 1: Allgemeine Pflichten (Art. 24-31)
Abschnitt 2: Sicherheit personenbezogener Daten (Art. 32-34)
Abschnitt 3: Datenschutz-Folgenabschätzung, vorherige Konsultation (35f)
Abschnitt 4: Datenschutzbeauftragter (Art. 37-39)
Abschnitt 5: Verhaltensregeln und Zertifizierung (Art. 40-43)

1.1.2 Datenschutz-Grundverordnung EU-DSGVO (2016/679) Teil 2

Kap. V: Übermittlungen personenbezogener Daten an Drittländer oder an internationale Organisationen (Art. 44-50)
Kap. VI: Unabhängige Aufsichtsbehörden
Abschnitt 1: Unabhängigkeit (Art. 51-54)
Abschnitt 2: Zuständigkeit, Aufgaben und Befugnisse (Art. 55-59)
Kap. VII: Zusammenarbeit und Kohärenz [Aufsichtsbehörden]
Abschnitt 1: Zusammenarbeit (Art. 60-62)
Abschnitt 2: Kohärenz (Art. 63-67)
Abschnitt 3: Europäischer Datenschutzausschuss (Art. 68-76)
Kap. VIII: Rechtsbehelfe, Haftung und Sanktionen (Art. 77-84)
Kap. IX: Vorschriften für besondere Verarbeitungssituationen (85-91)
Kap. X: Delegierte Rechtsakte und Durchführungsrechtsakte (92-93)
Kap. XI: Schlussbestimmungen (Art. 94-99)

1.1.2 Bundesdatenschutzgesetz (neu) BDSG

Teil 1: Gemeinsame Bestimmungen

Kap. 1: Anwendungsbereich und Begriffsbestimmungen (§§ 1-2)
Kap. 2: Rechtsgrundlagen der Verarbeitung personenbezogener Daten (§§ 3-4)
Kap. 3: Datenschutzbeauftragte öffentlicher Stellen (§§ 5-7)
Kap. 4: Bundesbeauftragter für Datenschutz und Informationsfreiheit (§§ 8-16)
Kap. 5: Europäischer Datenschutzausschuss, Zusammenarbeit von Bund und Ländern (§§ 17-19)
Kap. 6: Rechtsbehelfe (§§ 20-21)

Teil 2: Verarbeitungen gemäß Art. 2 DSGVO

[Sachlicher Anwendungsbereich: automatisierte Verarbeitung personenbezogener Daten sowie für die nichtautomatisierte Verarbeitung in einem Dateisystem gespeicherter pers.bez. Daten]

Kap. 1: Rechtsgrundlagen der Verarbeitung (§§ 22-31)
Kap. 2: Rechte der betroffenen Person (§§ 32-37)
Kap. 3: Pflichten der Verantwortlichen und Auftragsverarbeiter (§§ 38-39)
Kap. 4: Aufsichtsbehörden [der Länder] für nichtöffentliche Stellen (§ 40)
Kap. 5: Sanktionen (§§ 41-43)
Kap. 6: Rechtsbehelfe (§ 44)

Teil 3: Verarbeitungen gemäß Art. 1(1) EU-Richtlinie 2016/680

1.1.2 Bundesdatenschutzgesetz (neu) BDSG

Im BDSG hauptsächlich genutzte nationale Öffnungsmöglichkeiten

Besondere Kategorien (Art. 9 DSGVO; §§ 22, 28, 29 BDSG)
Zweckänderung (Art. 5 (1) b DSGVO; §§ 23, 24 BDSG)
Wissenschaftsprivileg (Art. 5 (1) b und 89 DSGVO; §§ 28, 29 BDSG)
Beschäftigungsverhältnis (Art. 88 DSGVO; § 26 BDSG)
Betroffenenrechte (Art. 13-21 DSGVO; §§ 32-37 BDSG)
Datenschutzbeauftragte (Art. 37-39 DSGVO; § 38 BDSG)
Aufsichtsbehörden (Art. 51-67 DSGVO; §§ 8-14, 17-19, 40 BDSG)

1.2 Datenschutzgesetze (Auszüge)

1.2.1 Sachlicher Anwendungsbereich

Art. 2 (1) DSGVO: „Diese Verordnung gilt für die **ganz oder teilweise automatisierte Verarbeitung** personenbezogener Daten sowie
für die **nichtautomatisierte Verarbeitung** personenbezogener Daten, die in einem **Dateisystem** gespeichert sind oder gespeichert werden sollen.“

Gilt nicht für die Verarbeitung zu persönlichen und familiären Zwecken (Haushaltsausnahme, Art. 2 (2) c DSGVO).
Gilt nicht für die Abwehr von Gefahren für die öffentliche Sicherheit (Art. 2 (2) d DSGVO).

Ähnlich § 1 (1) BDSG

1.2 Datenschutzgesetze (Auszüge)
1.2.1 Räumlicher Anwendungsbereich

§ 1 (1) BDSG: Öffentlich und nicht-öffentlich [wenige Sonderregelungen]

Nicht-öffentlich:
Niederlassungsprinzip: Niederlassung des Verantwortlichen in der EU, unabhängig vom Datenverarbeitungsort (Art. 3(1) DSGVO, §1(4)2 BDSG)
Verarbeitungsortprinzip: umgekehrt (Art. 3(3) DSGVO, §1(4)1 BDSG)
Marktortprinzip [NEU]: nicht nur für
in der EU niedergelassene Verantwortliche / Auftragsverarbeiter
Voraussetzung ist nach Art. 3 (2) DSGVO (§ 1 (4) 3 BDSG) lediglich, dass sich ein Angebot an einen nationalen Markt in der EU richtet oder dass die DV der Beobachtung des Verhaltens von Personen in der EU dient

Medienprivileg: Ausgleich zwischen Persönlichkeitsschutz und Kommunikationsfreiheiten bleibt Mitgliedstaaten vorbehalten (Art. 85 DSGVO).

1.2.2 Zulässigkeit der Verarbeitung 1

Grundsätze für die Verarbeitung personenbezogener Daten
Verbot mit Erlaubnisvorbehalt (Art. 5 (1) DSGVO):

a) **Rechtmäßigkeit**
b) **Zweckbindung**: festgelegte, eindeutige und legitime Zwecke
c) **Datenminimierung**: Beschränkung auf das notwendige Maß
(bisher: Datenvermeidung, Datensparsamkeit; jetzt: **Technikgestaltung**)
d) **Richtigkeit**: Unrichtiges unverzüglich löschen oder berichtigen
e) **Speicherbegrenzung** [Dauer]: nur so lange, wie für Verarbeitungszweck erforderlich
f) **Integrität, Vertraulichkeit** [**Datensicherheit**]: **TOM**

(2) **Rechenschaftspflicht**: Der Verantwortliche … muss dessen Einhaltung [des Absatzes 1] nachweisen können (auch Art. 24 (1) DSGVO).

1.2.2 Zulässigkeit der Verarbeitung 2

Rechtmäßigkeit der Verarbeitung

Art. 6 (1) DSGVO

„…, wenn mindestens eine der nachstehenden Bedingungen erfüllt ist:

a) Einwilligung für einen oder mehrere bestimmte Zwecke

- Art. 7 DSGVO: Nachweispflicht des Verantwortlichen
- Einwilligung von anderen Sachverhalten unterscheidbar und unabhängig
- Klare und einfache Sprache
- Einwilligung jederzeit widerrufbar

b) für die Erfüllung eines Vertrags

c) zur Erfüllung einer rechtlichen Verpflichtung

d) lebenswichtige Interessen … einer natürlichen Person zu schützen

e) … im öffentlichen Interesse … oder in Ausübung öffentlicher Gewalt

f) Wahrung der berechtigten Interessen des Verantwortlichen …, sofern nicht die … Grundrechte … der betroffenen Person überwiegen“

1.2.2 Zulässigkeit der Verarbeitung 3

Zweckänderung

Es sind nur solche Änderungen des Verarbeitungszwecks erlaubt, die mit dem ursprünglichen Erhebungszweck vereinbar sind (Art. 5 (1) b sowie Art. 6 (4) DSGVO). Die Datenschutz-Grundverordnung stellt in Art. 6 (4) Kriterien auf, die zu berücksichtigen sind. Hierzu zählen u. a. die Verbindung zwischen den Zwecken, der Gesamtkontext, in dem die Daten erhoben wurden, die Art der personenbezogenen Daten, etc.

Art. 5 (1) b DSGVO: „Weiterverarbeitung für im öffentlichen Interesse liegende Archivzwecke, für wissenschaftliche oder historische Forschungszwecke oder für statistische Zwecke gilt gemäß Artikel 89 Absatz 1 nicht als unvereinbar mit den ursprünglichen Zwecken“

§ 24 BDSG: öffentliche Sicherheit, zivilrechtliche Ansprüche

1.2.2 Zulässigkeit der Verarbeitung 4

Integrität, Vertraulichkeit [**Datensicherheit**]:

Schutz vor … unrechtmäßiger Verarbeitung und vor … Verlust … durch geeignete **technische und organisatorische Maßnahmen [TOM]** (Art. 5 (1) f DSGVO)

Privacy by Design/ Default; **Technikgestaltung**	(Art. 25 DSGVO)
Auftragsverarbeitung	(Art. 28 DSGVO)
Meldungen über Datenschutzverletzungen	(Art. 33, 34 DSGVO)
Datenschutz-Folgenabschätzung	(Art. 35, 36 DSGVO)
Betriebliche / behördliche Datenschutzbeauftragte	(Art. 37-39 DSGVO)
Selbstregulierung durch die Verantwortlichen	(Art. 40-43 DSGVO:

Verhaltensregeln (Code of Conduct), Zertifizierungen;
Register beim Europäischen Datenschutzausschuss)

1.2.3 Rechte der Betroffenen 1 (DSGVO)

12 Transparente Information, Komm., Modalitäten für Rechte-Ausübung
(1) präzise, transparente, verständliche Form, klare, einfachen Sprache
(3) innerhalb eines Monats (plus zwei), (5) unentgeltlich

13 Informationspflicht bei Erhebung bei der betroffenen Person
Verantwortlicher, Zweck, Rechtsgrundlage, Empfänger, Dauer etc.
Rechte auf Berichtigung, Löschung, Widerruf der Einwilligung, Widerspruch gegen Verarbeitung, Beschwerde bei Aufsichtsbehörde etc.
Einschränkungen: § 32 BDSG (öff. Sicherheit, Strafrecht, Zivilrecht etc.)
14 Informationspflicht bei anderweitigen Erhebungen
Zusätzlich: Quellen; automatisierte Entscheidungsfindung (Art. 22)
Einschränkungen: § 33 BDSG (öff. Sicherheit, Strafrecht, Zivilrecht etc.)

15 Auskunftsrecht: Angaben wie oben; Verweigerung dokumentieren
§ 34 (2) BDSG: Die zu diesem Zweck gespeicherten Daten dürfen nur dafür sowie für Zwecke der Datenschutzkontrolle verarbeitet werden

1.2.3 Rechte der Betroffenen 2 (DSGVO)

Art. 16 DSGVO: Recht auf Berichtigung
17 Recht auf Löschung
Für Zwecke nicht mehr nötig, Widerruf, Widerspruch, Unrechtmäßigkeit
§ 35 BDSG: bei unverhältnismäßig hohem Aufwand nur Einschränkung
17 (2) „Recht auf Vergessenwerden“ (bei Veröffentlichung der Daten)
18 Einschränkung der Verarbeitung (bisher: Sperrung)
Richtigkeit bestritten; Unklarheit über Widerspruch
19 Mitteilung an Empfänger bei Berichtigung, Löschung, Einschränkung

20 Recht auf Datenübertragbarkeit (zwischen Verantwortlichen)

21 Widerspruchsrecht (gegen Verarbeitung nach Art. 6 (1) e,f DSGVO)
uneingeschränkt bei Direktmarketing / Profiling (Ausnahme § 36 BDSG)

Art. 77 (1) DSGVO: Recht auf Beschwerde bei einer Aufsichtsbehörde
Art. 79 DSGVO: Recht auf gerichtlichen Rechtsbehelf (§ 44 BDSG)
Art. 82 (1) DSGVO: Recht auf Schadenersatz bei im(materiellen) Schäden

1.2.3 Rechte der Betroffenen: Werbung (Ges. gg. unlaut. Wettbewerb)

§ 7 UWG Unzumutbare Belästigungen
§ 7 (2) UWG Werbung mit 2. Telefonanruf, 3. elektronische Post (Spam)
ohne vorherige ausdrückliche Einwilligung
§ 7 (3) UWG Ausnahmen von (2):
1. E-Mail-Adresse im Rahmen eines Verkaufs erhalten
3. Kunde hat Verwendung nicht widersprochen
4. Kunde wird jedes Mal auf Möglichkeit des Widerspruchs hingewiesen

Newsletter gilt als E-Mail-Werbung,
aber schärfere Rechtsprechung ohne explizite gesetzliche Regelung:
Daten aus Kaufvertrag nicht einfach zum Newsletter-Versand verwendbar
Doppelte Bestätigung (double opt-in) erforderlich
opt-out genügt definitiv nicht

1.2.4 Automatisierte Einzelentscheidung

Art. 22 (1) DSGVO: „Die betroffene Person hat das Recht,
nicht einer ausschließlich auf einer automatisierten Verarbeitung —
einschl. Profiling — beruhenden Entscheidung unterworfen zu werden,
die ihr gegenüber rechtliche Wirkung entfaltet …“
(2)-(4) Ausnahmen: Vertragsabschluss, -erfüllung, Einwilligung,
Rechtsvorschrift, Überwachung von Betrug und Steuerhinterziehung;
Recht auf Eingreifen einer Person seitens des Verantwortlichen, auf
Anfechtung der Entscheidung; nicht: besondere Kategorien

Erwäg.Grund (71) Online-Kreditantrag, Online-Einstellungsverfahren
„Profiling“: Analyse oder Prognose bezüglich Arbeitsleistung,
wirtschaftliche Lage, Gesundheit, persönliche Vorlieben oder Interessen,
Zuverlässigkeit oder Verhalten, Aufenthaltsort oder Ortswechsel

Kurz: § 54 BDSG

1.2.4 Verbraucherkredite und Scoring

§ 30 (2) BDSG: „Wer den Abschluss eines Verbraucherdarlehensvertrags infolge einer Auskunft einer [Auskunftei] ablehnt,
hat den Verbraucher unverzüglich hierüber sowie über die erhaltene Auskunft zu unterrichten.“

§ 31 (1) BDSG: Wahrscheinlichkeitswert nur zulässig, wenn
2. Daten … nach wissenschaftlich anerkannten mathematisch-statistischen Verfahren nachweisbar … erheblich sind und
3. nicht ausschließlich Anschriftendaten genutzt wurden und
4. bei Anschriftendaten die betroffene Person … unterrichtet worden ist

(2) Wahrscheinlichkeitswert über Zahlungsfähig- und Zahlungswilligkeit: Nur solche offenen Forderungen berücksichtigungsfähig, für die Gerichtsurteil vorliegt, die der Schuldner ausdrücklich anerkannt hat …

1.2.5 Beschäftigte (Art. 88 DSGVO nationale Öffnungsklausel)

§ 26 (8) BDSG: „Bewerber … sowie Personen,
deren Beschäftigungsverhältnis beendet ist, sind Beschäftigte.“
„**Beschäftigte** sind:
1. Arbeitnehmerinnen und Arbeitnehmer
2. zu ihrer Berufsausbildung Beschäftigte 6. in Heimarbeit Beschäftigte
7. Beamte, … Richter …, … Soldaten sowie Zivildienstleistende“

§ 26 (1) BDSG: „Personenbezogene Daten von Beschäftigten dürfen für Zwecke des **Beschäftigungsverhältnisses** verarbeitet werden, wenn dies … für dessen Durchführung … erforderlich ist.…
Zur Aufdeckung von **Straftaten** … nur dann, wenn
zu dokumentierende tatsächliche Anhaltspunkte den Verdacht begründen,
… die Verarbeitung zur Aufdeckung erforderlich ist und
das schutzwürdige Interesse … des Beschäftigten … nicht überwiegt, …
Art und Ausmaß im Hinbl. auf den Anlass nicht unverhältnismäßig sind.“

1.2.6 Anforderungen an verantwortliche Stellen 1 (DSGVO)

Art. 24 DSGVO Verantwortung des Verantwortlichen
(1) … setzt unter Berücksichtigung … der unterschiedlichen Eintrittswahrscheinlichkeit und Schwere der Risiken … geeignete **technische und organisatorische Maßnahmen** um, um sicherzustellen und nachweisen zu können, dass die Verarbeitung gemäß dieser Verordnung erfolgt.

Art. 25 DSGVO Technikgestaltung, datenschutzfreundl. Voreinstellungen
Pseudonymisierung, Datenminimierung etc.

Art. 28 DSGVO Auftragsverarbeiter
(1) „Erfolgt eine Verarbeitung im Auftrag eines Verantwortlichen, so arbeitet dieser nur mit Auftragsverarbeitern, die hinreichend Garantien dafür bieten, dass geeignete TOM so durchgeführt werden, dass die Verarbeitung im Einklang mit den Anforderungen dieser Verordnung erfolgt und den Schutz der Rechte der betroffenen Person gewährleistet.“

1.2.6 Anforderungen an verantwortliche Stellen 2 (DSGVO)

Art. 30 DSGVO **Verzeichnis von Verarbeitungstätigkeiten**
Dokumentationspflicht von Verantwortlichen und Auftragsverarbeitern

Art. 32 DSGVO Sicherheit der Verarbeitung
TOM, Schutzniveau, Prüfverfahren etc. (ähnlich Art. 24)

Art. 33 DSGVO Meldung von Verletzungen an die Aufsichtsbehörde
(1) Im Falle einer Verletzung des Schutzes personenbezogener Daten meldet sie der Verantwortliche unverzüglich und möglichst binnen 72 Stunden nach Bekanntwerden der zuständigen Aufsichtsbehörde

Art. 34 DSGVO Benachrichtigung betroffener Personen über Verletzungen
(1) unverzüglich bei hohem Risiko

1.2.6 Anforderungen an verantwortliche Stellen 3 (DSGVO)

Art. 35 DSGVO **Datenschutz-Folgenabschätzung** (alt: Vorabkontrolle)
(1) Voraussichtlich hohes Risiko für Rechte und Freiheiten nat. Personen
(2) Verantwortlicher holt Rat des Datenschutzbeauftragten ein.
(3) a) systematische und umfassende Bewertung persönlicher Aspekte einschl. Profiling als Grundlage für rechtswirksame Entscheidungen
b) umfangreiche Verarbeitung besonderer Kategorien pers.bez. Daten
c) systematische umfangreiche Überwachung öffentlich zugängl. Bereiche
(4), (5) Aufsichtsbehörden erstellen Positivlisten und Negativlisten
(7) Inhalt der Datenschutz-Folgenabschätzung → Datenschutz-Bewusstsein

Art. 36 DSGVO **Vorherige Konsultation**
(1): „Der Verantwortliche konsultiert vor der Verarbeitung die Aufsichtsbehörde, wenn aus einer Datenschutz-Folgenabschätzung hervorgeht, dass die Verarbeitung ein hohes Risiko zur Folge hätte, sofern der Verantwortliche keine Maßnahmen zur Eindämmung des Risikos trifft.“

1.2.6 Anforderungen an verantwortliche Stellen 4 (DSGVO)

Selbstregulierung: Nachweis des rechtskonformen Verhaltens
von Verantwortlichen und Auftragsverarbeitern gemäß Art. 24-36 DSGVO

Art. 40 DSGVO Verhaltensregeln, Code of Conduct
Art. 41 DSGVO Überwachung durch akkreditierte Stellen

Art. 42 DSGVO Zertifizierung von Verantwortl. / Auftragsverarbeitern
Art. 43 DSGVO Akkreditierte Zertifizierungsstellen (§ 39 BDSG)

Datengeheimnis

§53 BDSG (Teil 3!): „Mit Datenverarbeitung befasste Personen dürfen personenbezogene Daten nicht unbefugt verarbeiten. Sie sind bei der Aufnahme ihrer Tätigkeit auf das Datengeheimnis zu **verpflichten**. Das Datengeheimnis besteht nach der Beendigung ihrer Tätigkeit fort.“

1.2.7 Datenschutzbeauftragter: Benennung (Art. 37 DSGVO)

Art. 37 (1) DSGVO: Der Verantwortliche und der Auftragsverarbeiter benennen auf jeden Fall einen Datenschutzbeauftragten, wenn
a) Verarbeitung von einer Behörde oder öffentlichen Stelle durchgeführt
b) Kerntätigkeit: Verarbeitungsvorgänge, welche aufgrund Art, Umfang und/oder Zwecke eine umfangreiche regelmäßige und systematische Überwachung von betroffenen Personen erforderlich machen, oder
c) umfangreiche Verarbeitung besonderer Kategorien von Daten gemäß Artikel 9 oder von personenbezogenen Daten über strafrechtliche Verurteilungen und Straftaten gemäß Artikel 10

(5) „Der Datenschutzbeauftragte wird auf der Grundlage seiner beruflichen Qualifikation und insbesondere des **Fachwissens** benannt, das er auf dem Gebiet des Datenschutzrechts und der Datenschutzpraxis besitzt …“.
(6) … Beschäftigter des Verantwortlichen / Auftragsverarbeiter oder seine Aufgaben auf der Grundlage eines Dienstleistungsvertrags erfüllen.

1.2.7 Datenschutzbeauftragter: nichtöffentliche Stellen (BDSG § 38)

§ 38 (1) BDSG: der Verantwortliche und der Auftragsverarbeiter benennen eine/n Datenschutzbeauftragte/n,
„soweit sie in der Regel **mindestens zehn Personen** ständig mit der automatisierten Verarbeitung personenbezogener Daten beschäftigen“.

„Nehmen der Verantwortliche / Auftragsverarbeiter Verarbeitungen vor, die einer **Datenschutz-Folgenabschätzung** nach Art. 35 DSGVO unterliegen,
oder verarbeiten sie personenbezogene Daten **geschäftsmäßig** zum Zweck der Übermittlung, der anonymisierten Übermittlung oder
für Zwecke der Markt- oder Meinungsforschung,
haben sie
unabhängig von der Anzahl der mit der Verarbeitung beschäftigten Personen
eine/n Datenschutzbeauftragte/n zu benennen.“

1.2.7 Datenschutzbeauftragter: Stellung (Art. 38 DSGVO)

Art. 38 (2) DSGVO: Der Verantwortliche / Auftragsverarbeiter stellen die für die Erfüllung der Aufgaben erforderlichen Ressourcen und den Zugang zu personenbezogenen Daten und Verarbeitungsvorgängen sowie die zur Erhaltung seines Fachwissens erforderlichen Ressourcen zur Verfügung.

(3) Der Verantwortliche / Auftragsverarbeiter stellen sicher, dass der Datenschutzbeauftragte bei der Erfüllung seiner Aufgaben keine Anweisungen bezüglich der Ausübung dieser Aufgaben erhält.
Er darf wegen der Erfüllung seiner Aufgaben nicht abberufen oder benachteiligt werden. Er berichtet unmittelbar der höchsten Managementebene des Verantwortlichen / Auftragsverarbeiters.

(4) Betroffene Personen können den Datenschutzbeauftragten bei allen Fragen zur Verarbeitung ihrer personenbezogenen Daten zu Rate ziehen.

1.2.7 Datenschutzbeauftragter: Aufgaben (Art. 39 DSGVO)

Art. 39 (1) DSGVO
a) Beratung des Verantwortlichen / Auftragsverarbeiters und der Beschäftigten hinsichtlich ihrer Pflichten nach den Datenschutzvorschriften
b) Überwachung der Einhaltung der Datenschutzvorschriften sowie der Strategien des Verantwortlichen / Auftragsverarbeiters für den Schutz personenbezogener Daten einschließl. der Zuweisung von Zuständigkeiten, der Sensibilisierung und Schulung der an den Verarbeitungsvorgängen beteiligten Mitarbeiter und der diesbezüglichen Überprüfungen;
c) Beratung im Zusammenhang mit der Datenschutz-Folgenabschätzung und Überwachung ihrer Durchführung gemäß Art. 35 DSGVO;
d) Zusammenarbeit mit der Aufsichtsbehörde;
e) Tätigkeit als Anlaufstelle für die Aufsichtsbehörde.

(2) Der Datenschutzbeauftragte trägt dem mit den Verarbeitungsvorgängen verbundenen Risiko gebührend Rechnung.

1.3 Öffentliche Kontrolle: Aufsichtsbehörden

Bundesbeauftragter für Datenschutz und Info-Freiheit (§§ 8-16 BDSG)

Art. 51 (1) DSGVO: „Jeder Mitgliedstaat sieht vor, dass eine oder mehrere unabhängige Behörden für die Überwachung der Anwendung dieser Verordnung zuständig sind.“
§ 40 (1) BDSG: „Die nach Landesrecht zuständigen Behörden überwachen im Anwendungsbereich der DSGVO bei den nichtöffentlichen Stellen die Anwendung der Vorschriften über den Datenschutz.“

Bayer. Landesbeauftragter für den Datenschutz; Beirat
Datenschutzregister (Behörden, die personenbezog. Daten verarbeiten)
Verordnung über das Datenschutzregister DSRegV

Bayer. Landesamt für Datenschutz-Aufsicht BayLDA, Ansbach

1.3 Öffentliche Kontrolle: Aufsichtsbehörden; effektive Durchsetzung

Art. 52 (1) DSGVO: Unabhängigkeit der Aufsichtsbehörden
(2) kein Weisungsempfang
(4) Jeder Mitgliedstaat stellt personelle, technische und finanzielle Ressourcen, Räumlichkeiten und Infrastrukturen sicher

Art. 53 DSGVO: Transparentes Ernennungsverfahren der Mitglieder
Art. 58 DSGVO Untersuchungsbefugnisse der Aufsichtsbehörden (damit auch Rechtsaufsicht ggü. anderen Behörden)

Art. 83 (4) DSGVO Sanktionen: „Geldbußen von bis zu 20 000 000 EUR oder im Fall eines Unternehmens von bis zu 4 % seines gesamten weltweit erzielten Jahresumsatzes des vorangegangenen Geschäftsjahrs verhängt, je nachdem, welcher der Beträge höher ist“:
Art. 84 DSGVO: Öffnungsklausel für andere länderspezifische Sanktionen
§§ 41-43 Strafvorschriften, Bußgeldvorschriften

1.3 Öffentliche Kontrolle: Aufsichtsbehörden; Koordination

One-Shop-Stop-Mechanismus:
Eine federführende Aufsichtsbehörde am Sitz der Hauptniederlassung des Verantwortlichen auch bei grenzüberschreitender Tätigkeit

Art. 56 DSGVO: Federführende Aufsichtsbehörde (Hauptniederlassung)
Art. 60 DSGVO: Zusammenarbeit der federführenden A. mit anderen
§ 18 BDSG Zusammenarbeit der Aufsichtsbehörden (Bund und Länder)
§ 19 BDSG Zuständigkeiten der Aufsichtsbehörden

Kohärenzverfahren:
Art. 63 DSGVO: Zusammenarbeit der Aufsichtsbehörden „zur einheitlichen Anwendung“ der DSGVO in der EU (One-Stop-Shop-Fälle)
Art. 64-66 Details zum Kohärenzverfahren

2. National Codes of Ethics and Professional Conduct
Berufsethik der Informatik – Nationale Ethikkodizes

2.0 USA
2.1 Canada
2.2 Australia
2.3 New Zealand
2.4 Germany: Ethical Guidelines of the GI
2.5 Informationsethik

2.0 USA

Association of Information Technology Professionals (AITP)
Code of Ethics; Standards of Conduct
http://www.aitp.org

IEEE
Code of Ethics
http://www.ieee.org

Association for Computing Machinery (ACM)
Code of Ethics and Professional Conduct
http://www.acm.org

Neumann, Peter G.: ACM Forum on Risks to the Public
in the Use of Computers and Related Systems. Articles in
ACM Software Engineering Notes; Communications of the ACM

2.1 Canada

Canadian Information Processing Society (CIPS):
Code of Ethics and Professional Conduct. Toronto [1]1985.
http://www.cips.ca/standards/isp/ethics/

1. Protect public interest and maintain integrity
2. Demonstrate competence and quality of service
3. Maintain confidential information and privacy
4. Avoid conflicts of interest
5. Uphold responsibility to the IT profession

2.2 Australia

Australian Computer Society (ACS):
Code of Ethics. Darlinghurst [1]1987.
http://www.acs.org.au/index.cfm?action=show&conID=coe

1. Uphold and advance the honor … of the profession of IT
3. Values, ideals (e.g. enhance quality of life of those affected by my work)
4. Standards of conduct
5. Priorities
6. Competence (e.g. provide products and services which match the operational and financial needs of my clients and employers)
7. Honesty
8. Social implications (e.g. consider and respect people's privacy)
9. Professional development
10. Information technology profession (e.g. seek advice from the Society when faced with an ethical dilemma I am unable to resolve by myself)

2.3 New Zealand

New Zealand Computer Society (NZCS):
Code of Ethics and Professional Conduct. Wellington [1]1987.
http://www.nzcs.org.nz/SITE_Default/about_NZCS/Code_of_Ethics.asp

1. Responsibility for the community comes before other responsibilities
2. Act with integrity, dignity, honor to merit the trust of the community and to contribute positively to the well-being of society
3. Treat people with dignity, good faith and equity, without discrimination, consideration for cultural sensitivities
4. Follow recognized professional practice
5. Develop knowledge, skills and expertise continuously
6. Apply skills and knowledge in the interests of clients or employers
7. Inform of the economic, social, environmental or legal consequences
8. Inform clients or employers of any interest in conflict with their interests

2.4 Germany

Gesellschaft für Informatik (GI):
Ethische Leitlinien / Ethical Guidelines
Informatik-Spektrum 16(1993) 238-240 [Capurro; Coy; Damker et al.];
Informatik-Spektrum 19(1996) 79-86 [Rödiger; Wilhelm]
Aktuelle Version 29.06.2018

https://gi.de/ueber-uns/organisation/unsere-ethischen-leitlinien/

Selbstverpflichtung; Begriffserläuterungen

Preamble

The German Informatics Society (GI) is a registered non-profit organization. With these guidelines, the GI seeks to establish that matters of professional ethics or moral conflicts become the subject of collaborative reflection and action.
The guidelines are designed to offer a point of orientation not only to members of the GI association, but to all persons involved in the design, manufacture, operation or use of IT systems.

The ethical guidelines outlined herein express the intent of GI members to conduct themselves in accordance with the values that form the basis of Basic Law for the Federal Republic of Germany and the Charter of Fundamental Rights of the European Union.

The GI and its members are committed to adhering to these guidelines. They also seek to insure that these guidelines find acknowledgement in public discourse outside the GI.

GI members are especially committed to respecting and protecting human dignity. Whenever norms of the state, society or the private sphere come into conflict with these values, GI members must address the issue.

GI members conduct themselves in such a way as to advocate for the right to self-determination in information and communication technologies, and for the right to guarantee confidentiality and integrity of IT systems.

GI members advocate for discrimination-free organizational structures which take into consideration the divergent needs and diversity of all human beings in the design, manufacture, operation and use of IT systems.

GI members seek to engage and educate the public in discourse concerning the ethical and moral issues pertinent to their individual and institutional conduct.
In a networked world, it is imperative that all potential courses of action be subject to interdisciplinary consideration regarding their foreseeable impact and potential consequences. This is the challenge for each of our members.

The fact that the guidelines established here are as open as they are is testimony to the fact that moral conduct cannot be governed by a definitive code of ethics or stringent regulations.

Section 1: Professional Competence

GI members stay abreast of the current state of science and technology in their respective areas of specialization; they take new developments into account and provide constructive criticism. GI members are constantly working to improve their professional competencies.

Section 2: Expertise and Communicative Competence

GI members are constantly improving their levels of expertise and communicative competencies in order to meet the demands relevant to their duties in the design, manufacture, operation and use of IT systems and to understand the surrounding professional and technical contexts.
In order to assess the consequences of IT-systems in the application environment and to propose suitable solutions, there must be a willingness to understand and take into account the rights, needs and interests of those parties who are impacted by them.

Section 3: Legal Expertise

GI members are familiar with and observant of pertinent legal regulations concerning the design, manufacture, operation and use of IT systems.
GI members, in conjunction with their expertise and professional competencies, participate actively in drafting legislative regulations.

Section 4: Powers of Discernment

GI members sharpen their powers of discernment to render themselves better equipped to contribute to design processes with individual and collective accountability.
This presupposes not only a willingness to call into question and to make judgments about individual and collective actions in public discourse, but also the ability to acknowledge the limits of one's own powers of discernment.

Section 5: Conditions of Employment

GI members are active proponents of socially equitable contractual agreements concerning terms of employment, inclusive of opportunities for professional development and shared governance.

Section 6: Organizational Structures

GI members advocate for organizational structures which foster and facilitate socially equitable contractual agreements concerning terms of employment.

Section 7: Teaching and Learning

GI members who are computer science instructors foster in their students the capacity for critical thinking;
they prepare learners to accept their own individual and collective responsibility, and they act as role models in this regard.

Section 8: Research

GI members who conduct research in the field of computer science adhere to the rules of best practices in scientific research.
Of particular importance in this regard is openness and transparency in dealing with criticism and conflicts of interest, the ability to express and to accept criticism as well as the willingness
to allow the impact of one's own scientific work in the research process to become the subject of discussion.
Scientific research breaches boundaries. These must be clearly articulated.

Section 9: Courage of Convictions

GI members staunchly advocate for the protection and safeguarding of human dignity, even when this is not explicitly mandated by laws, contracts or other norms, or when these stand in direct opposition to the protection and safeguarding of human dignity.
This applies even in situations in which GI members' obligations to clients conflict with their responsibility to third-party stakeholders.

Section 10: Social Accountability

In the design, manufacture, operation and use of IT systems, GI members should contribute to the betterment of local and global living conditions.
GI members are responsible for the social and societal consequences of their work.
Their influence on positioning, marketing and further development of IT systems should contribute to the socially acceptable and sustainable application of these technologies.

Section 11: Facilitating Self-Determination

GI members work toward ensuring that those people impacted by the usage and conditions of use of IT systems are granted adequate opportunity to participate in the design of these systems.
This is especially pertinent with regard to systems whose application involves the exerting influence over, monitoring, or surveillance of said populations.

Section 12: The German Informatics Society

The German Informatics Society encourages its members to adhere to these guidelines at all times.
The GI shall attempt to mediate between parties in situations in which conflicts arise.

2.5 Informationsethik

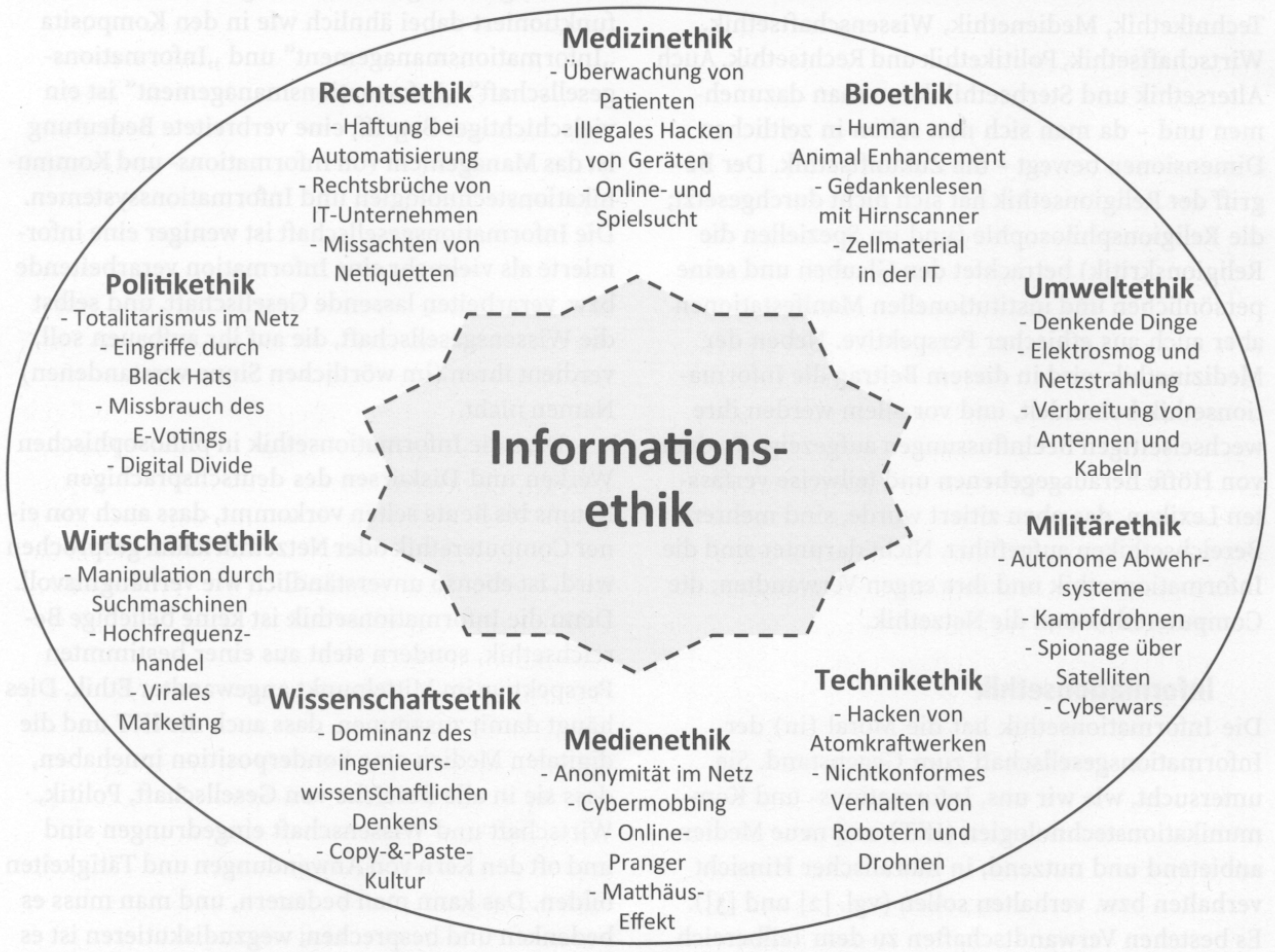


Abb. 1 Das „Ethik-Ei" – Informationsethik im Zentrum der Bereichsethiken (in Anlehnung an [1])

GESELLSCHAFT
FÜR INFORMATIK

Die Ethischen Leitlinien der Gesellschaft für Informatik e.V. (GI)

Berlin/Bonn, Juli 2018

GI.DE

Präambel

GESELLSCHAFT FÜR INFORMATIK

Die Gesellschaft für Informatik e.V. (GI) will mit diesen Leitlinien bewirken, dass **berufsethische oder moralische Konflikte** Gegenstand gemeinsamen Nachdenkens und Handelns werden. Die Leitlinien sollen den GI-Mitgliedern und darüber hinaus allen Menschen, die IT-Systeme entwerfen, herstellen, betreiben oder verwenden, eine **Orientierung** bieten.

Die vorliegenden Leitlinien sind Ausdruck des Willens der GI-Mitglieder, ihr Handeln an den Werten auszurichten, die dem **Grundgesetz der Bundesrepublik Deutschland** und der **Charta der Grundrechte der Europäischen Union** zu Grunde liegen. Die GI und ihre Mitglieder verpflichten sich zur Einhaltung dieser Leitlinien. Sie wirken auch außerhalb der GI darauf hin, dass diese im öffentlichen Diskurs Beachtung finden.

Präambel

GESELLSCHAFT FÜR INFORMATIK

Die GI-Mitglieder fühlen sich insbesondere dazu verpflichtet, die **Menschenwürde** zu achten und zu schützen. Wenn staatliche, soziale oder private Normen im Widerspruch zu diesen Werten stehen, muss dies von den GI-Mitgliedern thematisiert werden.

Die GI-Mitglieder treten dafür ein, das Recht auf **informationelle Selbstbestimmung** und das Recht auf Gewährleistung der **Vertraulichkeit und Integrität** informationstechnischer Systeme durch ihr Handeln zu befördern.

Die GI-Mitglieder setzen sich dafür ein, dass Organisationsstrukturen **frei von Diskriminierung** sind und berücksichtigen bei Entwurf, Herstellung, Betrieb und Verwendung von IT-Systemen die unterschiedlichen Bedürfnisse und die **Diversität** der Menschen.

Präambel

GESELLSCHAFT
FÜR INFORMATIK

Die GI-Mitglieder wollen den **Diskurs über ethische und moralische Fragen** ihres individuellen und institutionellen Handelns mit der Öffentlichkeit aufnehmen und Aufklärung leisten. In einer vernetzten Welt ist es notwendig, Handlungsalternativen im Hinblick auf ihre absehbaren Wirkungen und möglichen Folgen interdisziplinär zu thematisieren. **Hier ist jedes Mitglied gefordert**.

Der offene Charakter der nachfolgenden Artikel macht deutlich, dass es **keine abschließenden Handlungsanweisungen** oder starren Regelwerke für moralisch gebotenes Handeln geben kann.

Fach-, Sach- & kommunikative Kompetenz

GESELLSCHAFT FÜR INFORMATIK

Art. 1 Fachkompetenz

Das GI-Mitglied eignet sich den Stand von Wissenschaft und Technik in seinem Fachgebiet an, berücksichtigt ihn und kritisiert ihn konstruktiv. Das GI-Mitglied verbessert seine Fachkompetenz ständig.

Art. 2 Sachkompetenz und kommunikative Kompetenz

Das GI-Mitglied verbessert laufend seine Sachkompetenzen und kommunikativen Kompetenzen, so dass es die seine Aufgaben betreffenden Anforderungen an Entwurf, Herstellung, Betrieb und Verwendung von IT-Systemen und ihre fachlichen und sachlichen Zusammenhänge begreift. Um die Auswirkungen von IT-Systemen im Anwendungsumfeld beurteilen und geeignete Lösungen vorschlagen zu können, bedarf es der Bereitschaft, die Rechte, Bedürfnisse und Interessen der Betroffenen zu verstehen und zu berücksichtigen.

Juristische Kompetenz & Urteilsfähigkeit

GESELLSCHAFT FÜR INFORMATIK

Art. 3 Juristische Kompetenz

Das GI-Mitglied kennt und beachtet die einschlägigen rechtlichen Regelungen bei Entwurf, Herstellung, Betrieb und Verwendung von IT-Systemen. Das GI-Mitglied wirkt im Rahmen seiner Fach- und Sachkompetenzen an der Gestaltung rechtlicher Regelungen mit.

Art. 4 Urteilsfähigkeit

Das GI-Mitglied entwickelt seine Urteilsfähigkeit, um an Gestaltungsprozessen in individueller und gemeinschaftlicher Verantwortung mitwirken zu können. Dies setzt die Bereitschaft voraus, das eigene und das gemeinschaftliche Handeln im gesellschaftlichen Diskurs kritisch zu hinterfragen und zu bewerten sowie die Grenzen der eigenen Urteilsfähigkeit zu erkennen.

Arbeitsbedingungen & Organisationsstrukturen

GESELLSCHAFT FÜR INFORMATIK

Art. 5 Arbeitsbedingungen

Das GI-Mitglied setzt sich für sozial verträgliche Arbeitsbedingungen mit Weiterbildungs- und Gestaltungsmöglichkeiten ein.

Art. 6 Organisationsstrukturen

Das GI-Mitglied tritt aktiv für Organisationsstrukturen ein, die sozial verträgliche Arbeitsbedingungen sowie die Übernahme individueller und gemeinschaftlicher Verantwortung fördern und ermöglichen.

Lehren, Lernen & Forschen

GESELLSCHAFT FÜR INFORMATIK

Art. 7 Lehren und Lernen

Das GI-Mitglied, das Informatik lehrt, fördert die Fähigkeit zum kritischen Denken, bereitet die Lernenden auf deren individuelle und gemeinschaftliche Verantwortung vor und ist hierbei selbst Vorbild. Das GI-Mitglied, das in Schule, Hochschule oder Weiterbildung Informatik lernt, fordert dies von den Lehrenden ein.

Art. 8 Forschung

Das GI-Mitglied, das auf dem Gebiet der Informatik forscht, hält im Forschungsprozess die Regeln der guten wissenschaftlichen Praxis ein. Dazu gehören insbesondere die Offenheit und Transparenz im Umgang mit Kritik und Interessenkonflikten, die Fähigkeit zur Äußerung und Akzeptanz von Kritik sowie die Bereitschaft, die Auswirkungen der eigenen wissenschaftlichen Arbeit im Forschungsprozess zu thematisieren. Wissenschaftliche Forschung stößt an Grenzen. Diese müssen verständlich gemacht werden.

Zivilcourage & Soziale Verantwortung

GESELLSCHAFT FÜR INFORMATIK

Art. 9 Zivilcourage

Das GI-Mitglied tritt mit Mut für den Schutz und die Wahrung der Menschenwürde ein, selbst wenn Gesetze, Verträge oder andere Normen dies nicht explizit fordern oder dem gar entgegenstehen. Dies gilt auch in Situationen, in denen seine Pflichten gegenüber Auftraggebenden in Konflikt mit der Verantwortung gegenüber anderweitig Betroffenen stehen. Dies kann in begründeten Ausnahmefällen auch den öffentlichen Hinweis auf Missstände einschließen.

Art. 10 Soziale Verantwortung

Das GI-Mitglied soll mit Entwurf, Herstellung, Betrieb und Verwendung von IT-Systemen zur Verbesserung der lokalen und globalen Lebensbedingungen beitragen. Das GI-Mitglied trägt Verantwortung für die sozialen und gesellschaftlichen Auswirkungen seiner Arbeit. Es soll durch seinen Einfluss auf die Positionierung, Vermarktung und Weiterentwicklung von IT-Systemen zu deren sozial verträglicher und nachhaltiger Verwendung beitragen.

Selbstbestimmung & die Rolle der GI

GESELLSCHAFT FÜR INFORMATIK

Art. 11 Ermöglichung der Selbstbestimmung

Das GI-Mitglied wirkt darauf hin, die von IT-Systemen Betroffenen an der Gestaltung dieser Systeme und deren Nutzungsbedingungen angemessen zu beteiligen. Dies gilt insbesondere für Systeme, die zur Beeinflussung, Kontrolle und Überwachung der Betroffenen verwendet werden können.

Art. 12 Die Gesellschaft für Informatik

Die Gesellschaft für Informatik ermutigt ihre Mitglieder, sich in jeder Situation an den Leitlinien zu orientieren. In Konfliktfällen versucht die GI zwischen den Beteiligten zu vermitteln.

Die Ethischen Leitlinien der GI

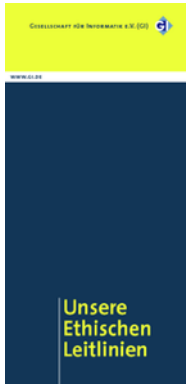
GESELLSCHAFT FÜR INFORMATIK

1994: Die ersten Ethischen Leitlinien

2004: Überarbeitung der Ethischen Leitlinien

2018: Erneute Überarbeitung

https://gi.de/ueber-uns/organisation/unsere-ethischen-leitlinien/



INFORMATIK BEWEGT,
INFORMATIK IST ZUKUNFT,
WIR SIND INFORMATIK!

GESELLSCHAFT
FÜR INFORMATIK

Berlin/Bonn, Juli 2018

GI.DE

Data modeling

Wiederholung: Software Engineering

Baupläne (Architektur) und Entwurfsregeln (Architekturkonzept)
Vorsicht: Statt *Architekturkonzept* sagt man oft mit *Architektur*!

Hausbau **Regeln für Baupläne**	**Produktentwicklung** **allgemein**	**Wirtschafts-Informatik** **Regeln für IS-Entwicklg.**
Wie **entwirft** man Pläne?	**Modellierungs-Methoden**	Phasenkonzepte (software process) und Entwurfsebenen
Welche dem Bauablauf entspr. **Verfeinerungen** braucht man?	**Modell-Ebenen**	
Welche **Ansichten** braucht man?	**Modell-Sichten** **Modell-Aspekte**	Modellaspekt-Matrix
Wie **zeichnet** man Pläne?	**Modell-Notationen**	

Wiederholung: Multiperspektivität

	Static models	**Dynamic models**
Data models	**Data (structure) models:** data structure diagrams; entity-relationship models (ERM); UML class diagrams	**Information flow models:** information / data flow charts / diagrams; Structured Analysis (SA); UML use case diagrams
Function models	**Function structure models:** compositional function trees; Jackson trees	**Behavior / process models:** algorithms (functions); Nassi-Shneiderman diagrams; (control) flow charts; business process models; UML activity diagrams; (UML sequence diagrams)

Wiederholung: Phasenkonzepte und Entwurfsebenen

Main phase	**Subphase, model level**	**Methods (ex.)**
Analytic phase: problem analysis	Elicitation of the current state of the org	Systems analysis, Reverse engineering
	Analysis of the current state of the organization	Requirements engineering, OOA
	Design of the planned state of the org. (sociotechnical IS)	Requirements engineering, BPM
	Design of the business concept of the IT system (technical IS)	Reference mod., test case description
Synthetic phase: IT system develop-ment	Design of the techn. concept of the IT system independ. of developm. tool	OO design, design patterns
	… depending on development tool	Unit tests
	Programming	Coding conv., agile programming
	Test	V model tests

Data modeling – Model levels

Auf den unmittelbar vorangehenden Folien geht es um eine Wiederholung aus Software Engineering, aus dem Sie die Termini "analytische Phase" und "synthetische Phase" brauchen.

Bemerkung: Im Folgenden werden wir von der analytischen und synthetischen Datenmodellierung sprechen. Die beiden Termini haben nichts mit der analytischen und synthetischen Phase im Software Engineering zu tun!

Für Ihr Verständnis von Datenmodellen ist sehr wichtig, dass es nicht einfach nur Datenmodelle gibt, sondern
Datenmodelle auf unterschiedlichen Modellebenen mit unterschiedlichen Modellzwecken.
Diese Modelle können sehr verschieden aussehen, da es abhängig vom Modellzweck unterschiedliche Modellierungsziele und Modellierungskriterien gibt.

Wenn Sie mit irgendwelchen Internetquellen oder Lehrbüchern arbeiten, dann wird der Unterschied zwischen verschiedenen Modellzwecken häufig nicht gemacht, und es entstehen furchtbare Modelle, die gleichzeitig verschiedene Modellzwecke erfüllen sollen, was natürlich nicht geht.

Im Folgenden unterscheiden wir konzeptuelle und logische Datenmodelle.
Wir fokussieren uns auf konzeptuelle Datenmodelle.

Data modeling – Model levels

Introductory example: customers → orders → order lines ← products

1 Conceptual model / Information model

(information-relevant level – **analytic phase**)
Goal: understandable for domain experts ("users"), easy to modify,
mathematically optimized, IT independent,
stable basis for a logical / implementation model

controlled redundancy (no modification anomalies):
– **basic attributes** only (no reconstruction possible if deleted)
– 3NF with foreign keys

primary data, domain data

This is the focus of our DB course.

2 Logical model / Implementation model

(implementation-relevant level – **synthetic phase**)
Goal: performance optimization for reports in an application area

uncontrolled redundancy (danger of inconsistency):
– **derived attributes** (in addition to, derived from basic attributes)
  data aggregation (Datenverdichtung)
– horizontal and vertical fragmentation
– denormalization (parts in 1NF or 2 NF, conditional relationships)

secondary data (in addition to primary data), administration data: **indexes**

The optimization methods often depend on the particularities of an individual DBMS which you best learn in a dedicated training.

This is not the focus of our DB course.

3 Internal model / Speichermodell

The form how DBMS store data (B trees etc.), not readable for humans

Data modeling – The two basic data modeling approaches

In data models, there are only two model layers: sets and attributes

This leads to two modeling approaches (not concurrent):
1 **synthetic method**: sets → attributes
First, you intuitively design tables and then, you look for suitable attributes.

2 **analytic method**: attributes → sets (normalization → 3NF)
You start with a list of attributes (UNF) and arrange them step by step in tables. This is called normalization.

The use of the two methods is not concurrent, but comparative with a mutual check of the results.

Bemerkung: Die Termini analytische und synthetische Datenmodellierung haben nichts mit der analytischen und synthetischen Phase im Software Engineering zu tun!

Data modeling – Atomization of attributes

In order to be as flexible as possible for reports, decompose the attributes into minimal meaningful units (kleinste bedeutungstragende Einheiten) from the very beginning – before starting normalization!
For this purpose, you have to use the knowledge about an application area.

This process is called **atomization** of attributes

(The word *atomicity* is also used in the context of transactions; next slide.)

Manche Internetquellen verwenden in völlig irreführender Weise das Wort Atomarität/Atomizität auch bei der 1NF. Dort sprechen wir aber eindeutig von Wiederholgruppen. Wiederholgruppen haben mit der Zerlegung von Attributen (bspw. Adresse) nicht das Geringste zu tun.

Example:
Decompose address
into country code, postal code, place, street, house number

Data modeling – Remark on transactions in relational databases

ACID principle

Atomicity: execution of transactions “either all or nothing”

Consistency: a transaction transforms a valid state into another valid state

Isolation: independency of transactions from one another
leads – in spite of parallel execution – to the same result
which would be obtained using serial execution

Durability: permanent storing of the results of transactions even in the case of power loss and crashes (non-volatile memory)

1 Mathematical modeling of sets (nodes)

Primary key

A primary key is an attribute (simple; einfacher Primärschlüssel) or
a group of attributes (compound; zusammengesetzter Primärschlüssel)
whose values uniquely identify the lines (entities) of a table (entity type).
This feature is called entity integrity.

The description of the primary key has to fit the description of the entity type.
(Die Namen der Primärschlüsselattribute müssen zum Namen der zugehörigen Tabelle passen.)

Bemerkung: Primärschlüssel sind heutzutage alphanumerisch.
Man verwendet leicht merkbare mnemotechnische Kürzel, z.B. für die Fakultäten der TH: IN, BW
Numerische Primärschlüssel (autoincrement integer) nur in folgenden Fällen:

- Lückenlos aufsteigender Nummernkreis (z.B. Rechnungs-, Buchungs-Nr.)
- Zur Bestimmung einer Prüfziffer (z.B. mod 11)
- Artifizieller Zähler (z.B. Statistik-Daten)

1 Mathematical modeling of sets (nodes)

Candidate key (Schlüsselkandidat)

A candidate key is an attribute (simple) or a group of attributes (compound) whose values uniquely identify the lines (entities) of a table (entity type). That is, candidate keys can work like primary keys.

For each table of a model, you have to choose one unique primary key from all of the possible candidate keys.

In everyday life, candidate keys don't play a big role.

Example: An invoice position table has three candidate keys:
PositionNr
InvoiceNr, ItemID
InvoiceNr, LineNr

1 Mathematical modeling of sets (nodes)

Relational databases are called relational as the tables are described in the form of *n*-dimensional mathematical **relations**,
and not – as you may read here and there – as they focus on relation(ship)s between tables.

The following concepts are only explained in class.

Mathematical concept of a relation

Subset of the cross product of 2 or more factor sets

2-dimensional in contrast to the concept of a function

n-dimensional

(EE_Slides_9_Meta, **Nodes**)

1 Mathematical modeling of sets (nodes)

There are four database terminologies depending on four different backgrounds.
In everyday life, you often switch between these four terminologies
so that you have to keep the entire terminological table in mind.

File management	Entities	Database tables	Algebra
File	Entity set	Table	Relation
File structure	Entity structure	Table structure	Relation structure
Data record	Entity	Row, line	Tuple
Data item	Attribute	Column	Factor (set)
Data item value	Attribute value	Elementary / cell value	Component value
Number of data items	Number of attributes	Number of columns	Degree (number of factors)
Number of data records	Number of entities	Number of rows	Cardinality (number of tuples)

2 Mathematical modeling of relationships (arcs) between sets

2.1 General relation of degree 2: many-to-many relationship

Example:
A Student can attend many courses.
A course can be attended by many students.

In the following, we will concentrate on two goals:

1 eliminate many-to-many relationships, focus on **functional relationships**
Eine n:m-Beziehung (relational) wird durch zwei funktionale Beziehungen und eine Zuordnungstabelle aufgelöst.

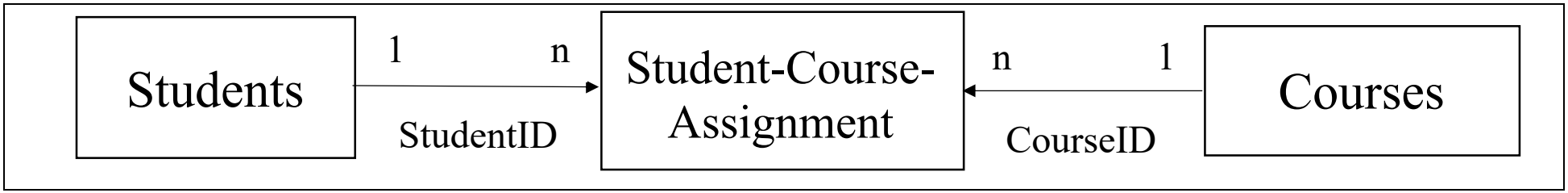
2 eliminate injective relationships, focus on **one-to-many relationships**
1:1-Beziehungen und 1:c-Beziehungen werden auf konzeptueller Ebene in jeweils einer Tabelle zusammengefasst.
Schauen Sie sich dazu auch meinen Foliensatz "Meta-models of information systems modeling approaches an", S. 12-14.

2.2 Transition to functional relationships: one-to-many relationship

A many-to-many relationship can always be replaced by two one-to-many relationships and an assignment table.

This should always be done in conceptual models.

Example (continued from 2.1):



PK: StudentID	PK: StudentID, CourseID FK: StudentID FK: CourseID	PK: CourseID

Note: An attribute can be part of a PK and a FK at the same time!

2.3 Keys

Primary key (entity integrity)
Foreign key / reference key (referential integrity)
A reference key is an attribute (simple) or a group of attributes (compound) which is a primary key in another table.

A foreign key always appears in the table on the ‘many’-side of a one-to-many relationship.
The corresponding primary key is in the table on the ‘one’-side.

Note:
– an attribute can be part of a PK and a FK at the same time!
– there can be several FKs in one single table
– there can be compound FKs

2.4 Types of functional relationships

1. Let $y \varepsilon \mathbf{W}$.

Then there are three types of instance relationships:
(1) There is no $x \varepsilon \mathbf{D}$: $f(x) = y$ [customer without order]
(2) There is one $x \varepsilon \mathbf{D}$: $f(x) = y$ [customer with one order]
(3) There is more than one $x \varepsilon \mathbf{D}$: $f(x) = y$ [customer with several orders]

f: **D** (orders) → **W** (customers) (function)
orders ← customers (many-to-one relationship)

There are no further types!

2. Expansion to the entire domain

2.1 There is at least one instance relationship of type (3): non-injective

The following cases do not contain any inst. relationship of type (3).

2.2 There are only instance relationships of type (1): **D** = {} not interesting

2.3 There are only instance relationships of type (2): one-to-one relationsh.

2.4 There are only instance relationships of types (1) and (2): injective
Conditional relationship; 1:c with c ε {0; 1}
Ex.: customer (one invoice address) has either a different delivery address or not.

Functional relationships between two entity types can be classified as
1) one-to-many relationships (non injective)
2) one-to-one relationships (bijective)
3) conditional relationships (1:c with c ε {0; 1})

This classification is complete.

Relationships have to be verbalized in two directions.
Example: one-to-many relationship between customers and orders
1 in the 'many'-direction: One customer can have (0, 1 or) many orders
2 in the 'one'-direction: One order belongs to exactly one customer
(referential integrity)

Attention: one-to-many includes the case 0 on data record level,
e.g. a (new) customer without any order

2.5 Elimination of injective relationships

One-to-one and conditional relationships are injective (do not carry any additional information):
coincide in one database table.

Only one-to-many are not injective (carry additional information).

**In the field of conceptual data modeling,
only one-to-many relationships have to be considered.**

Modellierung der Referenzen bei 1:1- und 1:c-Beziehungen

Weil man diese beiden Beziehungstypen bei konzeptuellen Modellen praktisch ganz ausschließt, stellt sich dieses Problem nur bei logischen Modellen, die nicht Thema dieser Vorlesung sind.

Deshalb nur kurz:

1. Fall: Fragmentiert man eine DB-Tabelle vertikal, so haben die beiden neuen Tabellen, die ja zum gleichen konzeptuellen Entitätstyp gehören, das/ie gleiche/n Primärschlüssel-Attribut/e.
Dieser Fall ist trivial. Man braucht keine Fremdschlüssel.

Beispiel 1: Die Tabelle Kunden wird geteilt in Adressdaten und Finanzdaten, normalerweise 1:1.
(1:c kann vorkommen, wenn man von einem Kunden noch keine Finanzdaten kennt.)

2. Fall: Man will zwei verschiedene konzeptuelle Entitätstypen 1:1 oder 1:c miteinander verbinden.
Wie bei 1:n-Beziehungen (dort werden die zu einem PK-Wert gehörigen n-seitigen Entitäten über einen Index über den FK der n-seitigen Tabelle gefunden) genügt grundsätzlich ein einziger Fremdschlüssel in eine 1-Richtung.

Beispiel 2: Rechnungen und Mahnungen sollen 1:c verbunden werden. Dann ergänzt man in der c-seitigen Tabelle (Mahnungen) einen Fremdschlüssel, der auf die jeweilige Rechnung verweist.
Man kann die 1:c-Beziehung aber auch schon konzeptuell zu einer 1:n-Beziehung expandieren, wenn man zu einer Rechnung mehrere Mahnungen zulassen will.

Beispiel 3: Man will monogame, heterosexuelle menschliche Paare 1:1 modellieren, Entitätstypen Frauen und Männer.
Auch hier genügt (wie bei 1:n) ein einziger Fremdschlüssel in eine der beiden Richtungen.
Diese 1:1-Abhängigkeit könnte man übrigens auch zu n:m expandieren, wenn man menschliche Beziehungshistorien und polygame Beziehungen mit erfassen will.

Im Prinzip behandelt man im Datenmodell 1:1 und 1:c wie 1:n.
Das könnte aber bei Anwenderfehlern zu Problemen führen:
Sei A die Tabelle auf der „linken“ Seite der Beziehung. Dann könnte fälschlicherweise in Tabelle B bei verschiedenen B_IDs die gleiche A_ID stehen.
Das kann man durch wechselseitige Fremdschlüssel abfangen, die gleichzeitig die Funktion von alternativen Primärschlüsseln haben. Dadurch kann man Anwenderfehler, die das Verhältnis 1:1 oder 1:c zerstören würden, bereits bei der Eingabe verhindern.

2.6 Classification of types of relationships [meta-models]

1 numeric (cardinality)
2 syntactic (reference keys)
3 semantic (compositional, taxonomic)

2.7 Treatment of selected relational relationships

c:n → use a dummy on the ‘c’-side, then you get 1:n

c:c → different, depending on application; maybe expand to n:m

2.8 DBA-defined attribute descriptions

Example:
Goal: unique attribute descriptions in the entire data model
Use: table abbreviation plus semantic attribute description

Client_ID
In table Client: C_Client_ID
In table Orders: O_Client_ID

2.9 Notations for data models

Meine eigenen Notationen:
– PK durch Unterstreichen, FK durch ein Sternchen *
– 1:n-Beziehung durch Pfeil in n-Richtung mit Fremdschlüssel
– 1:n-Beziehung schließt auf Datensatzebene den Fall 0 mit ein

Wenn Sie die Notationsweise FK für Fremdschlüssel verwenden:
– unterscheiden Sie verschiedene FK in einer Tabelle durch FK1, FK2 etc.

The following notations are only explained in class.
Entity-Relationship Models
min max notation
Oracle diagrams
Krähenfuß-Notation
(EE_Slides_9_Meta, **Arcs**)

3 Normalization: Motivation

1 Bei fester UNF-Attributliste liefert die Normalisierung ein konzeptuelles Datenmodell (3NF) (vgl. oben Folien „Model levels“),
das mathematisch optimiert ist
das nur Basisattribute enthält
das von den Eigenheiten einzelner Modellentwickler unabhängig ist
das von der Wahl des UNF-Primärschlüssels unabhängig ist
(1/2NF können sich je nach UNF-Primärschlüssel unterscheiden)
das leicht änderbar ist
das eine stabile Basis für darauf aufbauende logische Datenmodelle bildet
das ein ähnliches Ergebnis wie intuitive synthetische Modellierung liefert

2 Analytische Datenmodellierung (Normalisierung) und synthetische Datenmodellierung ergänzen sich gegenseitig und werden in der Praxis kombiniert verwendet
(vgl. oben Folie „The two basic data modeling approaches“).

3 Normalization: Conditions starting from UNF

Die drei Normalisierungsbedingungen lauten:
- keine Wiederholgruppen (1NF)
- keine Schlüsselteil-Abhängigkeiten (2NF)
- keine transitiven Abhängigkeiten (3NF)

UNF-Attribute:

Die Liste der UNF-Attribute darf nur atomare Basisattribute enthalten (zerlegt in kleinste bedeutungstragende Einheiten).
Diese Liste darf während der Durchführung des Normalisierungskalküls keine Änderungen erfahren.
Es dürfen keine Attribute hinzugefügt (außer Fremdschlüssel bei neuen Tabellen), weggelassen oder zerlegt werden.

3 Normalization: Problems 1

1 Parallele Abhängigkeiten:
Bei jedem der drei Normalisierungsschritte werden Gruppen von Attributen, die die jeweils geprüfte Normalisierungsbedingung nicht erfüllen, gleichzeitig in neue Tabellen ausgeklammert.

2 Geschachtelte (nested) Abhängigkeiten:
Bei jedem der drei Normalisierungsschritte sind die neu entstandenen Tabellen nochmal auf die gerade geprüfte Normalisierungsbedingung zu prüfen und daraus ggf. weitere Tabellen auszuklammern.
Geschachtelte Abhängigkeiten werden in mehreren Schritten aufgelöst.
Für jede Schachtelungsebene entsteht ein neues Datenmodell.

3 Normalization: Problems 2

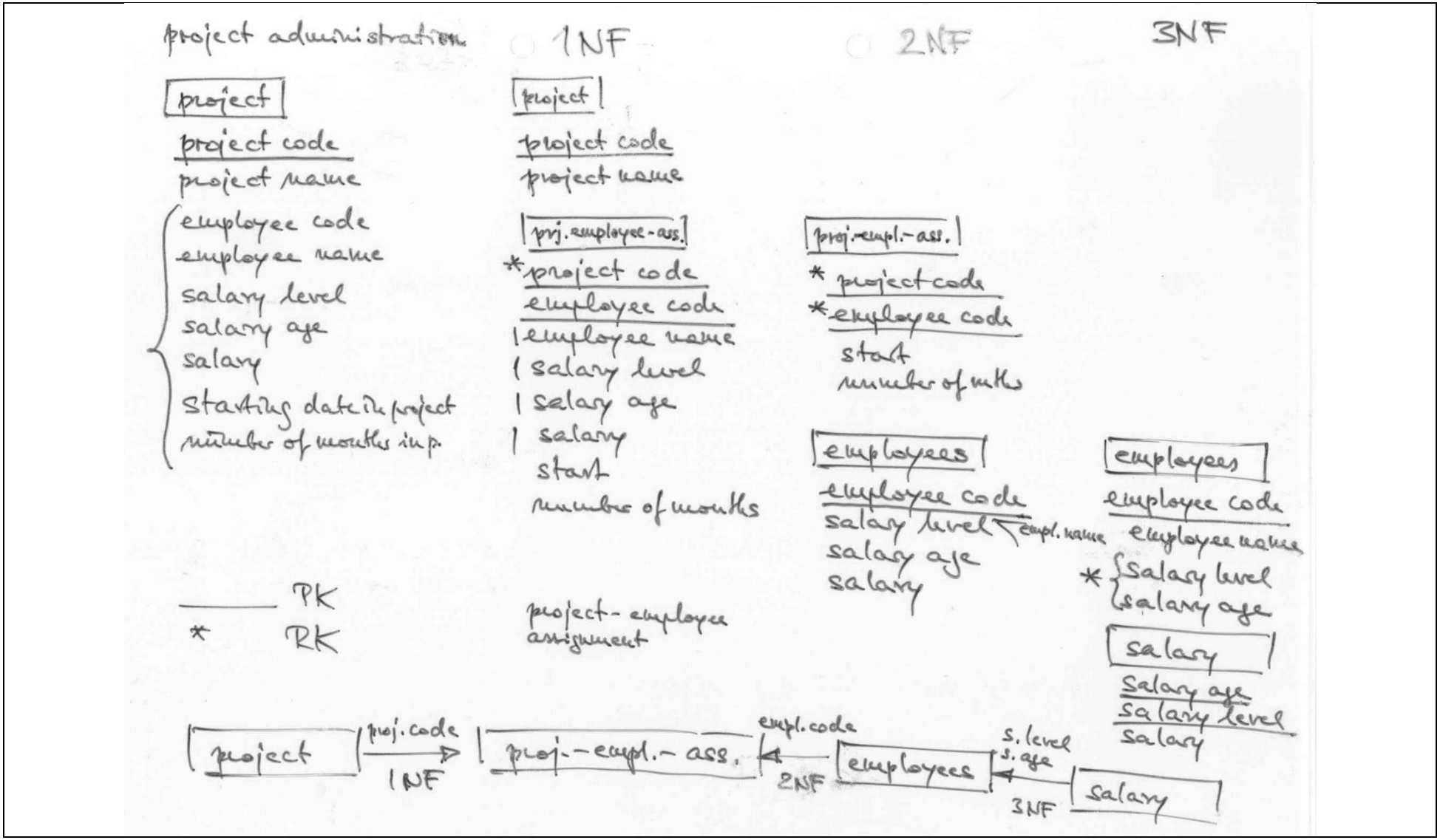
3 Hat man ausgehend von einer UNF die drei Normalisierungsschritte ausgeführt, so ist das Ergebnis nicht unbedingt schon ein 3NF-Modell. Ein Modell liegt erst dann in 3NF vor, wenn alle Tabellen in 3NF sind. Das erkennt man durch Überprüfung der drei Normalisierungsbedingungen für jede einzelne Tabelle.
Ist das entstandene Modell noch nicht in 3NF, beginnt man in einem zweiten Durchlauf von vorne mit dem ersten Normalisierungsschritt.

4 Das Ergebnis des Normalisierungskalküls ist immer ein wegeindeutiges Modell. Das entspricht häufig nicht der Wirklichkeit in Organisationen, so dass ggf. Beziehungen und Fremdschlüssel in einer manuellen Optimierung ergänzt werden.

5 Normalisierung ist nur für kleinere Modelle (10 Tabellen) gut geeignet. Größere Modelle zerlegt man in Teilmodelle, normalisiert diese und integriert sie zu einem Gesamtmodell.

3 Normalization: Example – Project administration

(only tables with changes are mentioned)



3 Normalization 0

UNF

List of attributes (atomized); need not be structured

Information-relevant (basic) attributes
Not: implementation-relevant (derived) attributes

Choose “suitable” one (or more) attribute(s) as UNF key
Note: different suitable UNF keys lead to the same 3NF.

Choose a name for the UNF table
Note: UNF key name has to fit UNF table name.

Result of mathematical normalization in 3NF:
unique-path graph

3 Normalization 1

1NF

Find repeating groups (Wiederholgruppen) of attributes.
  Choose a fix value of the UNF key and
  check whether the other attributes can have more than one value.
Remove them to subordinate tables.

Copy UNF key to subordinate tables as foreign key.
Determinate suitable primary keys of subordinate tables.

Note: new table in the direction of a one-to-many arrow

Es sind solche Attribute auszuklammern, die in Bezug auf einen festgehaltenen Wert (!!!)
des UNF-Schlüssels verschiedene Werte annehmen können.
Manche Internetquellen verwenden in völlig irreführender Weise das Wort Atomarität/Atomizität
(Zerlegung der Attribute in kleinste bedeutungstragende Einheiten) auch bei der 1NF.
Wiederholgruppen haben mit der Zerlegung von Attributen aber nicht das Geringste zu tun.

3 Normalization 1

Exceptions

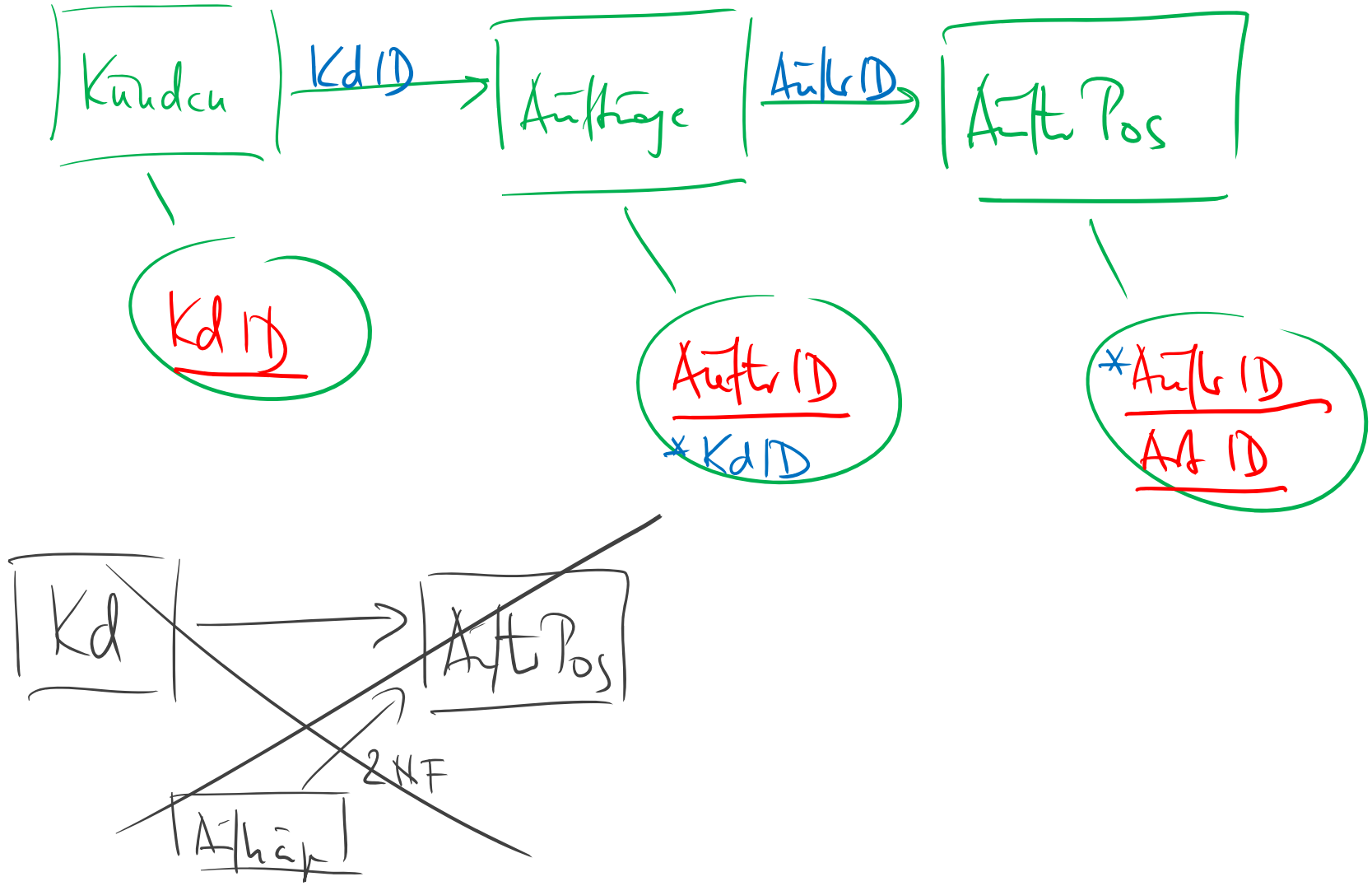
1 Nested (geschachtelt) repeating groups
isolation in at least two steps

2 Parallel repeating groups
simultaneous isolation

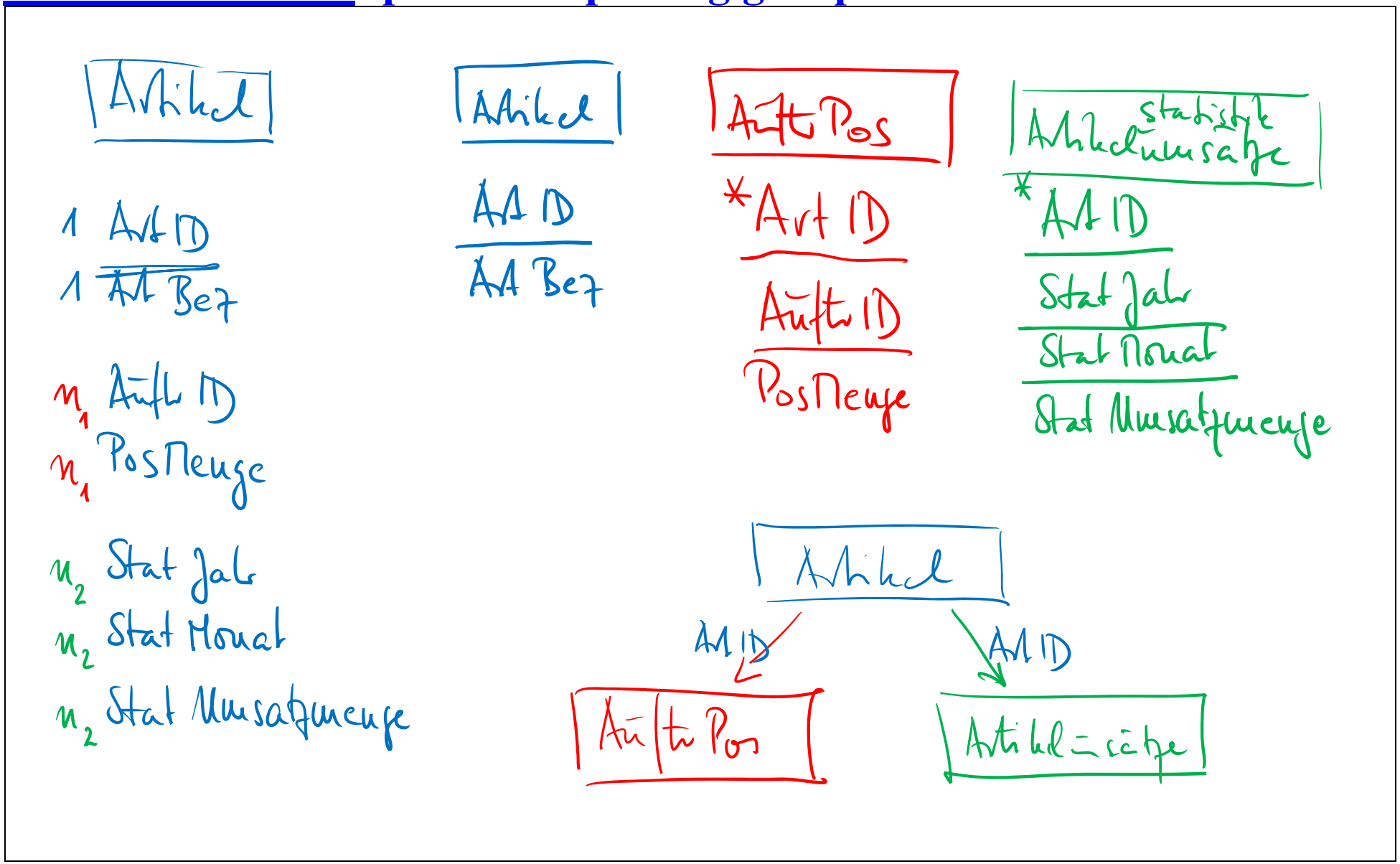
3 Dangerous thumb / heuristic rule:
Determination of primary key in 1NF:
Duplicated UNF key plus one or more attributes
Note: this rule does not apply in the case of functional (one-to-many) interdependencies between parts of the primary key.

Nested and parallel dependencies can also occur in the derivation of 2/3NF.

3 Normalization 1: nested repeating groups (also example for exception 3)



3 Normalization 1: parallel repeating groups



3 Normalization 2

2NF

Isolate attributes which functionally depend on one part of the primary key only (Schlüsselteil-Abhängigkeiten).

New table with the corresponding part of the original primary key as new primary key

Note: consider tables with compound primary keys only.

Note: new table against the direction of a one-to-many arrow

3 Normalization 2: nested functional dependencies

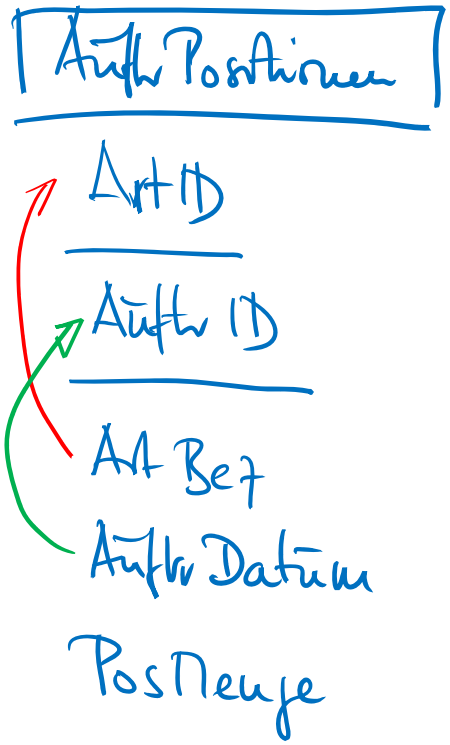
Nested dependencies only if the primary key consists of at least three parts

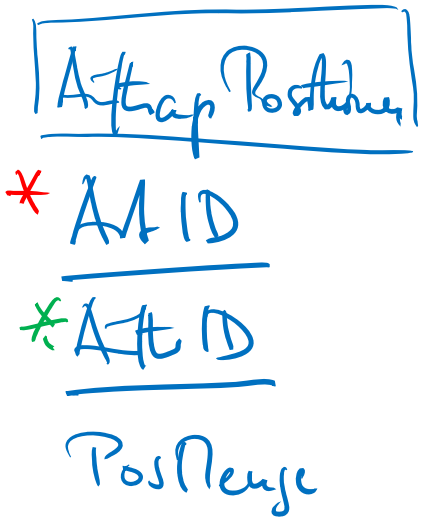
Auftragspositionen

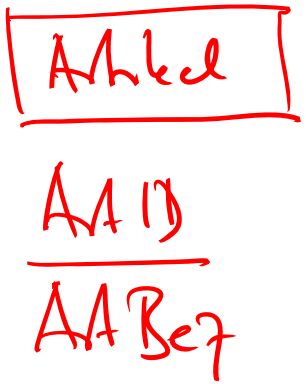
KdID
KdName
AuftrLfdNr
AuftrDatum
ArtID
(ArtBez)
PosMenge

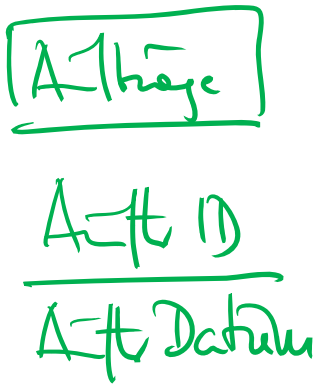
AuftrID = KdID + AuftrLfdNr

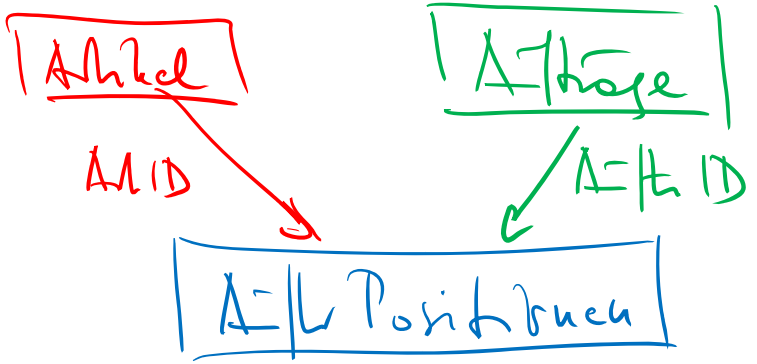
3 Normalization 2: parallel functional dependencies











Parallele Schlüsselkandidaten bei zus. ges. NF-Schema

3 Normalization 3

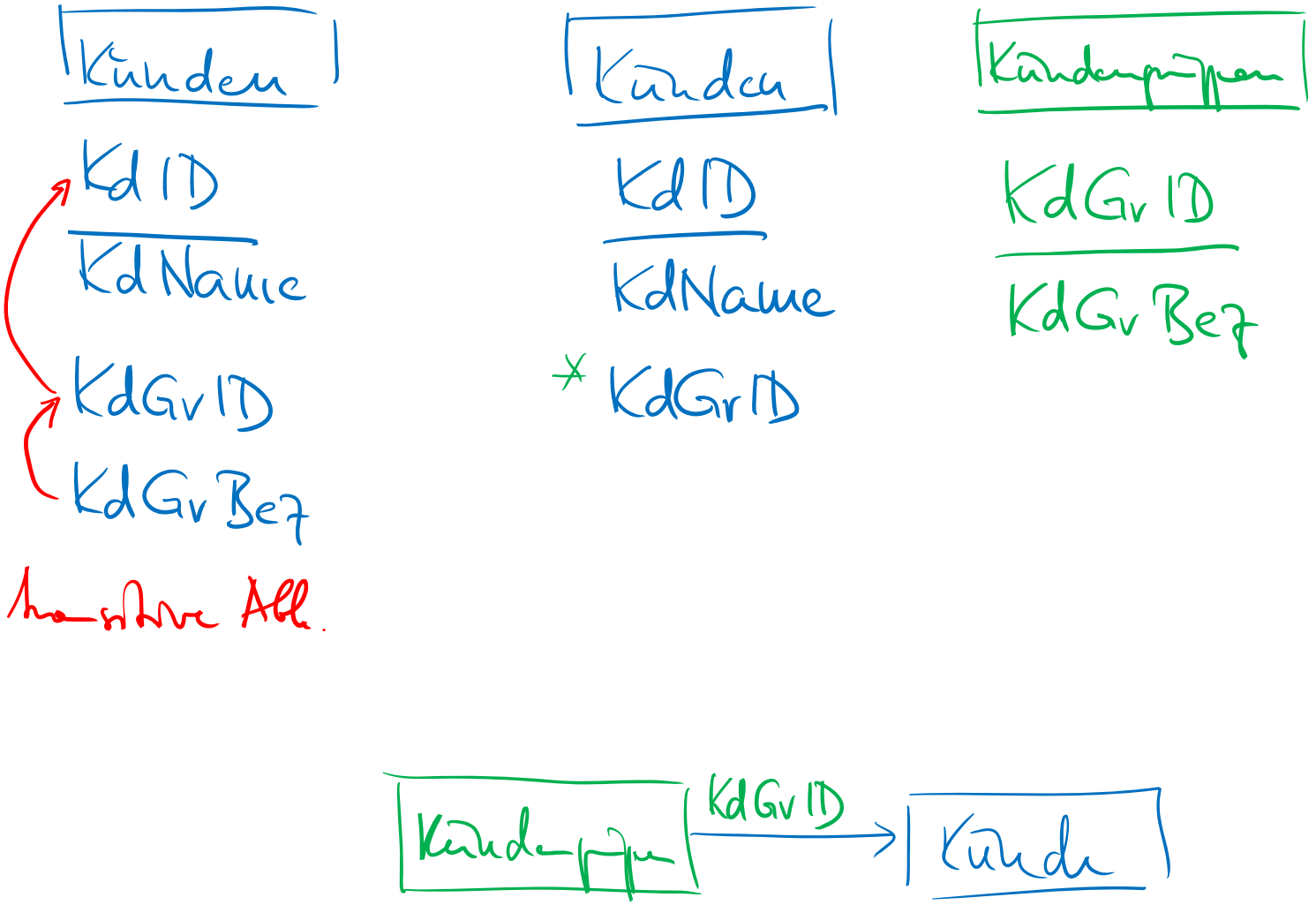
3NF

Isolate attribute(s) which do not directly depend on the primary key (or a part of it), but only via simple attributes.
Transitive dependency

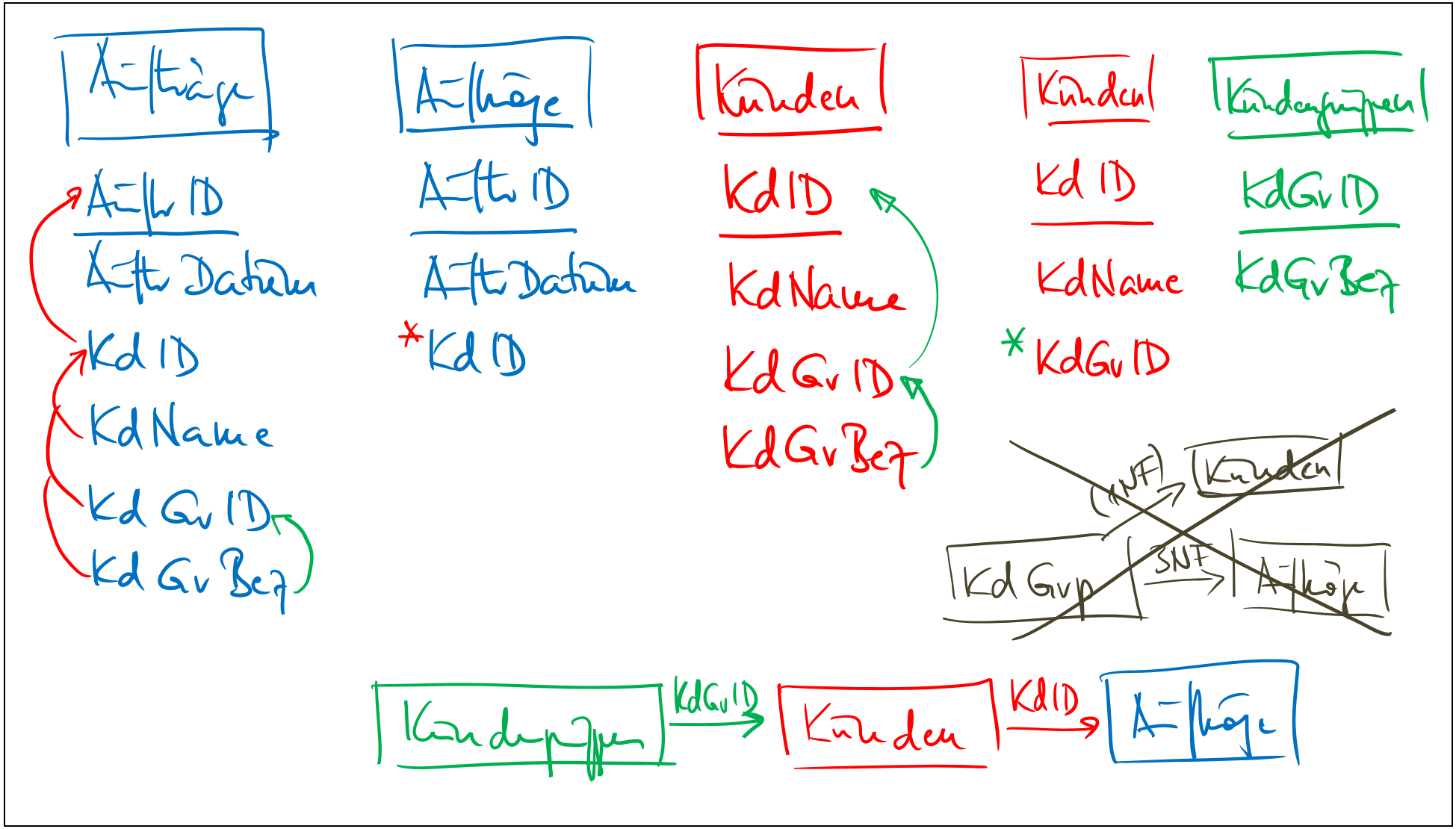
Note: new table against the direction of a one-to-many arrow

a ~ b and b ~ c => a ~ c

3 Normalization 3: transitive dependency



3 Normalization 3: nested transitive dependencies



3 Normalization 3: parallel transitive dependencies

Auftragspositionen

AuftrPosID
AuftrID
AuftrDatum
ArtID
ArtBez
PosMenge

AuftrDatum ist von AuftrPosID transitiv abhängig über AuftrID
ArtBez ist von AuftrPosID transitiv abhängig über ArtID

4 Normalization: particularities and first special cases

1 Stability of the UNF: no changes of attributes during normalization

2 Iteration of normalization

3 Result: unique-path graph

4 Ambiguity of decomposition:
most data models are not unique-path graphs (nicht wegeindeutig)
(order, delivery, order positions, delivery positions)
Then, several different ways of normalization are possible and correct.
Each of them leads to the same tables, but with different relationships.
– overlay different unique-path graphs (complicated)
– manually optimize one unique-path graph (quite simple)

4 First special cases

1 Transform a 2-dimensional table into a database table

Trivial: see students and courses in 2.2

2 Tree (bill of materials for one product)

Assume parent orientation towards product (root).
Each node without the root has got a parent node.

Data model:
Table ‘Nodes’ with a reflexive one-to-many relationship and
primary key: Node_ID
foreign key: Parent_node_ID

3 Network (bills of materials for several products)

Assume parent orientation towards the products
An arc is always defined by two nodes.

Tables ‘Nodes’ and ‘Arcs’ with two one-to-many relationships
and the foreign keys Parent_node_ID, Child_node_ID

Table ‘Nodes’
primary key: Node_ID
no foreign key

Table ‘Arcs’
primary key Parent_node_ID, Child_node_ID
foreign key 1: Parent_node_ID
foreign key 2: Child_node_ID

5 Reference models

Wenn man ein neues Modell für einen Anwendungsbereich entwickelt (egal ob Daten- oder Prozessmodell), gibt es dafür immer zwei Quellen:

– Beobachtungen und Interviews mit den Personen in diesem Anwendungsbereich

– Modelle, die man bereits kennt und die zu diesem Anwendungsbereich gewisse Ähnlichkeiten (Analogien) aufweisen.
Solche Modelle verwendet man als Referenzmodelle oder Vergleichsmodelle.
So kann ein Kunden-Auftragsverwaltungs-Modell als Referenzmodell für ein Lieferanten-Bestellverwaltungs-Modell dienen

Ein Modell ist also nicht schon von vornherein ein Referenzmodell, sondern wird erst durch seine Verwendung dazu.

Sind zwei Modelle ähnlich, so gibt es paarweise für jeden ähnlichen Entitätstyp einen Oberbegriff.
Beispiel: Kunde und Lieferant haben den Oberbegriff Geschäftspartner, Auftrag und Bestellung den Oberbegriff Vertrag.

Aus derartigen Oberbegriffen kann man ein generisches Modell ableiten, das seinerseits wieder als Referenzmodell dienen kann.

WICHTIG (Teilanalogien): Die Ähnlichkeiten (Analogien) zwischen einem Referenzmodell und einem neu zu entwickelnden Modell sind häufig nur teilweise (es muss nicht zu jedem Entitätstyp des Referenzmodells im neuen Modell einen analogen Entitätstyp geben und umgekehrt!!!), manchmal aber auch vollständig.

Ein in der Wirtschaftsinformatik verbreitetes generisches Referenzmodell, mit dem man viele Anwendungsbereiche zumindest teilweise abdecken kann, ist das folgende – immer aus der Sicht einer Organisation gedacht:

Geschäftspartnergruppen → Geschäftspartner →
→ Verträge → Vertragspositionen ←
← Vertragsgegenstände (physisch) ← Vertragsgegenstände (logisch) ← Vertragsgegenstandsgruppen

Die Grundgedanken bei diesem Typ Referenzmodellierung lauten zusammengefasst:

1. Es werden Vertragspartner zu Vertragsgegenständen in Beziehung gesetzt.
   Das geschieht über Verträge entweder mit genau einer oder mit mehreren Vertragspositionen.

2. Vertragspartner und Vertragsgegenstände können zu Gruppen zusammengefasst werden.

3. Falls Vertragsgegenstände als Exemplare individuell eindeutig identifizierbar sind, braucht man den Entitätstyp „physische Vertragsgegenstände“ (z.B.). Alle gleichartigen phys. Vertragsgegenstände fasst man zu einem „logischen Vertragsgegenstand“ zusammen. Ansonsten arbeitet man nur mit logischen Vertragsgegenständen.

Mit physischen Vertragsgegenständen meint man individuell eindeutig identifizierbare Vertragsgegenstände, z. B. Ausleihgegenstände, Buchexemplare einer Bibliothek mit Inventarisierungsnummer, Leihgeräte eines Vereins, Mietwagen, Gebrauchtwagen mit Fahrgestellnummer, große technische Anlagen.

Beispiel Bibliothek, Carsharing, Autovermietung, Verein
Ein log. Vertragsgegenstand ist etwa ein Buchtitel, zu dem eine Bibliothek mehrere Exemplare hat.
Eine Reservierung bezieht sich immer auf einen log. Vertragsgegenstand, weil es egal ist, welches Exemplar (phys. Vertragsgegenstand) von lauter gleichen Exemplaren Sie bekommen.
Die Ausleihe selbst bezieht sich dann auf einen phys. Vertragsgegenstand.
Bei einer Bibliothek muss man beide Ebenen von Vertragsgegenständen modellieren.

Beispiel Supermarkt / Baumarkt:
Wenn Sie eine Schachtel Schrauben kaufen, handelt es sich um einen n i c h t individuell identifizierbaren Vertragsgegenstand. Diese Schachteln tragen keine individuellen Codes.
Wenn Sie zwei Brote des gleichen Typs kaufen, dann sind sie nicht durch irgendeine ID unterscheidbar. Deshalb werden bei einem Baumarkt oder Supermarkt nur logische Vertragsgegenstände modelliert und keine physischen.

Beispiel Neuwagenkauf:
Sie bestellen einen Wagen eines bestimmten speziell konfigurierten Typs. Das ist ein log. Vertragsgegenstand.
Es werden aber mehrere Exemplare dieses Typs hergestellt. Sie bekommen eines davon mit einer bestimmten Fahrgestellnummer. Das ist dann ein phys. Vertragsgegenstand.
Die Bestellung bezieht sich also auf einen log. Vertragsgegenstand, die Rechnung auf einen phys. Vertragsgegenstand.

Beispiel Reisebüro:
Es gibt die Unterscheidung zwischen phys. Vertragsgegenstand (Durchführung einer Reise mit best. Programm zu einem best. Zeitpunkt) und log. Vertragsgegenstand (Reise mit best. Programm).

Vermietung: Ein Mietvertrag bezieht sich auf eine Wohnung als phys. Vertragsgegenstand.

5 Reference models

1. The two sources of model construction
– empiric: observation of an organization, interviews of the employees
– rationalistic: analogical transfer of reference models

2. Debtor model, creditor model, contract model

3. Examples of reference model structures in different business branches

4. Generic reference models and analogy based upon umbrella terms

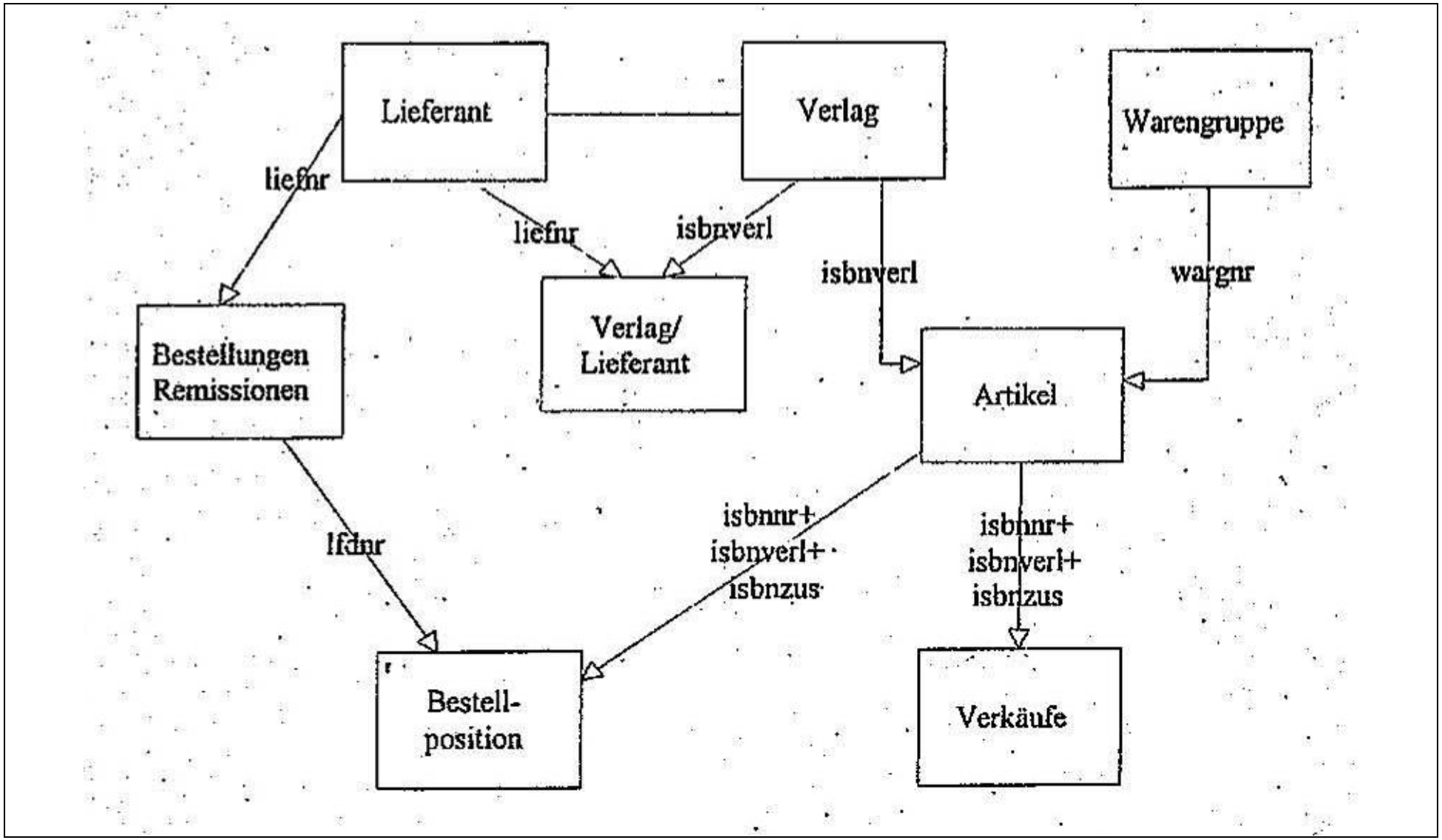
5. Partial analogies
– orders with one order position only
– individually identifiable items

(EE_Slides_7_Ratio)

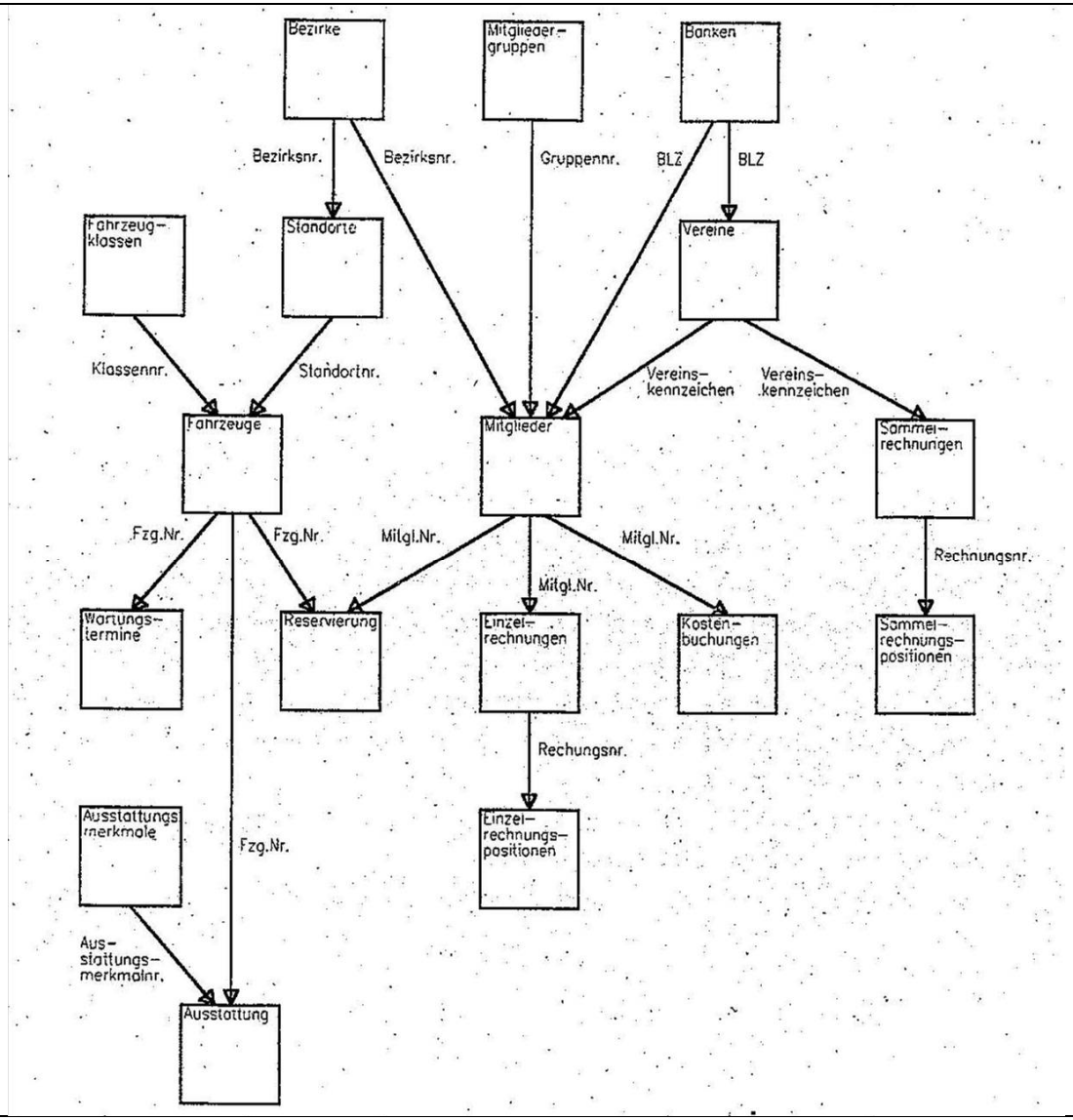
5 Reference models

creditor	debtor	umbrella terms generic model
supplier groups ↓ suppliers ↓ outgoing orders ↓ order lines ↑ raw materials ↑ material groups	customer groups ↓ customers ↓ incoming orders ↓ order lines ↑ products ↑ product groups	business partner groups ↓ business partners ↓ contracts ↓ contract positions ↑ contract items ↑ contract item groups

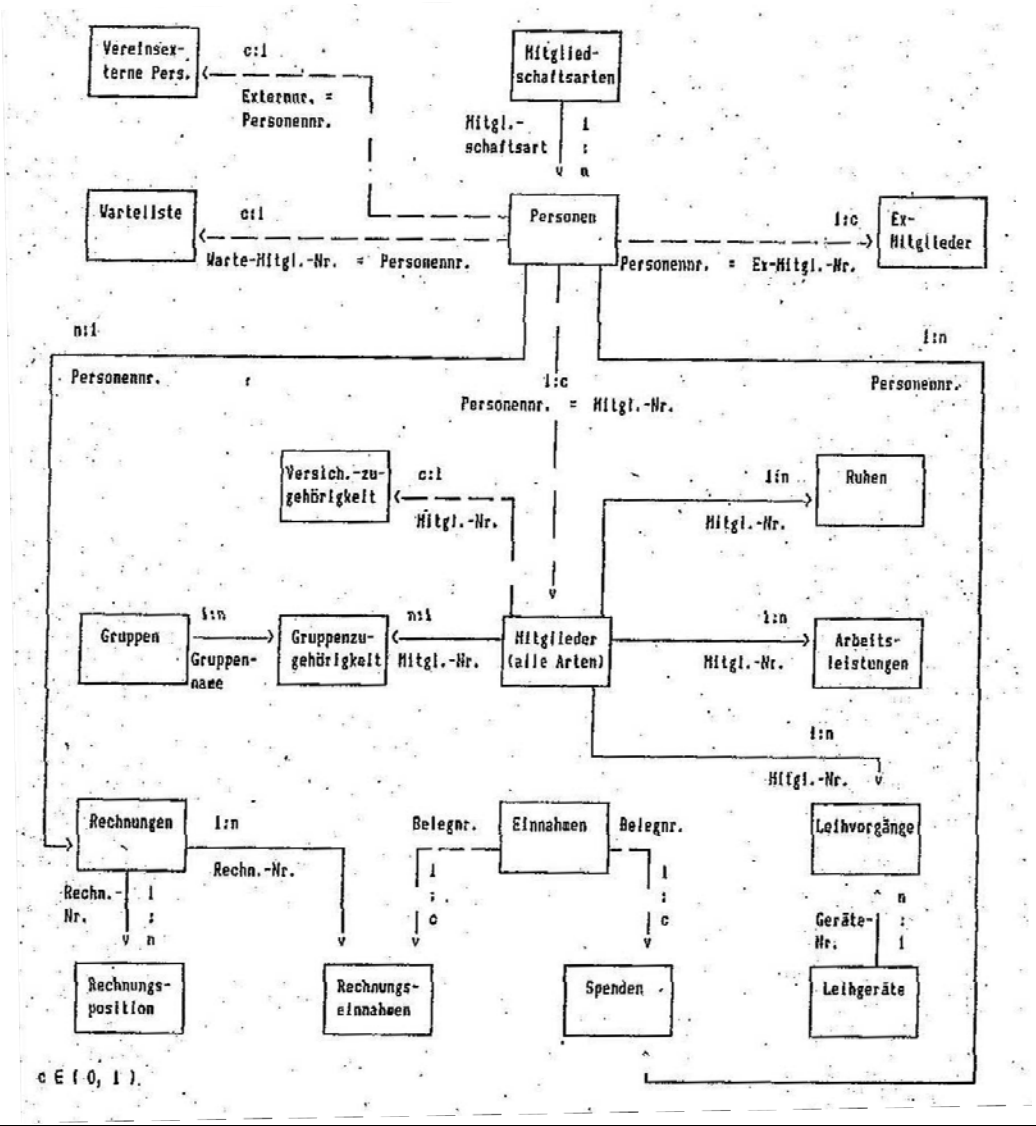
Comicladen


Lieferant
Verlag
Warengruppe
liefnr
liefnr
isbnverl
isbnverl
wargnr
Bestellungen
Remissionen
Verlag/
Lieferant
Artikel
lfdnr
isbnnr+
isbnverl+
isbnzus
isbnnr+
isbnverl+
isbnzus
Bestell-
position
Verkäufe

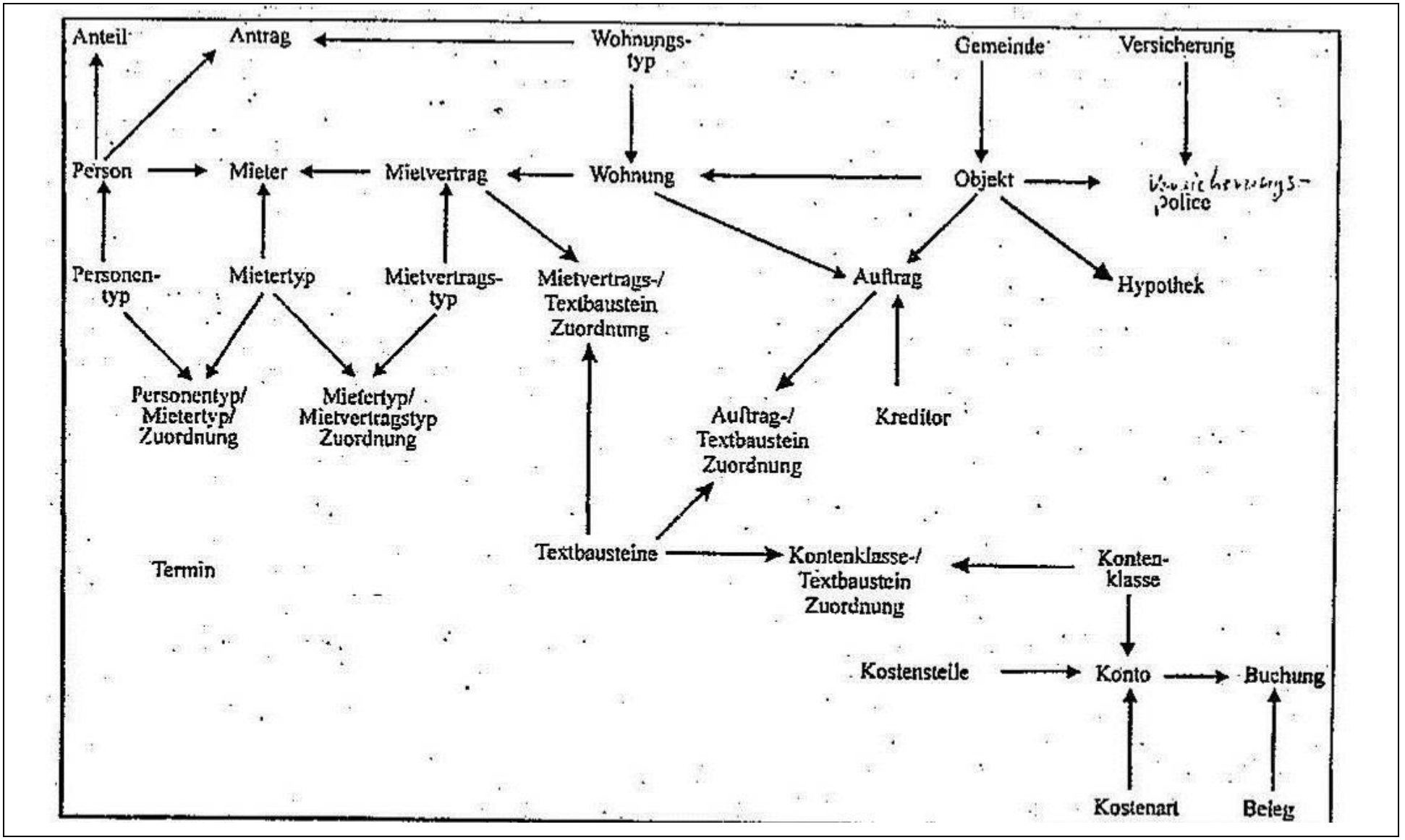
Car Sharing


Bezirke
Mitglieder-gruppen
Banken
Bezirksnr.
Bezirksnr.
Gruppennr.
BLZ
BLZ
Fahrzeug-klassen
Standorte
Vereine
Klassennr.
Standortnr.
Vereins-kennzeichen
Vereins-kennzeichen
Fahrzeuge
Mitglieder
Sammel-rechnungen
Fzg.Nr.
Fzg.Nr.
Mitgl.Nr.
Mitgl.Nr.
Rechnungsnr.
Mitgl.Nr.
Wartungs-termine
Reservierung
Einzel-rechnungen
Kosten-buchungen
Sammel-rechnungs-positionen
Rechungsnr.
Ausstattungs-merkmale
Fzg.Nr.
Einzel-rechnungs-positionen
Aus-stattungs-merkmalnr.
Ausstattung

Verein


Vereinsex-terne Pers.
c:1
Externnr. = Personennr.
Mitglied-schaftsarten
Mitgl.-schaftsart
1 : n
Warteliste
c:1
Warte-Mitgl.-Nr. = Personennr.
Personen
1:c
Personennr. = Ex-Mitgl.-Nr.
Ex-Mitglieder
n:1
Personennr.
1:c
Personennr. = Mitgl.-Nr.
1:n
Personennr.
Versich.-zu-gehörigkeit
c:1
Mitgl.-Nr.
1:n
Ruhen
Mitgl.-Nr.
Gruppen
1:n
Gruppen-name
Gruppenzu-gehörigkeit
n:1
Mitgl.-Nr.
Mitglieder (alle Arten)
1:n
Mitgl.-Nr.
Arbeits-leistungen
1:n
Mitgl.-Nr.
Rechnungen
1:n
Rechn.-Nr.
Belegnr.
Einnahmen
Belegnr.
Leihvorgänge
Rechn.-Nr.
1 : n
Geräte-Nr.
n : 1
1 : c
Rechnungs-position
Rechnungs-einnahmen
Spenden
Leihgeräte
c ∈ { 0, 1 }.

Vermietergenossenschaft



6 Special cases

6.1 Transform a 2-dimensional table into a database table √

6.2 Tree (bill of materials for one product) √

6.3 Network (bills of materials for several products) √

6.4 Coupling of three or more tables n:m:p etc. (e.g. time-table)

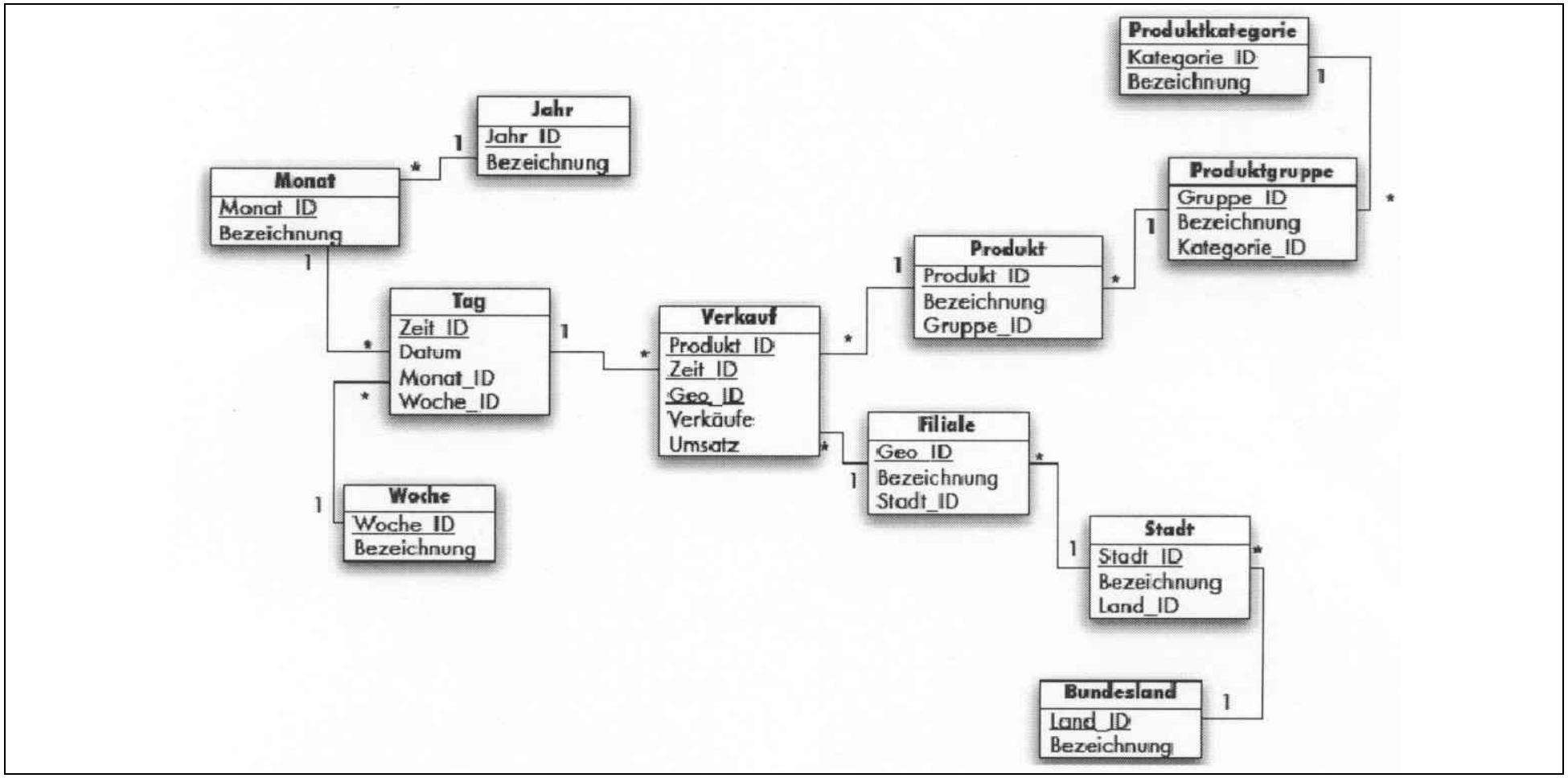
6.5 Customers, customer groups, customer categories

Only explained in class.

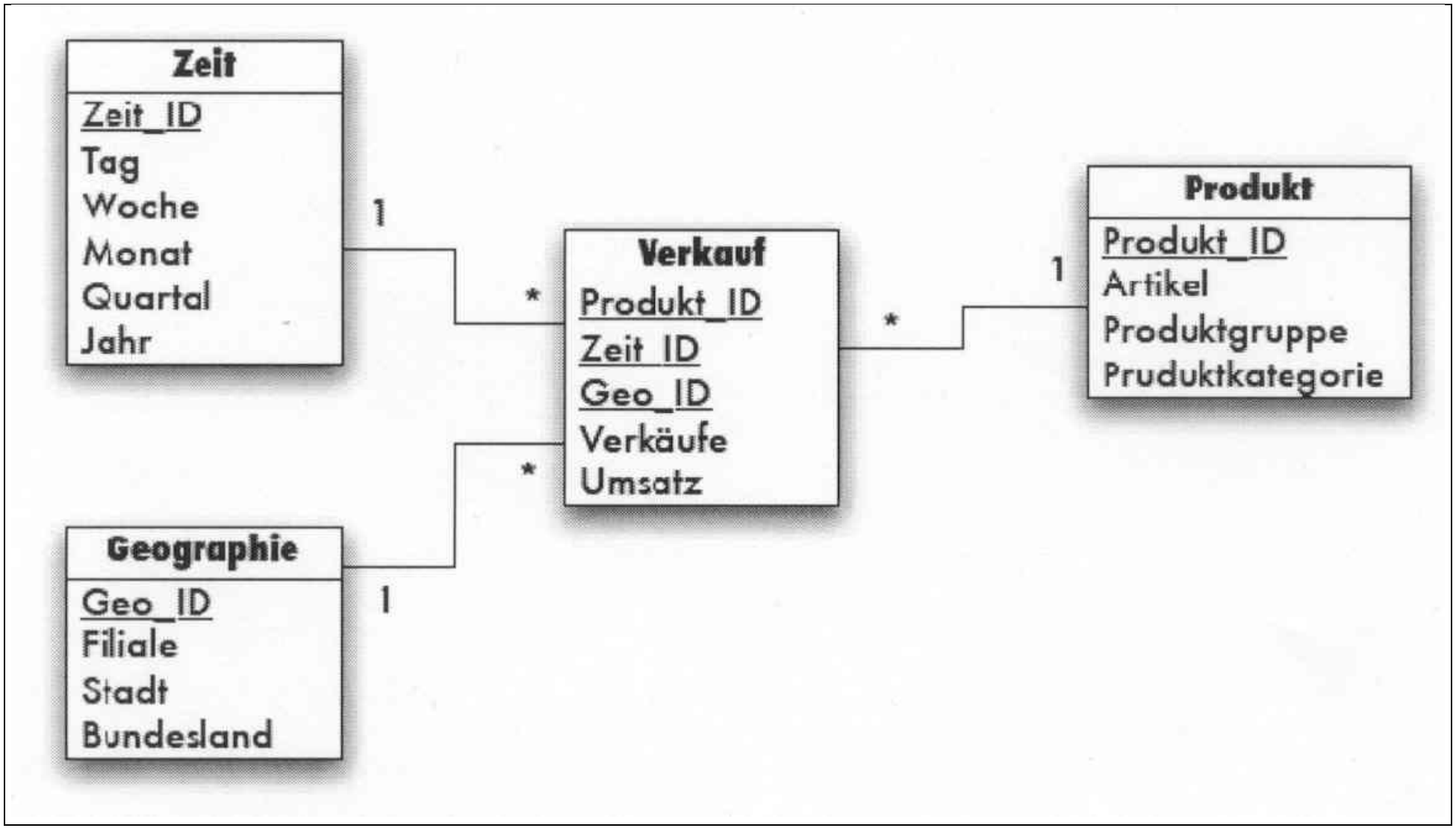
6 Special cases

6.6 Business Intelligence – Snowflake Schema (3NF, analytic phase)

Fact table(s) in the middle, dimension tables joined many-to-one



6.6 Business Intelligence – Star Schema (denormalized, synthetic phase)



6.6 Business Intelligence – Compacted snowflake schema

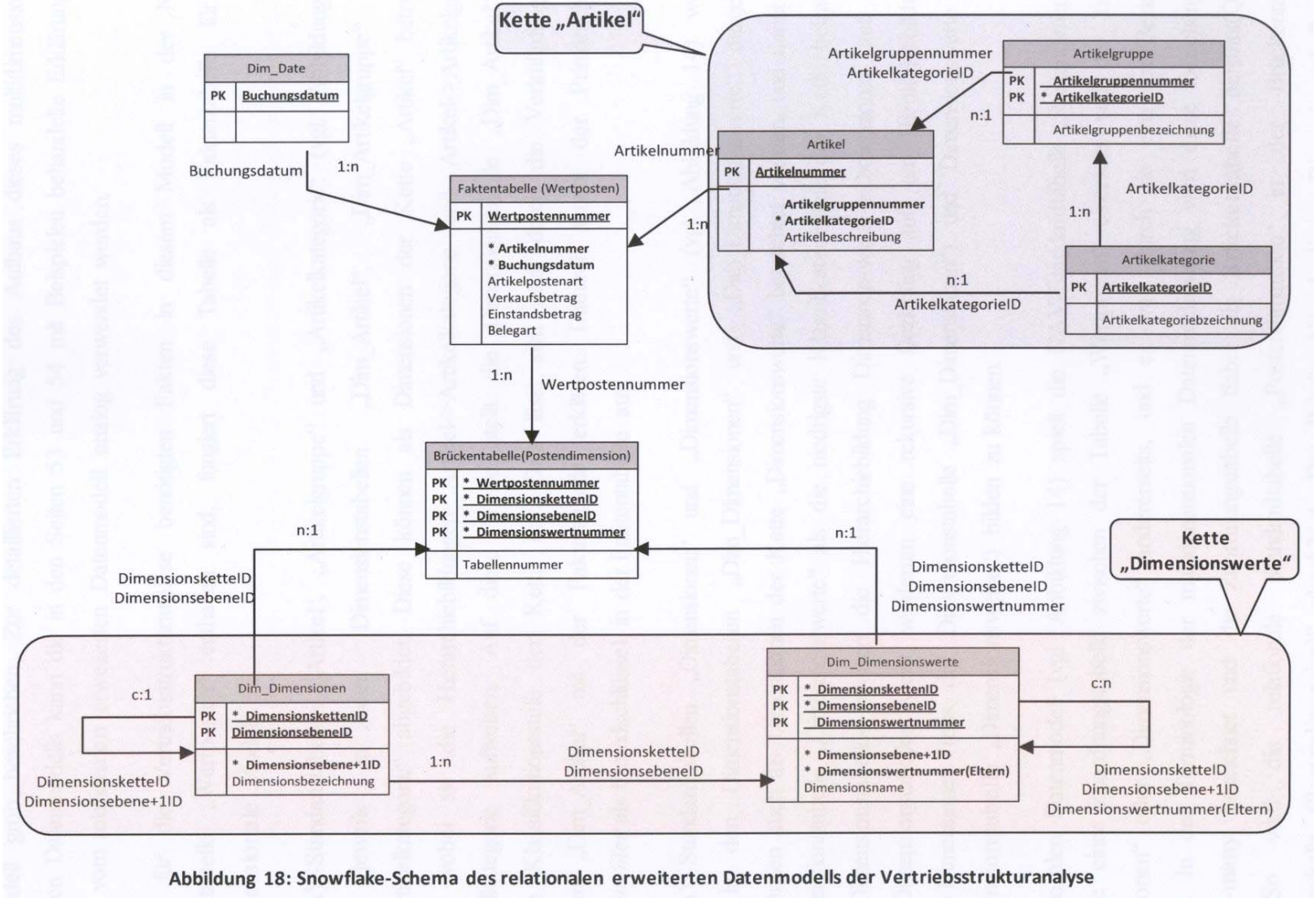
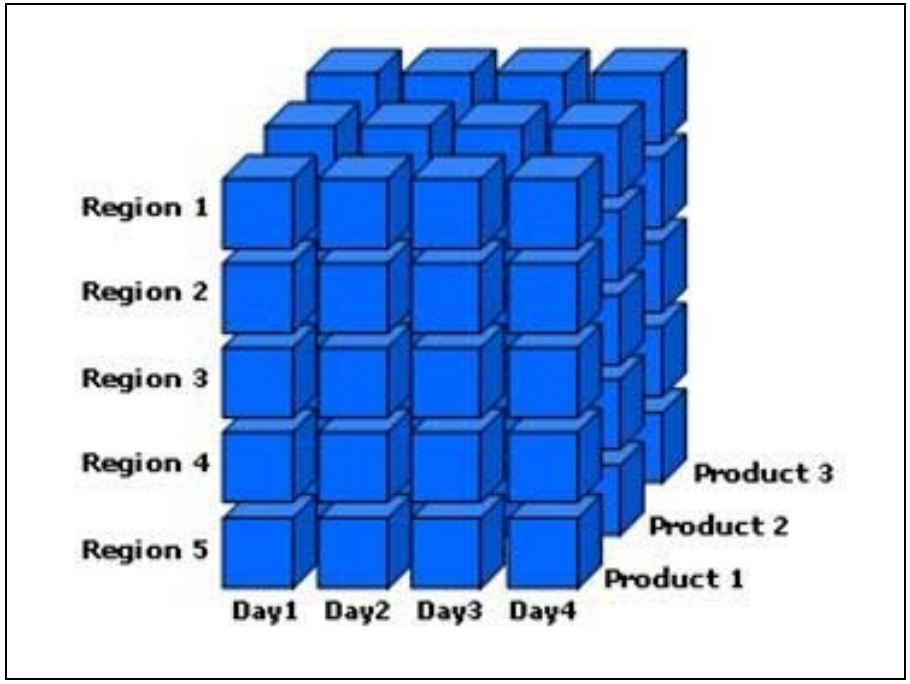


Abbildung 18: Snowflake-Schema des relationalen erweiterten Datenmodells der Vertriebsstrukturanalyse

6.6 Business Intelligence – Data Cube, Data Analysis

Online Analytical Processing OLAP



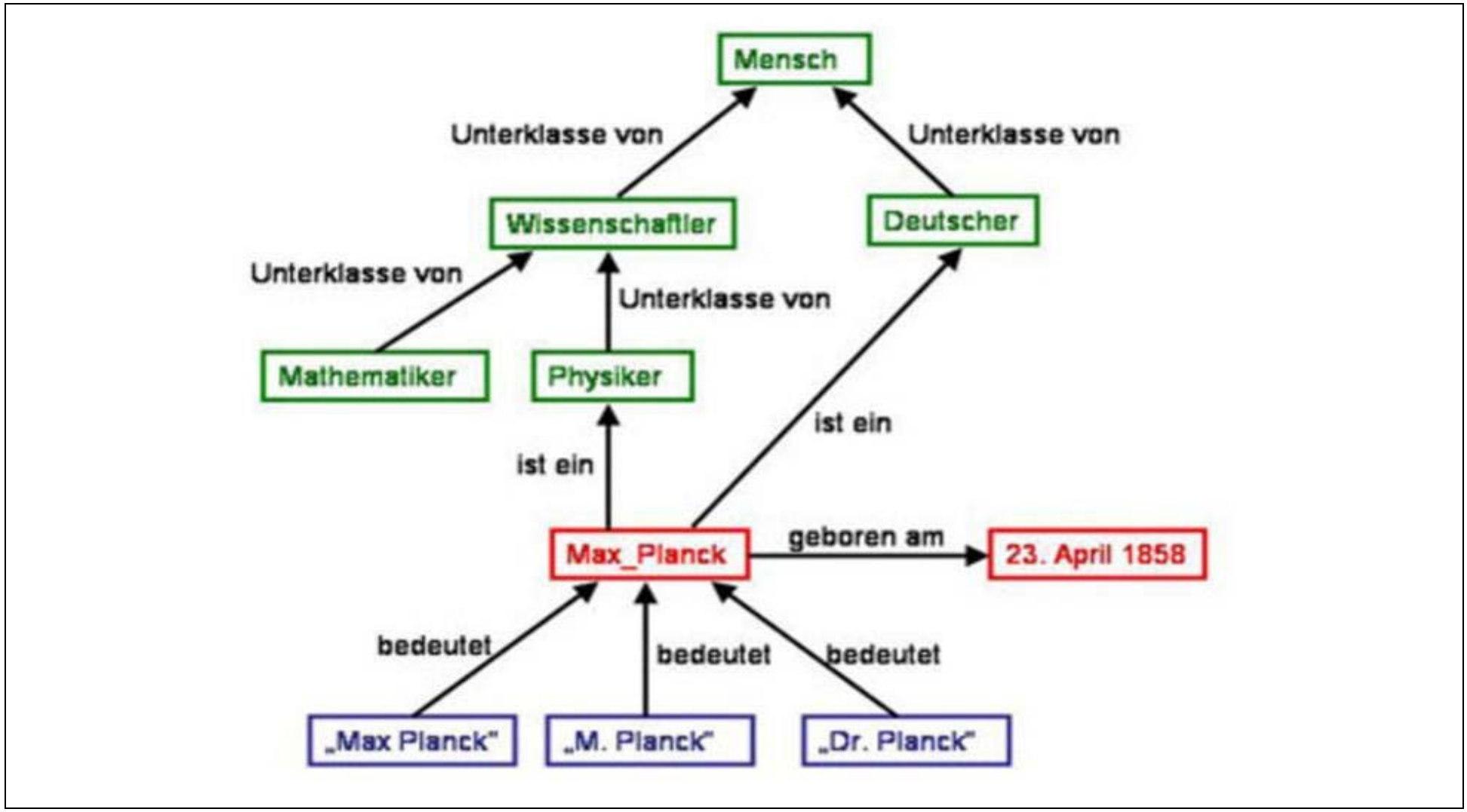
Slicing: Scheibe herausschneiden

Dicing: kleineren Würfel erzeugen

Roll-up: Herauszoomen

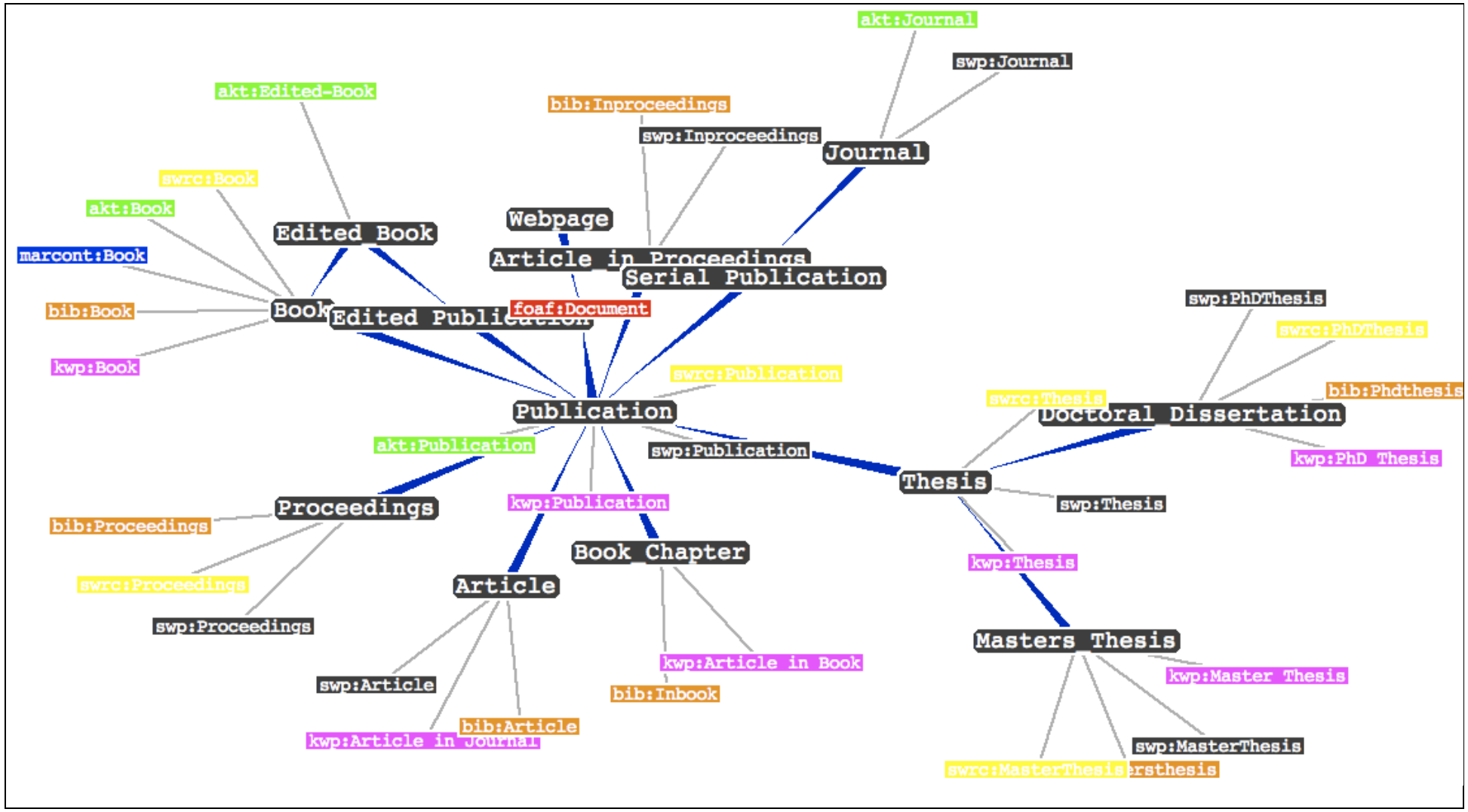
Drill-down: Hineinzoomen

6.7 Knowledge representation: ontology (semantic network)



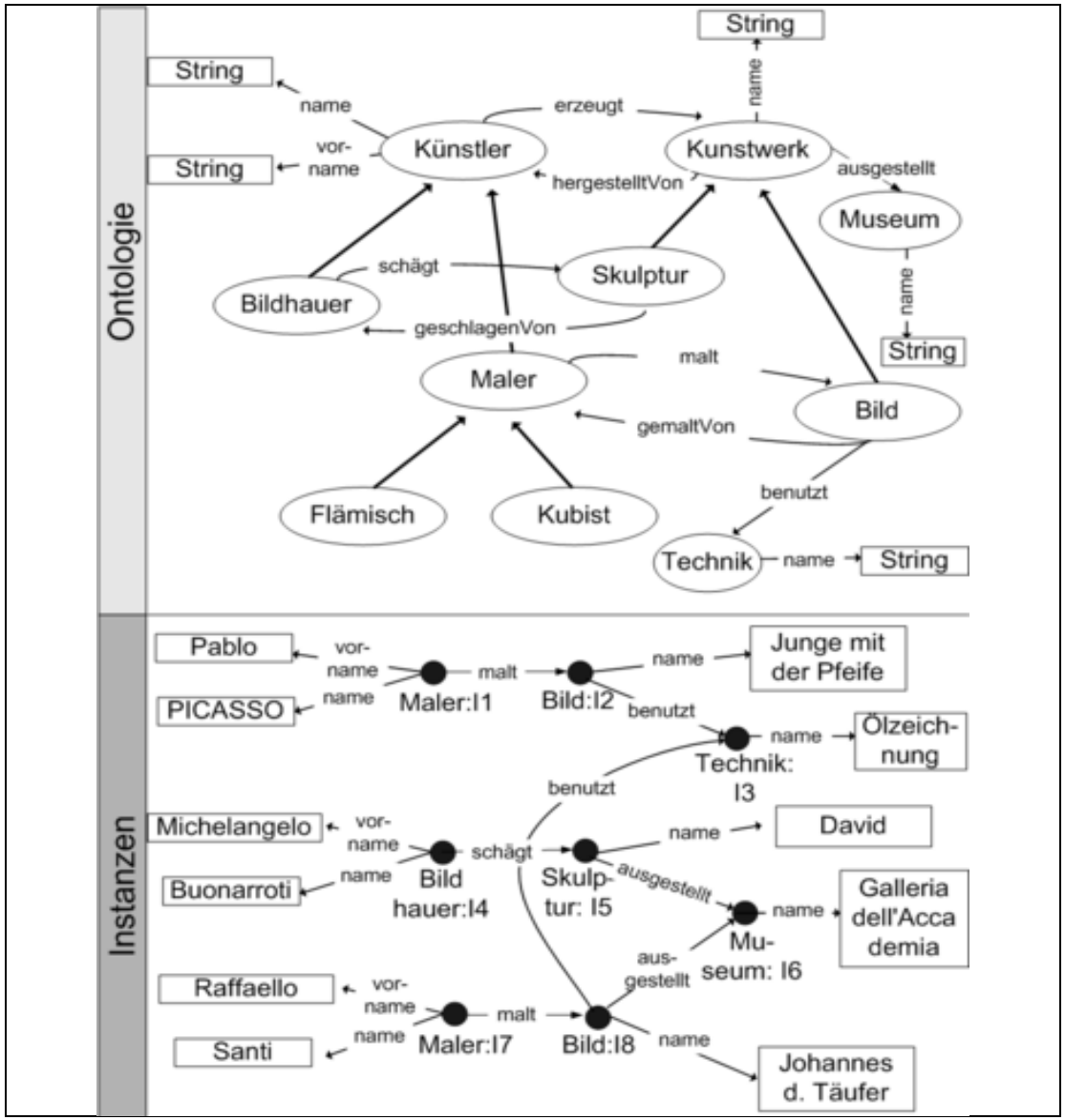
Ontologie über Personen

6.7 Knowledge representation: ontology (semantic network)



Ontologie über Veröffentlichungsarten

6.7 Knowledge representation: ontology (semantic network)



6.7 Knowledge representation: ontology representation languages

Resource Description Framework RDF:

XML-basiertes Rahmenwerk zur Beschreibung einzelner Ressourcen
Subjekt-Prädikat-Objekt-Tripel

RDF Schema RDFS

Web Ontology Language OWL: RDF-basiert

Web ontology:
OWL-Dokument, das eine Semantic-Web-Ontologie beschreibt

SPARQL Protocol And RDF Query Language SPARQL

SQL

SQL in der Lehrveranstaltung Datenbanken

Die Thematik SQL müssen Sie sich im Selbststudium erarbeiten.
Gegenstand dieser Lehrveranstaltung sind nicht bestimmte relationale DBMS (wie z. B. MS SQL Server, mySQL etc.) mit ihren Spezifika, sondern produktunabhängige Grundlagen!

Welche Teile von SQL müssen Sie lernen?
Es geht nicht um den vollen Umfang von SQL, sondern ausschließlich um den SELECT-Befehl!

Welche Bedeutung hat der SELECT-Befehl für die Klausur?
Aufgaben zum SELECT-Befehl bilden die Hälfte der Klausur!

Mit welchen Quellen können Sie sich einarbeiten?
– auf meiner Homepage: dieser Foliensatz ("Ergänzende Folien zu Datenbanken")
– auf meiner Homepage: "Einführung und Aufgaben zu SQL". Das ist eine Bachelorarbeit, die vor Jahren speziell zur Unterstützung des Selbststudiums von SQL geschrieben wurde
– im Prinzip jedes vernünftige Buch über SQL

SQL in der Lehrveranstaltung Datenbanken

Es gibt vier Dinge beim SELECT-Befehl, die Sie von Anfang an verstehen müssen:

1. Wenn man eine Aufgabenstellung (meistens sind das irgendwelche Auswertungen) in SQL übersetzen will, tut man das in zwei Schritten:
– zunächst vergegenwärtigt man sich die Aufgabenstellung in der Sprache der Mengenlehre (die Konzepte Vereinigung, Durchschnitt, Differenz von Mengen müssen Sie beherrschen!)
– erst dann formuliert man einen SELECT-Befehl

2. SQL ist eine Viertgenerationssprache (4GL, deklarative Sprache).
Jeder SELECT-Befehl wird aber von einem Interpreter prozedural (wie bei einer 3GL) abgearbeitet, d. h., Sie müssen sich über die Abarbeitungsreihenfolge der einzelnen Klauseln genau im Klaren sein (Folie „1.6 Abarbeitungsreihenfolge …“)

3. SQL ist keine orthogonale Sprache, d. h., Sie können eine Aufgabe syntaktisch häufig auf verschiedene Arten mit SELECT lösen.
Eine Klausuraufgabe kann von Ihnen also verlangen, eine Problemstellung mit einem bestimmten SQL-Syntax-Konzept zu lösen!

4. Abfragefilter: WHERE nur bei echten Satzfiltern, ON nur bei Join-Filtern.

1 SQL-Grundlagen

1.1 Relationale Grundoperationen

1. Selektion (Zeilenauswahl, Satzauswahl)
Beispiel: Alle Kunden in Nürnberg
WHERE-Bedingung: Satzauswahl, Satzfilter

2. Projektion (Spaltenauswahl)
Beispiel: Adressliste aller Kunden

3. Join (Verknüpfung, Verbund)
Theta-Verbund oder condition join
ON-Bedingung: Joinauswahl, Joinfilter
Beispiel: Auftragsdaten mit zugehörigen Kundendaten

WHERE- und ON-Bedingung sind nach ANSI-Standard zu unterscheiden.

1.1 Relationale Grundoperationen

4. Selektion und Projektion
Beispiel: Adressliste der Nürnberger Kunden

5. Join und Selektion
Beispiel: Alle Kunden- und Auftragsdaten der Nürnberger Kunden

6. Join und Projektion
Beispiel: Alle Auftragsdaten mit Kundennamen

7. Join und Selektion und Projektion
Beispiel: Auftragsdaten der Nürnberger Kunden mit Kundennamen

1.1 Relationale Grundoperationen: Satzfilter vs. Joinfilter

Ein Joinfilter (JOIN … ON) verbindet z w e i Tabellen
(meist über die Primärschlüssel-Fremdschlüssel-Beziehung)
Ein Satzfilter (WHERE) wählt Zeilen aus e i n e r Tabelle aus
(diese Tabelle kann selbst durch einen Join entstanden sein)

Beispiel: Liste der offenen Aufträge der Nürnberger Kunden
(AuftragsNr, AuftragsDatum, KdID, KdName)

```
SELECT AuftragsNr, AuftragsDatum, KdID, KdName
FROM Aufträge JOIN Kunden ON Auf_KdID = KdID
WHERE AuftragsStatus = ‘offen‘
AND KundenOrt = ‘Nürnberg‘
```

Man könnte Joinfilter auch mit WHERE formulieren, aber das wäre intransparent und entspräche nicht dem ANSI-Standard. Also verboten!

1.2 Sonderfälle des Join

1. [INNER] JOIN (kommutativ): Verknüpfung wo möglich, Default

2. LEFT / RIGHT [OUTER] JOIN (nicht kommutativ):
Inner Join plus nicht verknüpfbare Zeilen der Driving Table
Driving Table (unabhängig): alle Zeilen werden ausgegeben
Driven Table (abhängig): Zeilen nur ausgegeben, sofern Join möglich

Beispiel: Alle Kunden mit Aufträgen plus alle Kunden ohne Aufträge

Vorsicht: Anders als beim Inner Join, können WHERE (Satzfilter) und ON (Joinfilter) beim Outer Join unterschiedliche Wirkungen haben!
Eine WHERE-Bedingung in einem Outer Join
wird immer erst nach Erstellung des Outer Join ausgewertet und
kann so den Outer Join zerstören.
Besonders gefährlich: WHERE-Bedingung auf Attribute der Driven Table.

1.2 Sonderfälle des Join

3. Anti-Join oder negativer Join:
   nur die nicht verknüpfbaren Zeilen der Driving Table

Beispiel: Alle Kunden ohne Aufträge

```
SELECT KdID, KdName
FROM Kunden LEFT [OUTER] JOIN Aufträge
ON KdID = Auf_KdID
WHERE Auf_KdID IS NULL
```

Das geht auch ohne Join mit WHERE KdID NOT IN Subselect …
oder mit einer Mengendifferenz

Es gilt: Outer Join = Inner Join (mit ON) disjunkt vereinigt mit Anti-Join
→ dazu Mengendiagramm!

1.2 Sonderfälle des Join **(nach GROUP BY besprechen!)**

Problem-Beispiel für ON und WHERE beim OUTER JOIN

Alle Artikel, die in einem Statistikzeitraum geliefert wurden, und
alle Artikel, die in diesem Zeitraum nicht geliefert wurden, mit Mengen.

```
SELECT A_ArtID, A_ArtBez, SUM(L-Liefermenge)
FROM Artikel LEFT JOIN Lieferpositionen ON A_ArtID = L_ArtID
WHERE L_Datum BETWEEN StatAnf AND StatEnd
      OR L_Datum IS NULL
GROUP BY A_ArtID, A_ArtBez
```

Falsch, denn es gibt 3 Fälle:
1. u.a. im Statistikzeitraum gelieferte Artikel (L_Datum BETW … AND …)
2. gar nicht gelieferte Artikel (L_Datum IS NULL)
3. nur außerhalb des Statistikzeitraums gelieferte Artikel

→ bei diesen ist die WHERE-Bedingung nicht erfüllt

1.2 Sonderfälle des Join

Richtig:

```
SELECT A_ArtID, A_ArtBez, SUM(L_Liefermenge)
FROM Artikel LEFT JOIN Lieferpositionen ON A_ArtID = L_ArtID
AND L_Datum BETWEEN StatAnf AND StatEnd
GROUP BY A_ArtID, A_ArtBez
```

OUTER JOIN erzeugt Inner Join (mit ON-Bedingung) und
erweitert diesen um Anti-Join auf Outer Join.

1.2 Sonderfälle des Join

Oder klassisch:

```
SELECT A_ArtID, A_ArtBez, SUM(L_Liefermenge)
FROM Artikel JOIN Lieferpositionen ON A_ArtID = L_ArtID
WHERE L_Datum BETWEEN StatAnf AND StatEnd
GROUP BY A_ArtID, A_ArtBez
UNION
SELECT A_ArtID, A_ArtBez, 0 FROM Artikel
WHERE A_ArtID NOT IN
(SELECT DISTINCT L_ArtID FROM Lieferpositionen
WHERE L_Datum BETWEEN StatAnf AND StatEnd)
```

Inner Join disjunkt vereinigt mit Anti-Join
Mengenstruktur des Select-Ergebnisses für UNION angeglichen

1.2 Sonderfälle des Join

Ein äquivalentes Beispiel:

Alle Kunden, die (u. a.) in I/2019 Aufträge erteilt haben,
mit ihren Aufträgen und
alle Kunden, die in I/2019 keine Aufträge erteilt haben,
d. h. entweder noch gar keine Aufträge erteilt haben oder
nur Aufträge außerhalb des Statistikzeitraums I/2019 erteilt haben.

1.3 Gruppenbildung, Aggregatfunktionen, Gruppenauswahl

Gruppenbildung
ist eine Form der Datenverdichtung (engl. aggregation)
“Töpfe”, deren Etiketten bestimmte Wert-Tupel der GROUP-Attribute sind
z.B. GROUP BY KdID: jeder Wert der KdID (Kunde) bildet einen “Topf”

GROUP
Beispiel: Alle Kunden, die 2012 Aufträge erteilt haben (aus Auftragstab.)

Die Einschränkung auf 2012
funktioniert mit WHERE … oder mit HAVING YEAR(AuftragsDatum).
HAVING auf Gruppenattribute ist grundsätzlich erlaubt.
Aber es werden überflüssige Gruppen erzeugt (Performance!),
deren Irrelevanz ohne Aggregatfunktion
bereits an jedem einzelnen zugehörigen Datensatz erkennbar ist.
Deshalb besser mit WHERE filtern.

1.3 Gruppenbildung, Aggregatfunktionen, Gruppenauswahl

GROUP und Aggregatfunktion SUM
Beispiel: Alle Kunden mit jeweiligem Gesamtauftragswert in 2012

GROUP, SUM und HAVING
Beispiele: 1. Alle Kunden mit Gesamtauftragswert > 5000 € in 2012
2. Alle Nürnberger Kunden mit Gesamtauftragswert > 5000 € in 2012

Aggregatfunktionen ohne GROUP: Die gesamte Tabelle bildet eine Gruppe
Beispiel: Gesamtauftragswert über alle Kunden in 2012

HAVING kann mit den gleichen syntaktischen Bedingungsformen verbunden werden wie WHERE:
<, >, =, LIKE, (NOT) IN (Subselect), Tupelabfrage etc.
Aggregatfunktionen sind mit HAVING, nicht mit WHERE kombinierbar.

1.3 Gruppenbildung, Aggregatfunktionen, Gruppenauswahl

HAVING: Gruppenauswahl, -filter, Teilmengenauswahl
WHERE: Satzauswahl, -filter, Elementauswahl
ON: Joinauswahl, -filter

Aggregatfunktions-Attribute:
HAVING-Aggregatfunktionen brauchen nicht im SELECT zu erscheinen.
Aggregatfunktionen dürfen nur in SELECT und HAVING vorkommen!
Aggregatfunktionen dürfen in GROUP und WHERE nicht vorkommen!

Normale Attribute:
Jedes HAVING-Attribut muss auch ein GROUP-Attribut sein.
→ gilt nicht für Argumente von Aggregatfunktionen,
z.B. GROUP BY KdID HAVING SUM(Auftragswert) > 500 und 2.1 Nr. 2
Jedes SELECT-Attribut muss auch ein GROUP-Attribut sein.

1.4 Sonderthemen

Aliasing von Tabellen- und Attributnamen

DISTINCT
teilweise gleichwertig mit GROUP ohne SELECT-Aggregatfunktionen

Subselect
teilweise gleichwertig mit Join:
WHERE … (NOT) IN (Subselect)
HAVING … (NOT) IN (Subselect)
SELECT … FROM (Subselect)

1.5 Strukturierte Kommentierung

Benötigte Attribute (Ausgabeattribute, Attribute in Aggregatfunktionen, Gruppenattribute, Satzfilter-, Joinfilter-, Gruppenfilterattribute)
Benötigte Tabellen
Joinfilter (meist so formuliert: „Join der Tabellen A und B über ihre Primärschlüssel-Fremdschlüssel-Beziehung“ – es gibt auch andere!)
Satzfilter
Gruppierung
Gruppenfilter

Zusammengesetzte SELECT-Befehle können geschachtelte und/oder parallele Teilselects enthalten, die folgendermaßen zu kommentieren sind:
Obige Angaben für jeden Teil-SELECT,
dazu die Semantik jedes Teil-SELECTs (beschreibt dessen Ergebnis),
nicht aber für den äußersten SELECT eines Schachtel-SELECTs
(dessen Semantik ist gleich der Aufgabenstellung)

1.6 Abarbeitungsreihenfolge der Klauseln des SELECT-Befehls

vgl. Software Engineering: Gruppenverarbeitung (breakpoint analysis)

Input	
Verknüpfung mit Joinfilter	[INNER] JOIN mit ON
Ergänzung um Anti-Join	LEFT / RIGHT [OUTER]
(Hinzunahme der nicht verknüpfbaren Sätze der Driving Table)	
Satzfilter (Selektion)	WHERE
Gruppenbildung und –auswertung	GROUP
Gruppenfilter	HAVING
Spaltenauswahl (Projektion)	SELECT
Output	

2 SQL-Aufgaben zu Logik und Mengenlehre

Mengenoperationen (Differenz, Vereinigung, Durchschnitt) verlangen, dass die beteiligten Mengen die gleiche Struktur haben.

Ausgangspunkt: Lieferpositionstabelle mit FS KdID und ArtID
Artikelauswahl für einen bestimmten statistischen Berichtszeitraum

Struktur der Lösungen:
SELECT L_ArtID FROM Lieferpositionen
WHERE …
GROUP BY L_ArtID
HAVING …
WHERE … (NOT) IN (Subselect) / MINUS / UNION / INTERSECT
SELECT …
Die folgenden Lösungsangaben sind in verkürzter Form dargestellt!

2 SQL-Aufgaben zu Logik und Mengenlehre

Beispiel für eine Lieferpositionstabelle

KdID	ArtID	Liefermenge	Lieferdatum
X	4711	…	…
Y	4712		
X	4715		
Y	4715		
Z	4713		
X	4717		
Z	4717		
Y	4718		
Z	4718		
X	4719		
Y	4719		
Z	4719		

2 SQL-Aufgaben zu Logik und Mengenlehre

			von genau 1 Kd gekaufte Artikel	von genau 2 Kd gekaufte Artikel	von > 2 Kd gekaufte Artikel
u.a. von Kd X gekaufte Artikel		nur von Kd X gekaufte Artikel			
	u.a. von mind. 1 anderen Kd als X gekaufte Artikel	von Kd X und mind. 1 anderen Kd gekaufte Artikel		von Kd X und genau 1 anderen Kd gekaufte Artikel	von Kd X und mind. 2 anderen Kd gekaufte Artikel
		nur von mind. 1 anderen Kd als X gekaufte Artikel	nur von genau 1 anderen Kd als X gekaufte Artikel	nur von genau 2 anderen Kd als X gekaufte Artikel	nur von mind. 3 anderen Kd als X gekaufte Artikel

2.1 Auswahlbedingungen für e i n e n Kunden

1. Artikel, die u.a. von Kunde X gekauft wurden
(es kann auch andere Kunden geben, die diese Artikel kauften)
= Artikel, zu denen es mindestens eine Lieferposition gibt,
die sich auf den Kunden X bezieht / die den Kunden X betrifft

SELECT DISTINCT L_ArtID … WHERE L_KdID = 'X'

2. Artikel, die jeweils von genau einem beliebigen Kunden gekauft wurden
(es ist egal, welcher Kunde es ist, es darf aber nur einer sein)
= Artikel, deren Lieferpositionen sich auf genau einen Kunden beziehen

```
SELECT L_ArtID …
GROUP BY L_ArtID
HAVING COUNT (DISTINCT L_KdID) = 1
```

2.2 Negation, Komplement

3. Artikel, die u.a. von mind. einem anderen Kunden als X gekauft wurden (können auch von X gekauft worden sein)

```
SELECT DISTINCT L_ArtID … WHERE L_KdID <> 'X'
```

4. Artikel, die u.a. von genau einem anderen Kunden als X gekauft wurden (können auch von X gekauft worden sein)

```
SELECT L_ArtID …
WHERE L_KdID <> 'X'
GROUP BY L_ArtID
HAVING COUNT (DISTINCT L_KdID) = 1
```

2.2 Negation, Komplement

Übung:
Artikel, die vom Kd X und genau einem anderen Kunden gekauft wurden

Lösung:
Artikel, die u.a. von X gekauft wurden (1.)
geschnitten mit
Artikel, die u.a. von genau einem anderen Kunden als X gekauft wurden (4.)
(oder: Artikel, die von genau zwei Kunden gekauft wurden)

falsch:

```
SELECT L_ArtID …
WHERE L_KdID = 'X'
GROUP BY L_ArtID
HAVING COUNT (DISTINCT L_KdID) = 2
```

Satzfilterung vor Gruppierung, also liefert SELECT die leere Menge

2.3 Mengendifferenz

5.1 Artikel, die ausschließlich von Kunde X gekauft wurden
(es darf keinen anderen Kunden geben, der diese Artikel kaufte)

```
SELECT DISTINCT L_ArtID …
WHERE [L_KdID = 'X'
AND] L_ArtID NOT IN / MINUS
(SELECT L_ArtID …
WHERE L_KdID <> 'X')
```

Artikel, die [u.a. von Kunde X] gekauft wurden
ohne / minus
Artikel, die u.a. von mind. einem anderen Kunden als X gekauft wurden

falsch: WHERE L_KdID = 'X' AND NOT (L_KdID <> 'X')
logisch äquivalent zu L_KdID = 'X'

2.3 Mengendifferenz

5.2 Artikel, die ausschließlich von Kunde X gekauft wurden
(es darf keinen anderen Kunden geben, der diese Artikel kaufte)

```
SELECT L_ArtID …
GROUP BY L_ArtID
HAVING COUNT(DISTINCT L_KdID) = 1
MINUS
SELECT DISTINCT L_ArtID …
WHERE L_KdID <> 'X'
```

Artikel, die von genau einem Kunden gekauft wurden
ohne / minus
Artikel, die u.a. von mind. einem anderen Kunden als X gekauft wurden

2.3 Mengendifferenz

5.3 Artikel, die ausschließlich von Kunde X gekauft wurden
(es darf keinen anderen Kunden geben, der diese Artikel kaufte)

```
SELECT DISTINCT L_ArtID …
WHERE L_KdID = 'X'
AND L_ArtID NOT IN / MINUS
(SELECT L_ArtID …
GROUP BY L_ArtID
HAVING COUNT(DISTINCT L_KdID) > 1)
```

Artikel, die u.a. von Kunde X gekauft wurden
ohne / minus
Artikel, die von mehr als einem Kunden gekauft wurden

2.3 Mengendifferenz

5.4 Artikel, die ausschließlich von Kunde X gekauft wurden
(es darf keinen anderen Kunden geben, der diese Artikel kaufte)
als Mengendurchschnitt (kommutativ)

```
SELECT L_ArtID …
WHERE L_ArtID IN
   (SELECT L_ArtID …
    WHERE L_KdID = 'X')
GROUP BY L_ArtID
HAVING COUNT (DISTINCT L_KdID) = 1
```

Innerer Select: Artikel, die u.a. von Kunde X gekauft wurden
Äußerer Select: Artikel, die von genau einem Kunden gekauft wurden

WHERE L_KdID = 'X'ohne inneren Select liefert die u.a. von X gekauften

2.3 Mengendifferenz

5.5 Artikel, die ausschließlich von Kunde X gekauft wurden
(es darf keinen anderen Kunden geben, der diese Artikel kaufte)
als Mengendurchschnitt (kommutativ)

```
SELECT DISTINCT L_ArtID …
WHERE L_KdID = 'X'
AND L_ArtID IN / INTERSECT
(SELECT L_ArtID FROM L
GROUP BY L_ArtID
HAVING COUNT (DISTINCT L_KdID) = 1)
```

Artikel, die u.a. von Kunde X gekauft wurden
geschnitten mit
Artikel, die von genau einem Kunden gekauft wurden

2.3 Mengendifferenz

6.1 Artikel, die nur von mind. einem anderen Kunden,
aber nicht von Kunde X gekauft wurden

```
SELECT DISTINCT L_ArtID …
WHERE L_ArtID NOT IN / MINUS
(SELECT L_ArtID FROM L
WHERE L_KdID = ‘X’)
```

Alle verkauften Artikel
ohne / minus
Artikel, die u.a. von Kunde X gekauft wurden

falsch:
WHERE L_KdID <> ‘X’
findet Artikel, die u.a. von mind. 1 anderen Kunden als X gekauft wurden

2.3 Mengendifferenz

6.2 Artikel, die nur von mind. einem anderen Kunden,
aber nicht von Kunde X gekauft wurden

```
SELECT L_ArtID FROM Lieferpositionen
MINUS
SELECT L_ArtID FROM Lieferpositionen
GROUP BY L_KdID, L_ArtID
HAVING COUNT(*) > 0 AND L_KdID = 'X'
```

Alle verkauften Artikel
ohne / minus
Artikel, die u.a. von Kunde X gekauft wurden

2.3 Mengendifferenz

7.1 Artikel, die von genau einem Kunden, aber nicht von X gekauft wurden
= Artikel, die ausschl. von genau einem anderen Kd. als X gekauft wurden

```
SELECT L_ArtID …
WHERE L_ArtID NOT IN
(SELECT DISTINCT L_ArtID …
WHERE L_KdID = 'X')
GROUP BY L_ArtID
HAVING COUNT (DISTINCT L_KdID) = 1
```

Innerer Select: Artikel, die u.a. von Kunde X gekauft wurden
Äußerer Select: Artikel, die von genau einem Kunden gekauft wurden

2.3 Mengendifferenz

7.2 Artikel, die von genau einem Kunden, aber nicht von X gekauft wurden
= Artikel, die ausschl. von genau einem anderen Kd. als X gekauft wurden

```
SELECT L_ArtID …
GROUP BY L_ArtID
HAVING COUNT (DISTINCT L_KdID) = 1
MINUS
SELECT DISTINCT L_ArtID …
WHERE L_KdID = 'X'
```

Artikel, die von genau einem Kunden gekauft wurden
ohne / minus
Artikel, die u.a. von Kunde X gekauft wurden

2.3 Mengendifferenz

7.3 Artikel, die von genau einem Kunden, aber nicht von X gekauft wurden
= Artikel, die ausschl. von genau einem anderen Kd. als X gekauft wurden
als Mengendurchschnitt (kommutativ)

```
SELECT L_ArtID …
WHERE L_ArtID IN
(SELECT DISTINCT L_ArtID …
WHERE L_KdID <> 'X')
GROUP BY L_ArtID
HAVING COUNT (DISTINCT L_KdID) = 1)
```

Innerer Select: Artikel, die
u.a. von mind. einem anderen Kunden als X gekauft wurden
Äußerer Select: Artikel, die von genau einem Kunden gekauft wurden

2.3 Mengendifferenz

7.4 Artikel, die von genau einem Kunden, aber nicht von X gekauft wurden
= Artikel, die ausschl. von genau einem anderen Kd. als X gekauft wurden
als Mengendurchschnitt (kommutativ)

```
SELECT DISTINCT L_ArtID …
WHERE L_KdID <> 'X'
AND L_ArtID IN / INTERSECT
(SELECT L_ArtID …
GROUP BY L_ArtID
HAVING COUNT (DISTINCT L_KdID) = 1)
```

Artikel, die u.a. von mind. einem anderen Kunden als X gekauft wurden
geschnitten mit
Artikel, die von genau einem Kunden gekauft wurden

2.4 Durchschnitt / Und (kommutativ)

8. Artikel, von denen jeder einzelne u.a. sowohl von Kunde X als auch von Kunde Y gekauft wurde
(es kann auch andere Kunden geben, die diese Artikel kauften)

```
SELECT DISTINCT L_ArtID …
WHERE L_KdID = ‘X’
AND L_Art ID IN / INTERSECT
(SELECT L_ArtID …
WHERE L_KdID = ‘Y’)
```

Artikel, die u.a. von Kunde X gekauft wurden
geschnitten mit
Artikel, die u.a. vom Kunden Y gekauft wurden

falsch: WHERE L_KdID = ‘X’ AND L_KdID = ‘Y’

2.4 Durchschnitt / Und (kommutativ)

9. Artikel, die sowohl von Kunde X als auch von mindestens einem anderen Kunden gekauft wurden

```
SELECT DISTINCT L_ArtID …
WHERE L_KdID = ‘X’
AND L_ArtID IN / INTERSECT
(SELECT L_ArtID …
WHERE L_KdID <> ‘X’)
```

Artikel, die u.a. von Kunde X gekauft wurden
geschnitten mit
Artikel, die u.a. von mind. einem anderen Kunden als X gekauft wurden
(oder: Artikel, die von mindestens zwei Kunden gekauft wurden)

2.4 Durchschnitt / Und (kommutativ)

10. Artikel, von denen jeder einzelne ausschließlich sowohl von Kunde X als auch von Kunde Y gekauft wurde
(es darf keinen anderen Kunden geben, der diese Artikel kaufte)

```
(SELECT L_ArtID … WHERE L_KdID = 'X'
AND L_ArtID IN / INTERSECT
(SELECT L_ArtID … WHERE L_KdID = 'Y'))
AND L_ArtID NOT IN / MINUS
(SELECT L_ArtID … WHERE L_KdID <> 'X' AND L_KdID <> 'Y')
```

Artikel, von denen jeder einzelne u.a. sowohl von Kunde X als auch von Kunde Y gekauft wurde
ohne / minus
Artikel, die u.a. von mind. 1 anderen Kunden als X und Y gekauft wurden

2.5 Vereinigung / Inclusives Oder (kommutativ)

11. Artikel, von denen jeder einzelne u.a. von Kunde X oder (incl.) von Kunde Y gekauft wurde
(es kann auch andere Kunden geben, die diese Artikel kauften)
WHERE L_KdID = 'X' OR L_KdID = 'Y'

12. Artikel, von denen jeder einzelne ausschließlich von Kunde X oder (incl.) von Kunde Y gekauft wurde
(es darf keinen anderen Kunden geben, der diese Artikel kaufte)
SELECT L_ArtID … WHERE L_KdID = 'X' OR L_KdID = 'Y'
AND L_ArtID NOT IN / MINUS
(SELECT L_ArtID … WHERE L_KdID <> 'X' AND L_KdID <> 'Y')

Artikel, die u.a. von Kunde X oder (incl.) von Kunde Y gekauft wurden ohne / minus
Artikel, die u.a. von mind. 1 anderen Kunden als X und Y gekauft wurden

2.6 Mengenkomplement / symmetrische Differenz / Exclusives Oder (nur der Vollständigkeit wegen)

13. Artikel, von denen jeder einzelne u.a. entweder von Kunde X oder (excl.) Kunde Y gekauft wurde
(es kann auch andere Kunden geben, die diese Artikel kauften)

Vereinigung \ Durchschnitt

SELECT L_ArtID … WHERE L_KdID = 'X' OR L_KdID = 'Y'
AND L_ArtID NOT IN / MINUS
(SELECT L_ArtID … WHERE L_KdID = 'X'
AND L_ArtID IN / INTERSECT
(SELECT L_ArtID … WHERE L_KdID = 'Y'))

2.6 Mengenkomplement / symmetrische Differenz / Exclusives Oder (nur der Vollständigkeit wegen)

14. Artikel, von denen jeder einzelne ausschließlich entweder von Kunde X oder (excl.) Kunde Y gekauft wurde
(es darf keinen anderen Kunden geben, der diese Artikel kaufte)

Vereinigung \ Durchschnitt \ von mind. 1 anderen Kunden gekaufte Artikel

```
SELECT L_ArtID … WHERE L_KdID = 'X' OR L_KdID = 'Y'
AND L_ArtID NOT IN / MINUS
(SELECT L_ArtID … WHERE L_KdID = 'X'
AND L_ArtID IN / INTERSECT
(SELECT L_ArtID … WHERE L_KdID = 'Y'))
AND L_ArtID NOT IN / MINUS
(SELECT L_ArtID … WHERE L_KdID <> 'X' AND L_KdID <> 'Y')
```

3. Spannendere SQL-Aufgaben

1. Auftragsverwaltung

Anzahl / Liste der Artikel (ID, Bezeichnung), die in einem Statistikzeitraum die geringste/größte in diesem Zeitraum vorkommende Auftragsmenge hatten

Anzahl / Liste der Kunden (ID, Name), die in einem Statistikzeitraum den größten in diesem Zeitraum vorkommenden Auftragswert erbrachten

Die Orte (PLZ, Ortsname), an denen die meisten Kunden wohnen

Anzahl / Liste der Kunden (ID, Name), die erstmalig 2019 etwas bei unserem Unternehmen bestellt haben

Für jeden Kunden (ID, Name) der/die teuerste(n) Artikel (ID, Bezeichnung), den/die er in einem Statistikzeitraum gekauft hat (Tupel-Satzfilter!)

3. Spannendere SQL-Aufgaben

Zu jedem Kunden (ID, Name) die an dessen jüngstem Auftragsdatum bestellten Artikel (ID, Bezeichnung), sofern noch nicht anonymisiert

1. Zu jedem Kunden das jüngste Auftragsdatum bestimmen
2. KdID und jüngstes Auftragsdatum mit Aliasnamen versehen
3. Ergebnis aus 2 als virtuelle Tabelle im SELECT-Befehl
   in Join-Kette (Joinfelder: u.a. KdID und jüngstes Auftragsdatum)
   mit Auftragspositionen verbinden

Die Orte, an denen die meisten Kunden wohnen

1. Zu jedem Ort die Anzahl der Kunden bestimmen (COUNT)
2. Anzahl mit Aliasname „Kundenanzahl“ versehen
3. Maximum der Kundenanzahlen bestimmen
   (Ergebnis aus 2 als virtuelle Tabelle im SELECT-Befehl)
4. Die gesuchten Orte bestimmen (Kundenanzahl = Maximum)
   (Die Kundenanzahl muss in diesem Befehl 2mal bestimmt werden!)

3. Spannendere SQL-Aufgaben

2. Projektverwaltung

Anzahl / Liste der Projekte (ID, Bezeichnung) mit der geringsten Anzahl von Mitarbeitern je Projekt in einem Statistikzeitraum

Anzahl / Liste der Mitarbeiter (ID, Name), die in einem Statistikzeitraum an der größten Anzahl von Projekten mitarbeiten

Anzahl / Liste der Projekte (ID, Bezeichnung) mit der höchsten, in einem Statistikzeitraum für ein Projekt geleisteten Arbeitszeit

3. Spannendere SQL-Aufgaben

3. Bibliothek

(Anzahl der) Benutzer (ID, Name), die heute die höchste heute vorkommende Anzahl von offenen Ausleihvorgängen je Benutzer erreicht haben, d. h. die heute die höchste heute vorkommende Anzahl von Büchern der Bibliothek in Besitz haben

(Anzahl der) Benutzer (ID, Name), die heute die höchste heute vorkommende Anzahl von neuen Ausleihvorgängen je Benutzer erreicht haben

Zu jedem Benutzer die an dessen jüngstem Ausleihdatum ausgeliehenen Medien (Verfasser, Titel), sofern noch nicht anonymisiert

3. Spannendere SQL-Aufgaben

4. Lieferpositionen (Kunden → Lieferpositionen ← Artikel)

Liste derjenigen Artikel, die in einem Berichtszeitraum von allen / keinem Kunden gekauft wurden (Hinweis: über Anzahl der Kunden)

Alle Artikel und alle in einem Statistikzeitraum verkauften Artikel mit den jeweiligen Liefermengensummen
Vorsicht: Lieferpositionstabelle kann auch Lieferungen aus anderen Jahren enthalten, dann wird OUTER JOIN falsch.

Anzahl aller Artikel, von denen in einem Statistikzeitraum insgesamt mehr als 10 Stück verkauft wurden

3. Spannendere SQL-Aufgaben

5. Lieferantenverwaltung

Zu jedem Lieferanten den/die angebotenen Artikel mit dem höchsten vorkommenden Einzelpreis (Tupel-Satzfilter!)

4. Weitere SQL-Konzepte

Zählen mit ROW-NUMBER OVER PARTITION

TOP, LIMIT

Alfred Holl

Meta-Models of IS Modeling Approaches

The primary focus are principles of modeling and basic elements of models independent of their graphical representation.

Only the secondary focus are individual notations of model representations (semantic networks with nodes and arcs / edges), such as Jackson, SA, UML etc.

0 Aspects of IS models and their notations (multi-perspectivity)

1 Rules for graphical model representations

2 Static function(-oriented) models: function structure models

3 Dynamic data(-oriented) models: information flow models

4 Static data(-oriented) models: data (structure) models

5 Dynamic function(-oriented) models: control flow models

6 Conclusion

0 Aspects of IS models and their notations

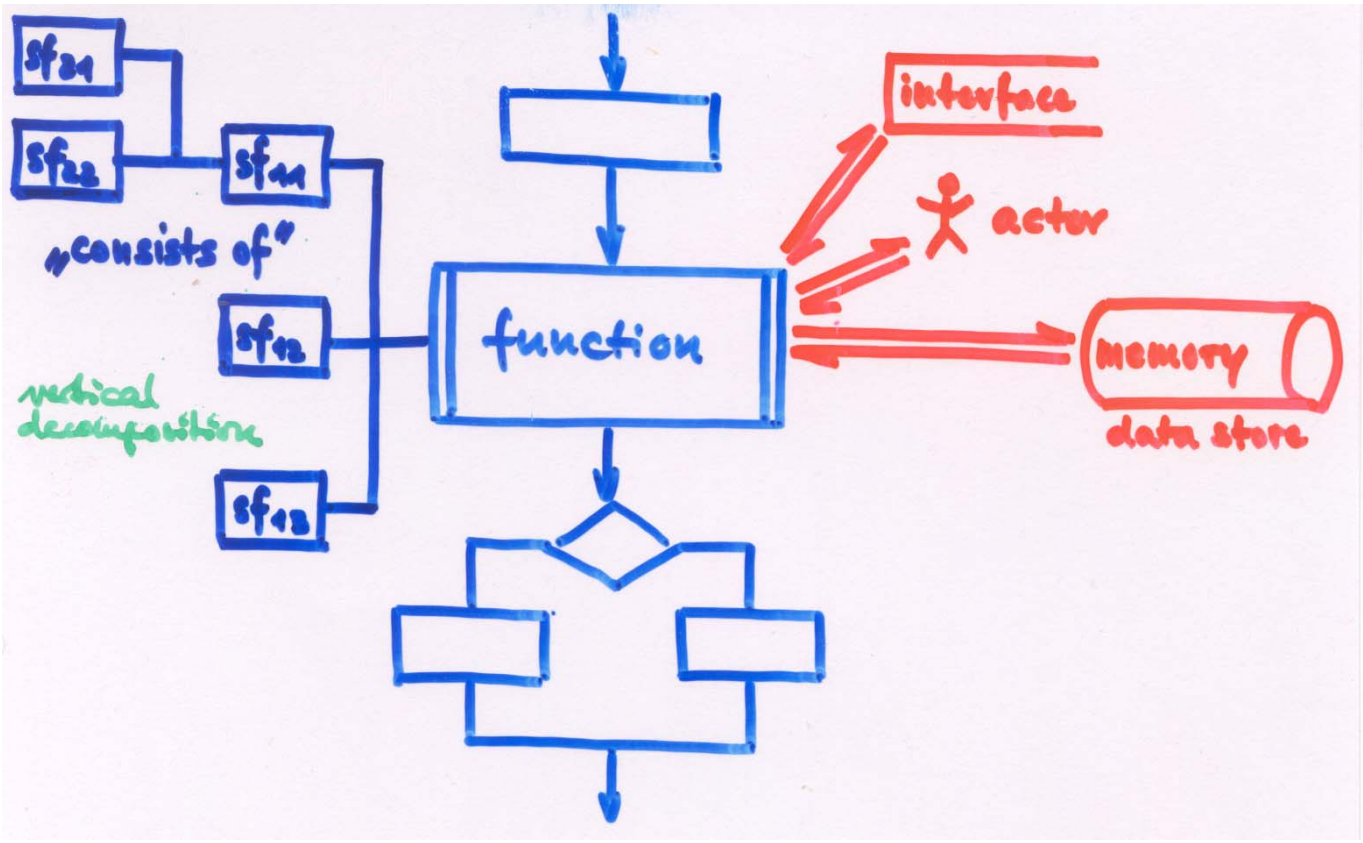
Horizontal multi-perspectivity / decomposition: static and dynamic data and function models 1

	static/structure models	dynamic/behavior models
data models	data (structure) models: data structure diagrams; entity-relationship models (ERM) UML class diagrams	information flow models: information / data flow charts / diagrams; Structured Analysis (SA); UML use case diagrams
function models	function structure models: compositional function trees; Jackson trees	control flow models: algorithms (functions); Nassi-Shneiderman diagrams, block diagrams (flow charts); business process models; UML activity diagrams; (UML sequence diagrams)

Each of the four aspects represents a certain perspective.

0 Aspects of IS models and their notations

Horizontal multi-perspectivity / decomposition: static and dynamic data and function models 2



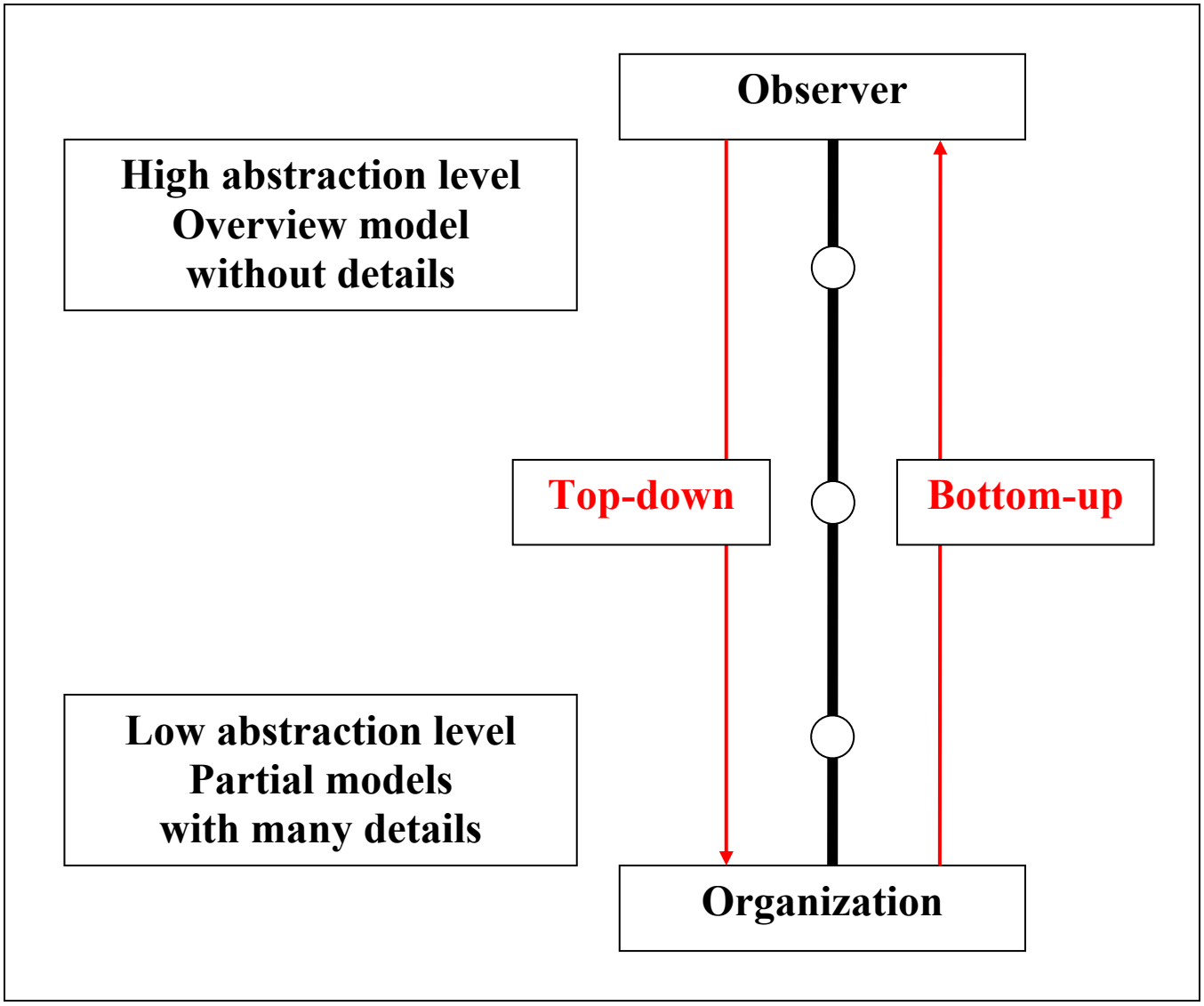
Static function model: function structure model irrespective of tests, iterations, sequences	Dynamic function model: control flow model	Dynamic data model: information flow model	Static data model: data structure model

0 Aspects of IS models and their notations

Vertical multi-perspectivity / decomposition: levels of abstraction

Using design methods (top-down, bottom-up, inside-out), models have to be decomposed into small and transparent partial models on different levels of abstraction (hierarchical levels with different degrees of abstraction).

Every level of abstraction represents a certain perspective.



1 Rules for graphical model representations

All of the nodes have unequivocal (distinct) names.

Vertical multi-perspectivity

Hierarchic decomposition (top-down refinement)
of all structure components on different abstraction levels
→ compatibility of adjacent abstraction levels

Horizontal multi-perspectivity

→ compatibility of adjacent model aspects

---

7 to max. 9 nodes per diagram ('chunk', 'Superzeichen')
cf. Miller, George A.: *The magical number 7, plus or minus 2*, The Psychological Review 63(1956), 81-97

→ transparent arrangement of the symbols
→ sequence of comments according to this arrangement

Diagrams and comments mutually complete each other.

2 Function structure models

function trees, ‘consists of’-hierarchies

irrespective of control flow (conditions, iterations)

3.1 Information flow models: Elements 1

Nodes:

1. functions

– template for function names: verb + noun (object)
→ quantifier if singular and plural cannot be distinguished

2. stores, memories

Types of information media ('Datenträger'; to be marked in diagrams):
– digital stores / media
– paper (printout, document ('Beleg', 'Schriftstück'), card index ('Kartei'), binder ('Ordner'))
– mental memories, skills, knowledge (head)
→ type purity (only one information type in every store)
→ type compatibility of the attached information flows
→ type conversion only with functions

3. actors (intra-system and extra-system (SA -interfaces); extra-system ones can be intra- or extra-organizational): function owners, responsible persons ('Aufgabenträger', 'Funktionsträger'), persons involved ('Beteiligte'); institutions; IT systems

3.1 Information flow models: Elements 2

Arcs:

data / information flows (type purity)
– between functions and stores / actors
→ not between different functions
– distinction analogous to store types
(digital, paper, oral communication)
– flow direction (cf. store access modes read / write)
if necessary material and money flows

Not: processes, procedural structures
Auxiliary approaches:
– graphic arrangement of the function symbols
– [event flows (a second arrow type; cf. CASE4.0)]

3.2 Information flow models: Notations

Traditional data flow chart

Process matrix (data flow chart in the form of a table):
– functions, stores, access mode

Structured Analysis (SA) according to Tom de Marco 1978:
– level 0: context diagram (system delimitation)
– only extra-system (external) interfaces / actors

UML use case diagram (‘Anwendungsfalldiagramm’)
– intra-system and extra-system actors

4.1 Data structure models: Elements
1 Entity / object types 1

Nodes:

Entity types, object types, (object) classes

1. attribute structure (attribute ≈ property dimension)
– attributes of the objects (instances) of a class (Instanzattribute)
– attributes of the corresponding class (Klassenattribute)

2. attached procedures, elementary functions, services, 'methods' (called by messages): data encapsulation
– services of the objects (instances) of a class (Instanzmethoden)
– services of the corresponding class (Klassenmethoden)

3. object types
– transient: exist during run-time only
– persistent: stored in permanent memories

→ individual elements: entities, objects, object instances

Relation between SA and OO
– refine SA data stores to object types / classes
– refine SA functions to OO services

4.1 Data structure models: Elements
1 Entity types 2

Entity, object
not: a real object, but: a model, a tuple of attribute values

Entity structure: attribute structure
2 entities have the same structure iff they have the same attributes

Set of entities
Set of arbitrary entities with the same structure

The following problem leads to a definition of an entity type:
A customer table shall only contain one address for each customer, the current address.

Entity type: 2 properties
structure property: invariable attribute structure
compatibility property: definition of a primary key
2 entities with the same structure are compatible iff they can be elements of the same entity set iff their primary key values are different (entity integrity).

Entity set
Set of compatible entities with the same structure (table).

There are always several entity sets of an entity type.
Not every set of entities with the same structure is an entity set.

	World 1	World 3
single object	one real object	entity
set - type	set of homogeneous real objects	entity type

Data store (digital, paper, head): can contain several entity types

4.1 Data structure models: Elements
2 Relationships, associations 1

Arcs:

Relationships, associations (OO) regard
the interrelations between entity types from 3 points of view:

1. numeric classification: cardinality, multiplicity (OO)
– n:m many-to-many relationship
– 1:n one-to-many relationship
– 1:1 one-to-one relationship
– 1:c conditional relationship (c = 0, 1)
– special two-dimensional relationships such as c:c, c:n
– multidimensional relationships: n:m:p, n:m:p:q (e.g. time-table)

CAUTION: The cardinality is also a type
that is, it can have different values for individual entities.

Example:
A one-to-many relationship "customers → orders" can contain
– individual customers with many (two or more) orders
– individual customers with one order
– individual customers with no orders

2. syntactic description: primary key → reference key
– on entity type level: key attributes
– on entity level: key attribute values

→ Referential integrity required!

4.1 Data structure models: Elements
2 Relationships, associations 2

3. semantic interpretation

3.1 simple association without any semantic interpretation

Examples:
1:n customers – orders
1:c employees – (head of) – department

3.2 compositional relationship:
‘consists of’, ‘whole-part’, aggregation

Composition (special case of aggregation): existential dependence

Examples:
1:n orders – order lines
1:c car – air condition

4.1 Data structure models: Elements 2 Relationships, associations 3

3. semantic interpretation

3.3 taxonomic relationship:

'is a', generalization (umbrella term) / specialization
inheritance of attributes and services
polymorphism of services

Example:
1:c business partners – customers in combination with
1:c business partners – suppliers

Interpretation different in OO class diagrams:
no instances of abstract classes

general concept	basic class abstract class	↑ generalization ↓ specialization (with inheritance)
special concept	derived class	

4.2 Data (structure) models: Special models and their notations

1. Pure data models in general

all features mentioned without services (attached procedures)

ERM: entity-relationship model (Chen) and its extensions

DSD: data structure diagram (Bachman) and
Oracle diagram:
no semantic interpretation of relationships

2. 3NF models: special pure data models reduced to axioms (controlled redundancy)

– no services (attached procedures)
– no semantic interpretation of relationships
– one-to-many relationships only

favorite diagram: DSD

3. Static object models

Thesis: should be based on 3NF data models in order to have a stable basis quite independent of subjective influence

all features mentioned
logical primary and reference keys are not always used

UML class diagram

5.1 Control flow models: Elements 1

In the following, behavior meta-models will be examined from the point of view of information systems.

That is, there will be a focus on the activity-on-node variant.

The activity-on-arc variant (state transition networks, Petri nets), which is important for theoretical computer science approaches, will be excluded.

5.1 Control flow models: Elements 2

Nodes:

1. function, action (computer-aided or not)
   function unit, function module

– name from the view of the organization
– decomposition-marker: reference to sub-processes
– algorithm, internal logic in a note
– duration, start time, end time
– features, feature values (→ theory of gestalt)
– IT support: computer-aided or manual

2. initiating and resulting events

3. actor: person/role/department responsible for the action
   partly connected with data flow

4. external (business/communication) partners
   connected with data flow

5. data stores accessed: input data and output data
   connected with data flow

6. resources used (machines etc.)

	World 1 (reality)	World 3 (model)
single object, "instance"	one individual course of events in an organization	business process instance
set - type of similar objects	set of homogeneous courses of events	business process type

5.1 Control flow models: Elements 3

Arcs:

1. control flow: temporal interrelation of functions
(cf. structured programming)

– temporal succession: sequence (predecessors and successors)
mandatory or arbitrary (pseudo-parallelism)

– condition: alternative, selection (IF, XOR)
case discrimination (CASE)
or complex rule (decision table)
disjoint and complete (if incomplete a standard path)

– repetition: iteration, loop (WHILE or REPEAT)
test-first loop and test-last loop

– recursion

– simultaneousness: parallel processing, parallelism (AND)
mandatory or arbitrary (pseudo-parallelism)

CAUTION:
all control flow elements without the mere sequence must have
a divergent delimitor (begin) and
a convergent delimitor (end, synchronization);
the delimitors have to be arranged symmetrically in a diagram:
IF – ENDIF, CASE – ENDCASE, LOOP – ENDLOOP etc.

2. data flow (only partly)

3. mere connectors to actors and resources used

5.2 Control flow models: Special models and their notations 1

1. Classical notations

1.1 Traditional notations for structured programming

flow chart, block diagram ('Programm-Ablauf-Plan')
structure diagram, structogram (Nassi-Shneiderman diagram)
Jackson tree
- Jackson structured design (JSD)
- Jackson structured programming (JSP)

functions and control flow

1.2 Decision table

complex conditions and functions: rules

1.3 Network model(ing technique)

functions, sequence, parallel processing,
duration, start time, end time
→ critical path

1.4 Control flow plus data flow

HIPO: hierarchy plus input-process-output (Mills 1972, IBM)
functions, control flow, data stores, data flow

5.2 Control flow models: Special models and their notations 2

1.5 Swim lane diagrams

functions, control flow, responsible departments
predecessor of UML activity diagram
Arbeitsablaufdiagramm: Arbeitsschritte – Abteilungen
Organisationsprozessdarstellung (H. F. Binner)

2. Business process models

Event-driven process chain (EPC)
OMG Standard: Business Process Diagrams (BPD) using Business Process Model and Notation (BPMN)

functions, control flow
events
actors, partners, data stores, resources, data flow
swim lanes (responsible departments)

3. Dynamic object models

UML activity diagram

functions, control flow
events (non standard)
actors, partners, data stores, resources, data flow
swim lanes (responsible departments)

UML sequence diagram

classes, elementary functions called by messages, control flow

6 Conclusion: Model notations as semantic networks

All the models mentioned can be represented by semantic networks with nodes and arcs.

Analogies:
data model: entity types and relationships
process model: functions and control flow
different types of control flow correspond to different types of relationships between entity types
in addition:
data flow models contain different types of nodes:
functions, actors and data stores (entity types)

---

7 References

Böhm, Corrado; Jacopini, Giuseppe:
Flow diagrams, Turing machines and languages with only two formation rules.
***Communications of the ACM* 9(1966) 5, 366-371.**

Dijkstra, Edsger:
GOTO statement considered harmful.
***Communications of the ACM* 11(1968) 3, 147-148.**

Alfred Holl

Rationalistic approaches to IS modeling: analogy and reference models

1 Motivation 1

1.1 Analogical thinking, a cognitive strategy 1

Cognitive dilemma 1

(Neolithic) Humans need(ed) information (knowledge)
to quickly master new situations,
but humans cannot know every object of cognition.
They have too compare them with well-known situations.
=> the cognitive necessity of comparisons:
analogical thinking / reasoning

Purpose of analogical thinking:
Quick extension of the knowledge about some new situation
based upon a comparison, upon analogy,
no logical conclusion, but a heuristic strategy.

Procedure:
1. Situation
A new and a well-known object of cognition (requires memory!)
coincide in some features.
2. Assumed consequence (analogical knowledge transfer)
Assumption of analogy:
They coincide in all their "important" features,
at least one more feature.

Assumption of a strong analogy starting from a weak one,
assumption of an extensibility of an existing analogy.
(In German: Analogieschluss = Schluss auf stärkere Analogie)

The correctness of assumptions of analogy cannot be proved.

1.1 Analogical thinking, a cognitive strategy 2

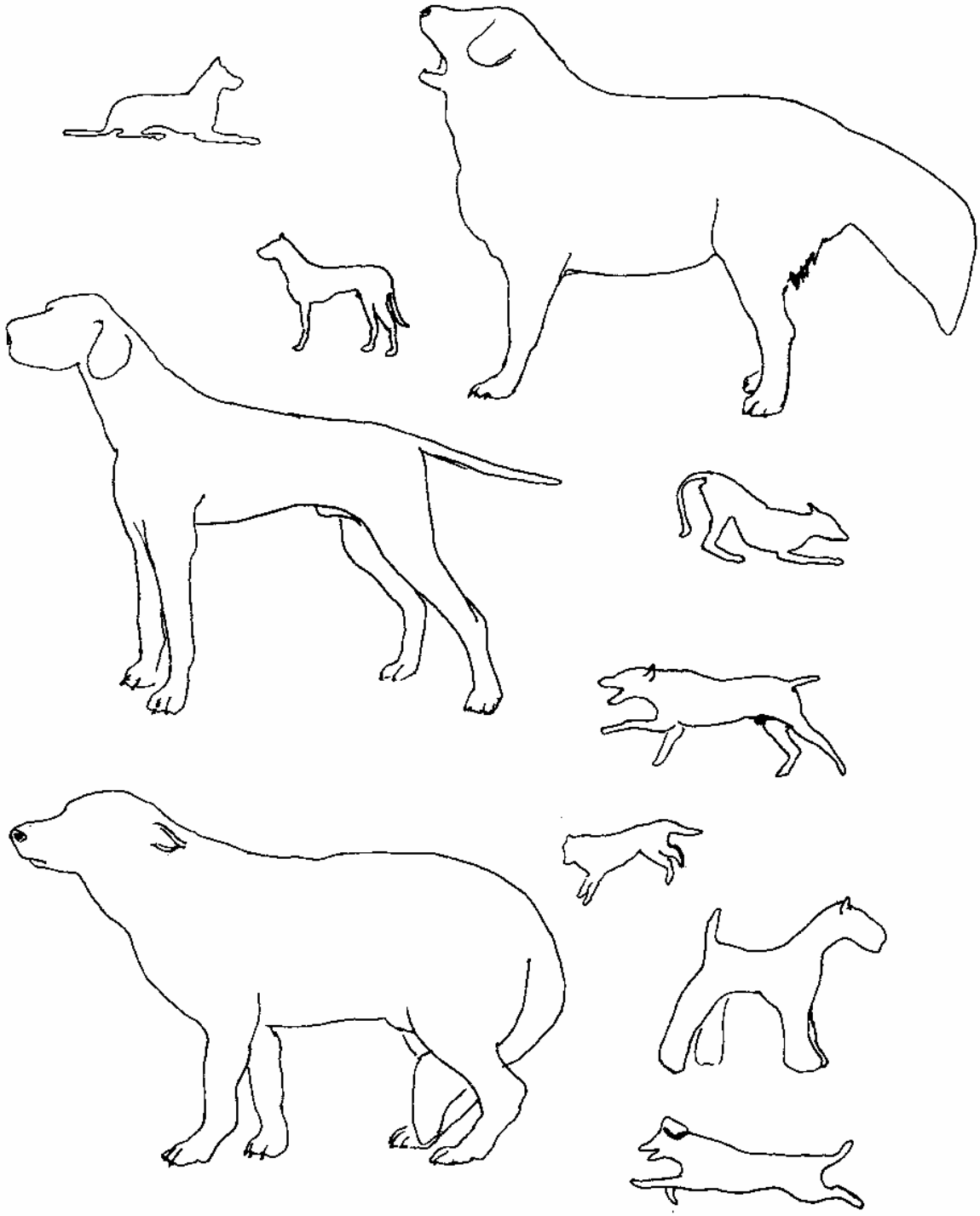


Abb. 13: Beispiel für Gestaltwahrnehmung des Menschen. Jede dieser Figuren erkennen wir als „Hund", obwohl es sich nur um grobe Umrißzeichnungen handelt. Wir abstrahieren die für Hunde essentiellen Merkmale, die „Hundegestalt", und erkennen diese auch in vereinfachenden Abbildungen, sofern nur die betreffenden Merkmale herausragen.

Assignment of individuals to a type using key features (Wuketits, Entdeckung des Verhaltens, 1995, 71)

1 Motivation 3

1.2 Examples for analogical thinking in IS 1

- **Data Mining** techniques (statistical and non-statistical), knowledge discovery in databases: similarities of data objects are used for inductive type construction.
- Transfer of **reference models** to "analogical" application fields;
  cf. purchase and sales depts. in a company
  (vs. the second source of modeling: observation / interview)
- Taxonomy in OO class models (**generalization**)
- Almost all IS models are **type models**.
  We don't model an individual customer, but a customer type.
  We don't model individual sales processes, but a sales process type.
  (Exception: model of an individual machine)
- The concept of analogy can be used for static and dynamic situations, e.g. data structures and process structures.
- Pattern recognition
- Design patterns
- etc.

1 Motivation 4

1.2 Examples 2 – Two sources for model design

Popper's World 1 (reality): empiristic method/approach
Organization, company, department
observation and interviews (W3)
of employees by a model designer
(contrary to natural sciences: only observation)
preliminary description in pre-formal models: natural language
abstraction
check whether terminology is mathematically well-defined
final type construction
formalization (degree of pre-formalization is different)
reduction to axioms

often used for peripheral areas of models
often used for individual parts of an organization
(nominalist point of view: enumeration of individual objects)

Popper's World 3 (models, concepts, ideas): rationalistic method
reference models
activation in a model designer's brain
analogy-based transfer

often used for central areas of models
often used for standard parts of an organization, e.g. accounting
(universalist point of view: search for general principles)

Final step: integration of individual and reference models.

All steps have to be taken in World 2.

1 Motivation 5

1.2 Examples 3 – Two sources for model design

external world ↓ World 1 objects of cognition	phenomenon, individual experience ↓ World 2 knowledge of an individual subject of cognition			model, theory ↓ World 3 common knowledge
	perception, cognitive processes (empiristic) ↓ reconstruct. of World 1 →	memory	learning rationalistic ↓ activations of World 3 ←	
←	↓ creation, induction ↓ design, influence ←	← new ideas, knowledge	→ publi-cation	→

Bi/trilateral semiotic sign		
materialized signifiant, vox	code of interpretation	signifié, conceptus W2|W3
object of cog.		

Model as complex bi/trilateral semiotic sign		
materialized model repre sentation	code of interpretation	model meaning W2|W3
object of cog.		

1 Motivation 6

1.2 Examples – Variables, type models 3

IS experts do not design models of single real objects, such as of individual customers, of the processing of individual orders, (that is up to the organization's employees)
but general models, such as
the common properties of all of the customers,
of the processing of all of the orders.
This fact is the basis for the rationalization potential of IS.

Models with variables: type / class models:
(intensional set definition, that is, no enumeration)

– data model of a set of analogous / equivalent real objects:
tuple of attributes (variables); entity type; OO-class
e.g. customers in general

– function model (algorithm) for a set of equivalent problems:
e.g. algorithm for the calculation of the greatest common divisor of two natural numbers (variables) in general
e.g. algorithm for the processing of orders in general

Models without variables: individual / instance models:

– data model of a single real object:
tuple of attribute values (constants); entity; OO-instance
e.g. one individual customer

– function model for a single problem:
e.g. for the calculation of the greatest common divisor of the two natural numbers 12 and 30 (constants)
e.g. for the processing of order no. 4711

2 Analogy – coincidence of feature values 1

Relation between two objects of cognition
(segments of reality, models; objects, data, processes):
Similarity, comparability, compatibility, associability,
equivalence (in terms of mathematics; → equivalence relation)
– some equal / common features
(tertium comparationis: base of comparison)
– some different features

Example: debtor and creditor management in a company
common: flow of data, goods, money between business partners
different: flow direction (inward, outward),
incoming / outgoing orders,
status of goods (raw material, final product)

Distinction:
– functional analogy: two processes deliver the same result
irrespective of the way of constructing the result
(→ functional model)
Example: copying a text with a copying machine vs. by hand
– structural analogy: two objects of cognition coincide
in selected structural components

We restrict ourselves to the latter kind of analogy.

In biology, analogy has a special meaning (vs. homology):
two recent similar morphological forms
without phylogenetic relationship, without a common ancestor
Examples:
– fins of whales and fish
– wings of bats, birds and flying reptiles

2 Analogy – coincidence of feature values 2

Formalization of the principle of analogy
in order to make models more transparent and better comparable

Feature *F* (based on theory of gestalt):
– dimension *D*
– value *V*
(cf. attributes and attribute values in data modeling)

Example: Feature *F* (*D* color, *V* red)

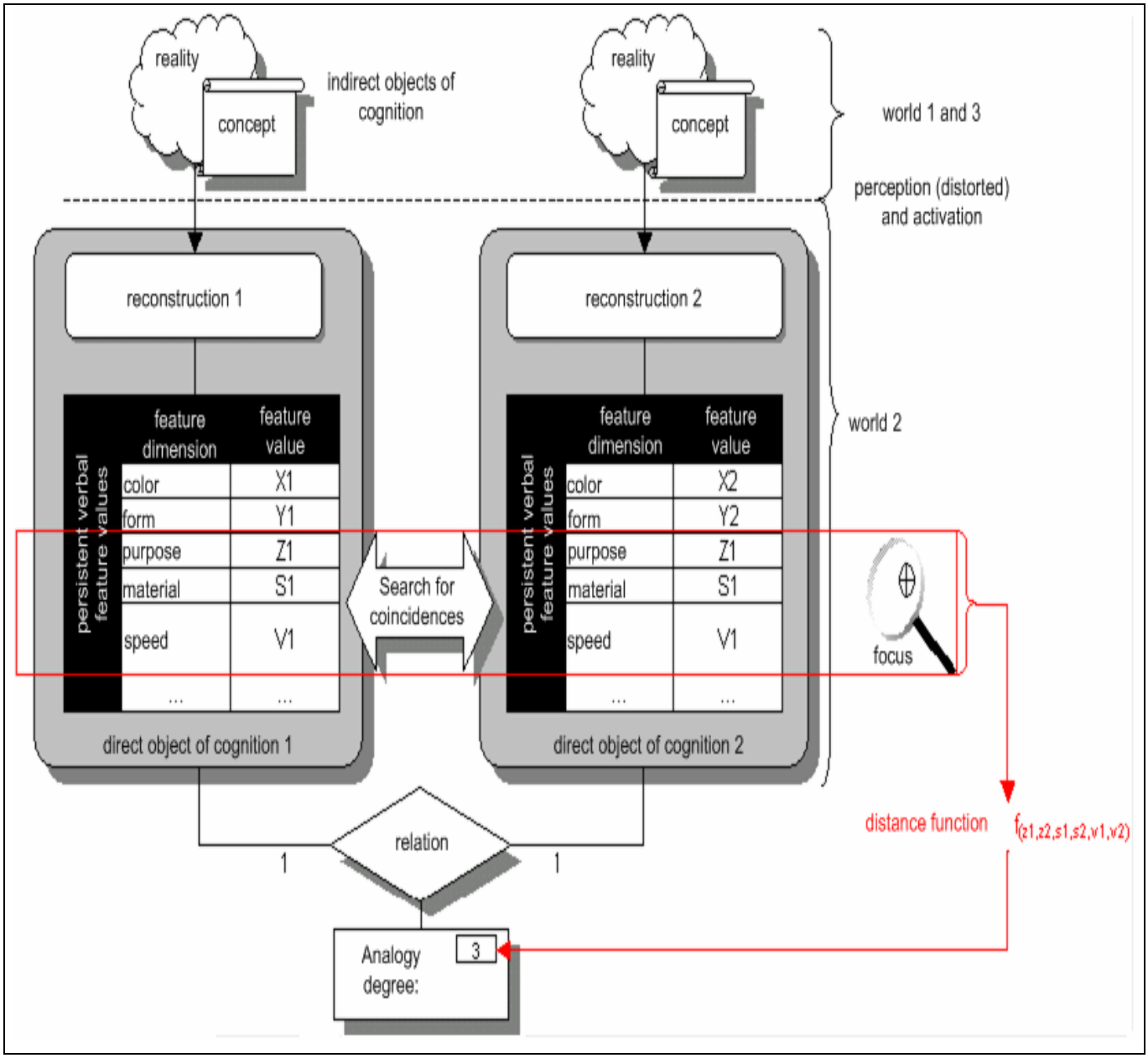
Degree of analogy
between two objects of cognition based on *n* features
calculated by using a weighted measure / function of proximity / similarity:

$$f(V_{11},V_{12},V_{21},V_{22},..,V_{n1},V_{n2}) = \sum_{\substack{i=1..n \\ V_{i1}=V_{i2}}} W_i$$

Pick out the common features in the above set of *n* features.

Two (or *m*) objects of cognition are defined as analogous iff they have
– equal essential (distinctive) features
(are considered as relevant for the comparison)
– different / equal accidental (non-distinctive) features
(don't play any role for the comparison)

2 Analogy – coincidence of feature values 3



Analogies are based upon coincidences of feature values (Holl / Auerochs, Analogisches Denken, 2004, Fig. 2)

3 Analogical thinking 1

3.1 Type construction – induction 1

Type (some sort of a model):
– constituted by equal / common essential features
– found via induction from similar objects of cognition
– a verbal description (umbrella term) can be constructed comprising just the analogous objects of cognition belonging to this type
– different or equal accidental features (e.g. size, number of employees of an organization etc.)

Example:
Customer and supplier (business partners) with
– essential features: name, address, contact person, turnover etc. (short for formal Boolean features (name-yes-no, yes) etc.)
– accidental features: receiver or sender of orders

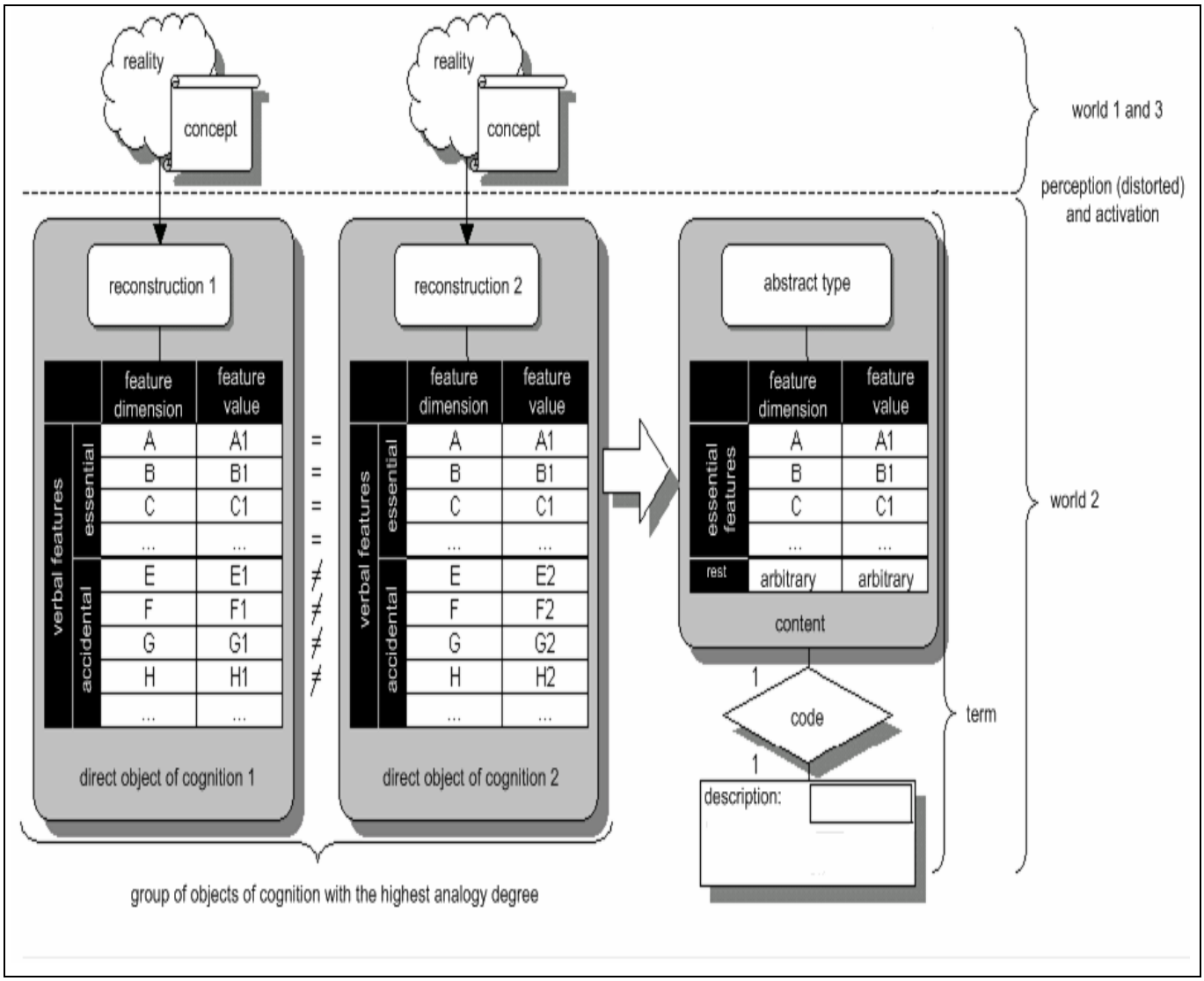
Type construction is done in every natural language where the essential features often remain implicit.

It can be formalized to serve scientific purposes.

Up until now, we distinguish between two kinds of features:
– essential features: common / equal (within a type), distinctive (towards other types)
– accidental features: common or not common non-distinctive

3 Analogical thinking 2

3.1 Type construction – induction 2



Induction step due to postulated analogy
(Holl / Auerochs, Analogisches Denken, 2004, Fig. 3)

3 Analogical thinking 3

3.1 Type construction – induction 3

**An object of cognition can be assigned
to different essential features, that is, to different types,
depending on the compared object of cognition.**

Analogy is always relative to a given set of essential features.

Example:
**Customer 1 – customer 2: customers with
more than 10,000 $ turnover a year**
**Customer 1 – customer 3: customers with
A-rating**
Customer 1 – customer 4: regular customers

Weak analogy: “few” essential features
Strong analogy: “many” essential features

**“An analogy can be more or less detailed and
hence more or less informative.”
(Konrad Lorenz, Analogy as a source of knowledge, 1974, 186)**

3 Analogical thinking 4

3.2 Levels of analogy

Analogy can be defined between objects of cognition on various levels of cognition/existence, between
– 1 objects of cognition of World 1
– 2 types, (parts of) models (World 3 objects of cognition)
– 3 objects of cognition of World 1 and types (World 3)

A type is also an object of cognition!

Examples (case 1):
Socrates, Aristotle;
this swan, that swan
customer 1, customer 2

Example (case 2):
philosopher, human;
ostrich, swan, bird
customers, suppliers

Examples (case 3):
Socrates, humans;
this swan, swans
customer 1, customers

3 Analogical thinking 5

3.3 Reasoning – deduction 1

1 Classification using essential features
2 Transfer using a pars-pro-toto strategy

Example (case 3) with true conclusion:
modus ponens (a sort of a syllogism = logical conclusion)
Humans are mortal.
common accidental (non-distinctive) feature of a type
Classification:
Socrates is a human.
coincidence object of cognition - type
in essential features (or key features, see 4)
Transfer:
Socrates is mortal.
common accidental feature of an object of cognition
(or essential feature if one starts with key features)

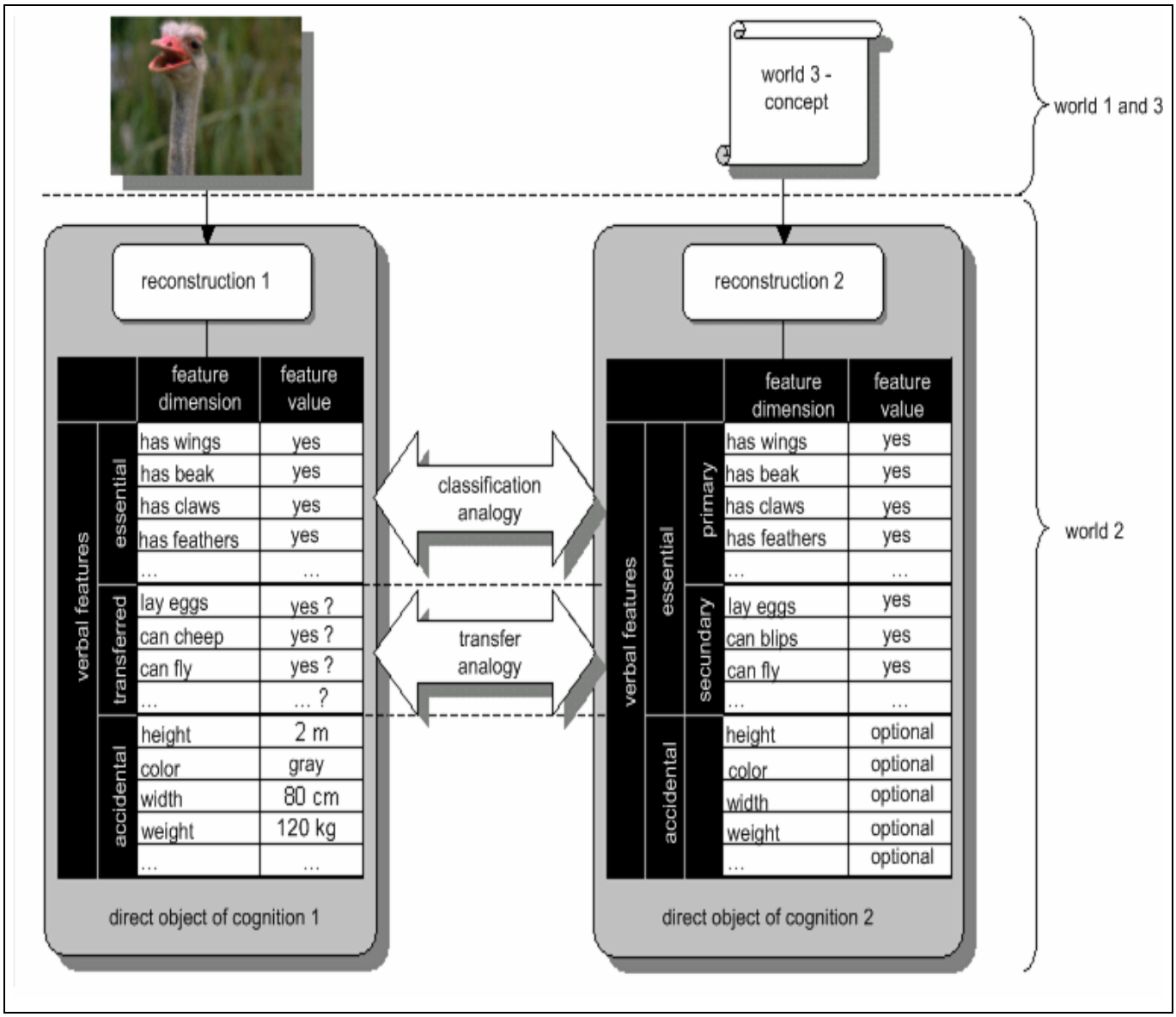
Example (case 3) with false conclusion:
Every swan is white.
This bird is a swan.
This bird is white.

Example (case 2) with false conclusion:
A swan can fly.
Ostrich and swan are analogous (are birds).
An ostrich can fly.

Correctness of assumptions of analogy:
– ⇔ adequacy of selected essential features (or key features)
– cannot be proved.
Risk: This kind of thinking can be a cognitive trap!

3 Analogical thinking 6

3.3 Reasoning – deduction 2



Deductive conclusion with the help of analogy
(Holl / Auerochs, Analogisches Denken, 2004, Fig. 4)

Up until now, we distinguish between <u>three kinds of features</u>:
– essential features (classification)
– common accidental features (transfer)
– different accidental features

3 Analogical thinking 7

3.4 Relation between analogy and induction / deduction

Induction
(due to cognitive dilemma 1)
Starting from some similar / analogous objects of cognition of the same type,
that is objects of cognition with the same essential features,
a theory / model of a common accidental feature is derived.
This is a creative, heuristic (not logical) procedure!

Deduction
Situation:
There is a theory about a common accidental feature of a type.
Classification: The type and some other object of cognition coincide in their essential features.
Transfer – analogical assumption – (logical) conclusion:
Type and object of cognition are analogous,
that is, they coincide in all their essential features,
therefore, the theory applies for the object of cognition.
(analogical transfer of common accidental features)

Or even in a weaker form (see 4):
Classification: The type and some other object of cognition coincide in key features.
Transfer – analogical assumption – (logical) conclusion:
Type and object of cognition are analogous,
that is, they coincide in all their key features,
therefore, the theory applies for the object of cognition.
(analogical transfer of common accidental features and secondary essential features)

3 Analogical thinking 8

3.5 Popper's fallibilism 1

Verification / falsification (Karl Popper)

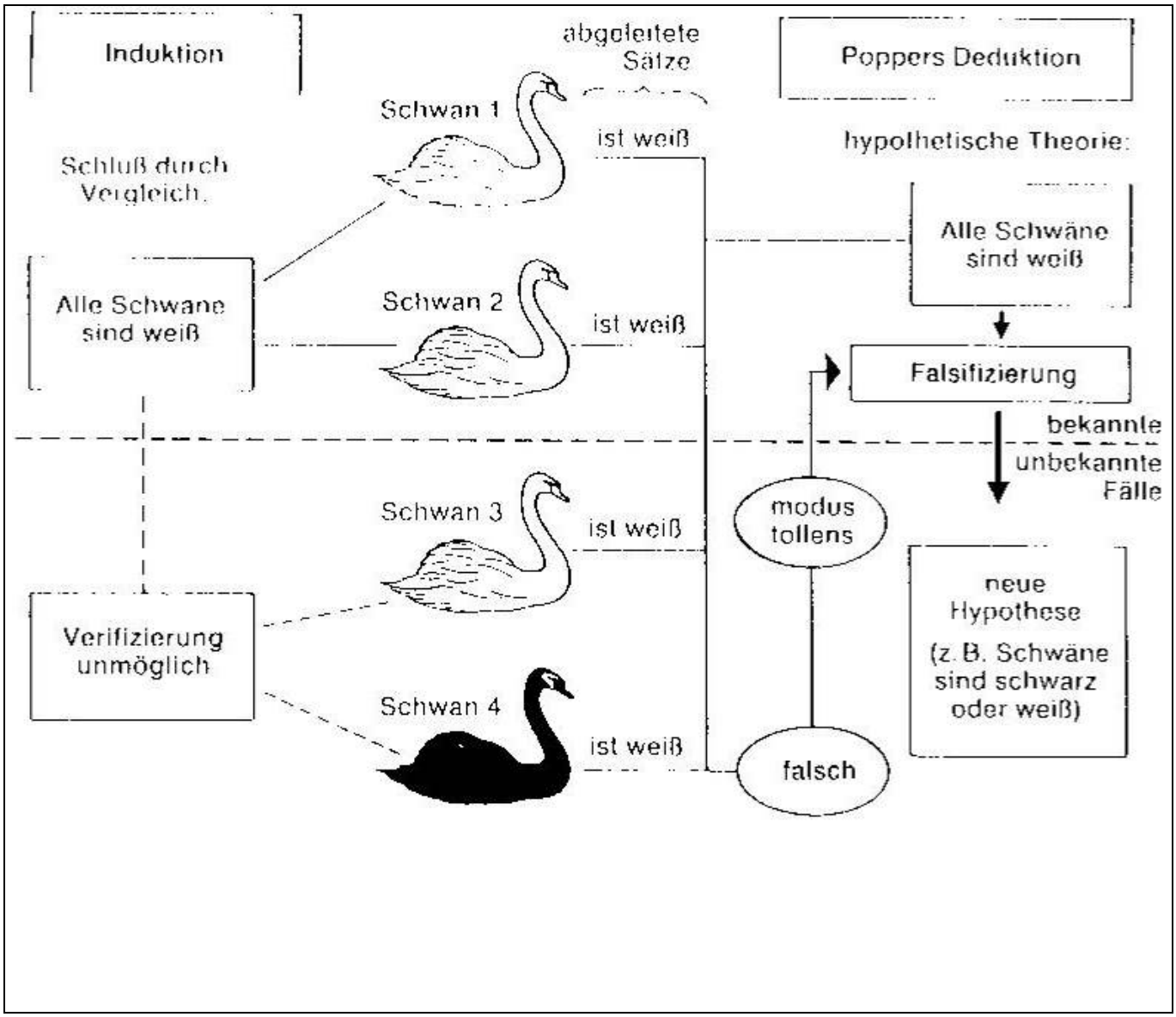
As we do not know all the objects of cognition of a given type, inductively derived theories cannot be proved;
cf. *every swan is white*, *every bird can fly*

That is – as we already know –
the correctness of assumptions of analogy cannot be proved and
the correctness of logical deductions
starting from an inductively derived (only falsifiable) theory cannot be proved.
The results cannot be more true than the pre-conditions.

Deduction works correctly only with well-defined mathematical objects.

3 Analogical thinking 9

3.5 Popper's fallibilism 2



Can swans be black?
(dtv-Atlas Philosophie, *, 228)**

3 Analogical thinking 10

3.5 Popper's fallibilism 3

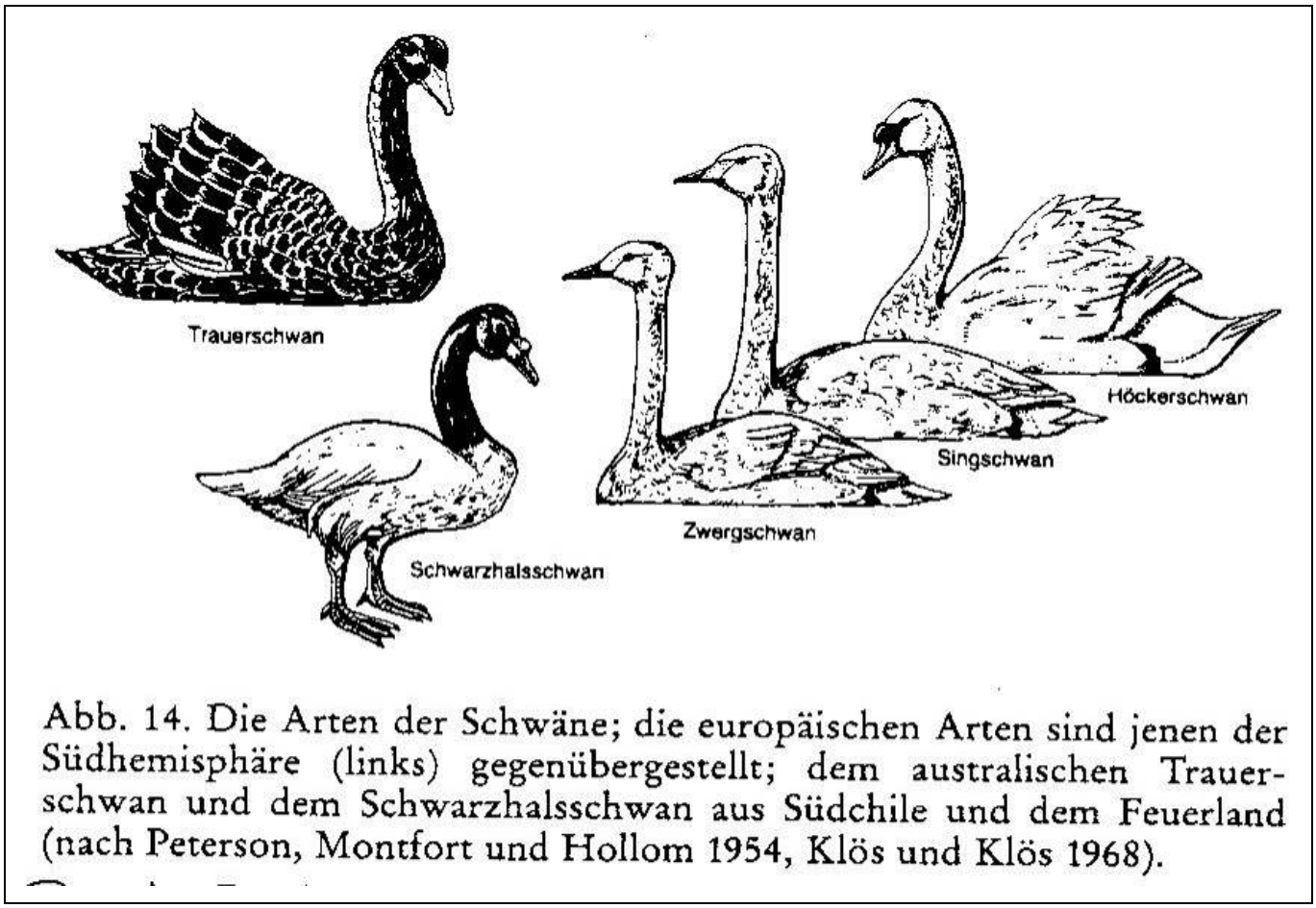


Abb. 14. Die Arten der Schwäne; die europäischen Arten sind jenen der Südhemisphäre (links) gegenübergestellt; dem australischen Trauerschwan und dem Schwarzhalsschwan aus Südchile und dem Feuerland (nach Peterson, Montfort und Hollom 1954, Klös und Klös 1968).

The genus "swan"
(Riedl, Biology of knowledge, 1984, 83)

4 Key feature based analogical thinking 1

Cognitive dilemma 2

(Neolithic) Humans need information to master these situations in the most adequate possible way, but
every object of cognition has numerous features,
among them not easily observable ones and even hidden ones.
The complete observation of all the essential features of an object of cognition is impossible,
it would take too much time or even destroy the object,
but quick reactions are necessary for survival.
cf. lion in the bush, roars, but is not visible

=> the cognitive necessity of partial comparisons
based upon only few features ("key features")

The cognitive strategy of analogical thinking is originally a heuristic cognitive pars-pro-toto (part instead of total) strategy based upon so-called key features
(Konrad Lorenz, Die angeborenen Formen möglicher Erfahrung, 1943, 240: key stimuli, pars-pro-toto reactions)

Key features (directly perceptible, e.g. optical):
– considered as important in the sense of the theory of gestalt
Konrad Lorenz 1959:
"Gestalt perception as source of scientific knowledge."
– (un)consciously, heuristically defined by observer/scientist
– not type-immanent, depending on object and observation

Example:
Customers and suppliers are
companies connected with our own company
by business transactions (data, goods, money)

4 Key feature based analogical thinking 2

Highly significant essential features can serve as key features.

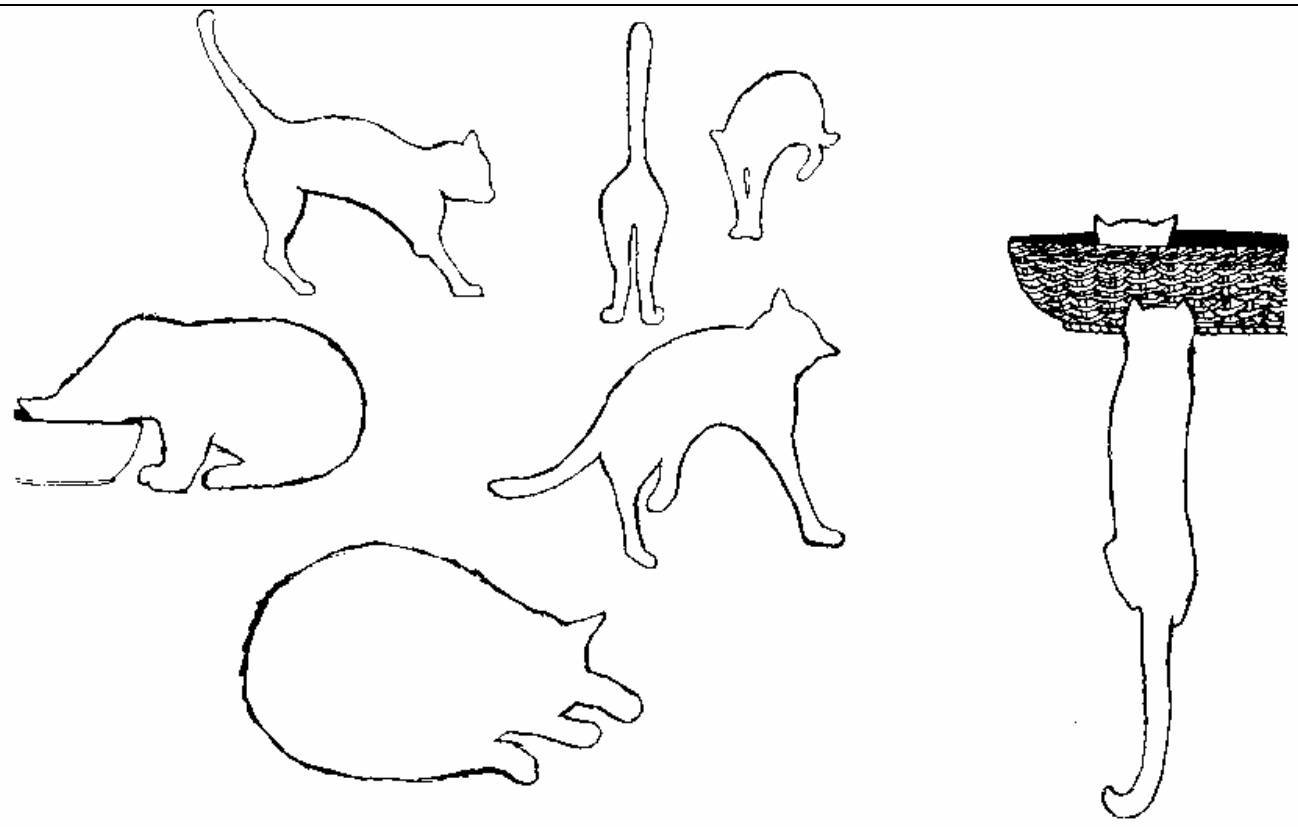


Abb. 25. Konstanz-Wahrnehmung der Gestalt. Obwohl die Netzhaut-Bilder von den gezeichneten Figuren in hohem Grade unterschiedlich sind, wird deren Betrachtung, durch die Abstraktion und Ergänzung in der Gestalt-Wahrnehmung, dennoch zu dem Schlusse führen, daß es sich in allen Gestalten um gleiches, ja um dasselbe handelt. Jede Darstellung erweist sich aus dem gesamten Hintergrundwissen über den Gegenstand komplettiert.

One animal or different animals?
(Riedl, Biology of knowledge, 1984, 167)

4 Key feature based analogical thinking 3

At last, we distinguish between four kinds of features:
– primary essential features (suitable as key features)
– secondary essential features (not suitable as key features)
– common accidental features (transfer)
– different accidental features

Example: human

Primary essential features (suitable as key features)
– shape of the body
– shape of the face
– movement on two legs
– ability to speak

Secondary essential features (not suitable as key features)
– cortex of the brain

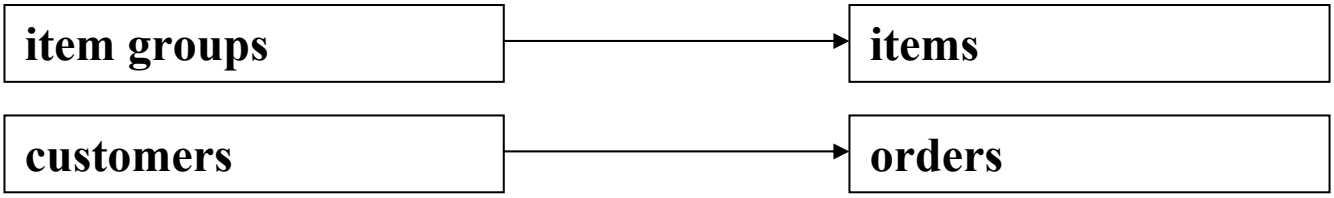
Common accidental features
– mortality

Different accidental features
– color of hair
– color of skin
– height
– sex

Essential features are common /equal and distinctive.
Accidental features are common or not and non-distinctive.
Secondary essential features and common accidental features can be used for analogical transfer.

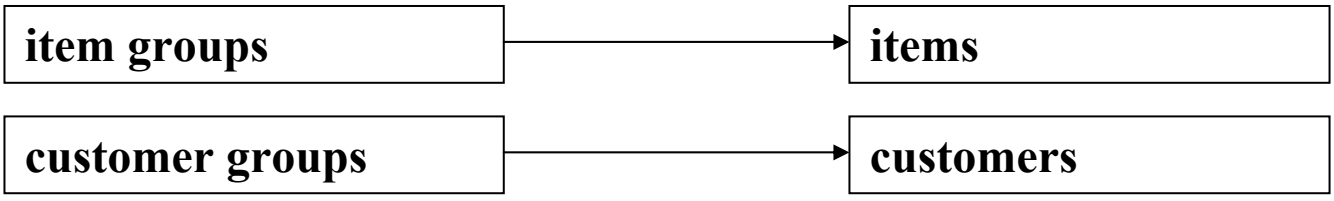
5.1 Data models: What degrees of analogy occur?

1 mere syntactic

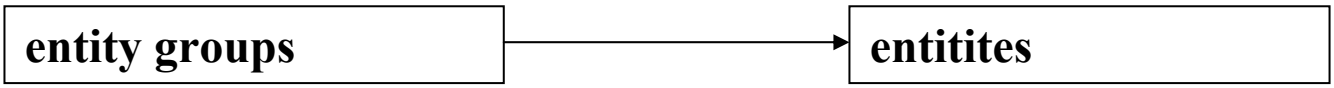


analogy: one-to-many relationship

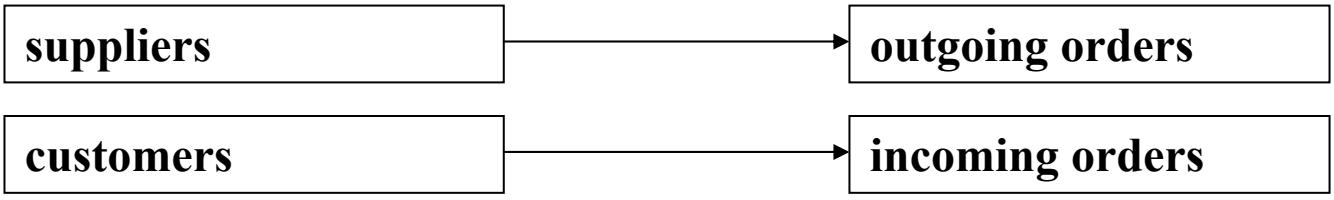
2 low degree, weak semantic



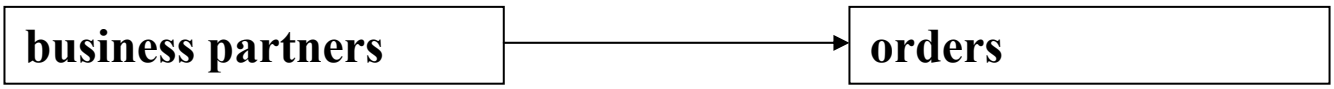
umbrella terms:



3 high degree, strong semantic



umbrella terms:



5.1 Data models: generic models; reference models

Two data models are analogous iff

(1) they are syntactically equal, that is
the structures of entity types and relationships are equal,
the mere diagrams without text coincide

(2) they are semantically analogous in the same degree, that is
syntactically corresponding entity types
are analogous in the same degree, that is
an umbrella term can be constructed
for each pair of corresponding entity types

Example

creditor	debtor	umbrella terms generic model
supplier groups ↓ suppliers ↓ outgoing orders ↓ order lines ↑ raw materials ↑ material groups	customer groups ↓ customers ↓ incoming orders ↓ order lines ↑ products ↑ product groups	business partner gr. ↓ business partners ↓ orders/contracts ↓ order lines ↑ items ↑ item groups

→ one-to-many relationship

5.1 Data models: What about partial analogies?

Complete model analogies are rare, that is syntactic equality is often not complete.

Example 1:

Number of order lines

orders with only one or with more order lines

customers library users	→	orders -	→	order lines borrow transactions	←	products books

Example 2:

Individual identifiability of items

individually identifiable items (library books, cars) or not individually identifiable items

borrow transactions order lines	←	books (copies) -	←	books (titles) products

5.2 Main functional areas of a company

Company management
Information management
Financial management, investments
Personnel management = human resources management
Accounting (ledger, accounts receivable, accounts payable)
Marketing, distribution, sales, order management
Materials management, inventory, purchasing, procurement
Production
Quality assurance/management
Product development, research and development
Customer support/service

Decomposition into smaller functional areas
which can be assigned to
employees (employee groups) in a matrix

6 References

pdf-files of my own publications: see my homepage.

Holl, Alfred:
Empirische Wirtschaftsinformatik und evolutionäre Erkenntnistheorie.
In: Becker, Jörg et al. (ed.): *Wirtschaftsinformatik und Wissenschaftstheorie. Bestandsaufnahme und Perspektiven*.
Wiesbaden: Gabler 1999, 163-207, ISBN 3-409-12002-5.
English translation on my homepage.

Holl, Alfred; Auerochs, Robert:
Analogisches Denken als Erkenntnisstrategie zur Modellbildung in der Wirtschaftsinformatik.
In: Frank, Ulrich (ed.): *Wissenschaftstheorie in Ökonomie und Wirtschaftsinformatik. Theoriebildung und –bewertung, Ontologien, Wissenmanagement.*
Wiesbaden: DUV 2004, 367-389, ISBN 3-8244-0738-8.