

Übung 5

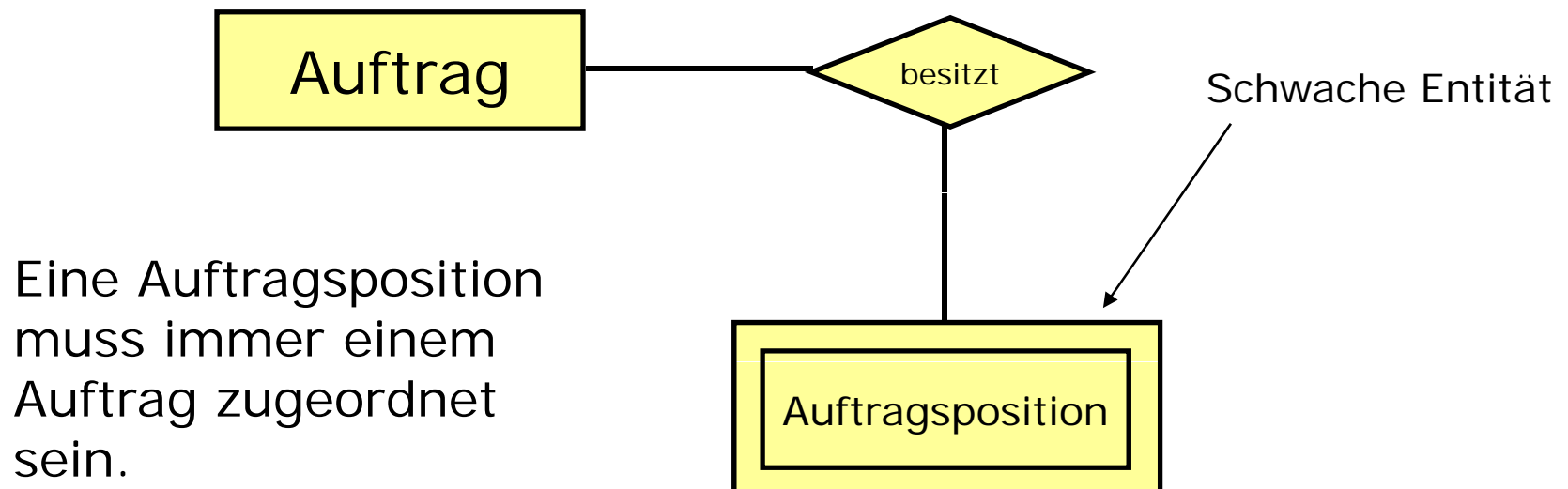
Entwicklung von Informationssystemen

Lösungshinweise

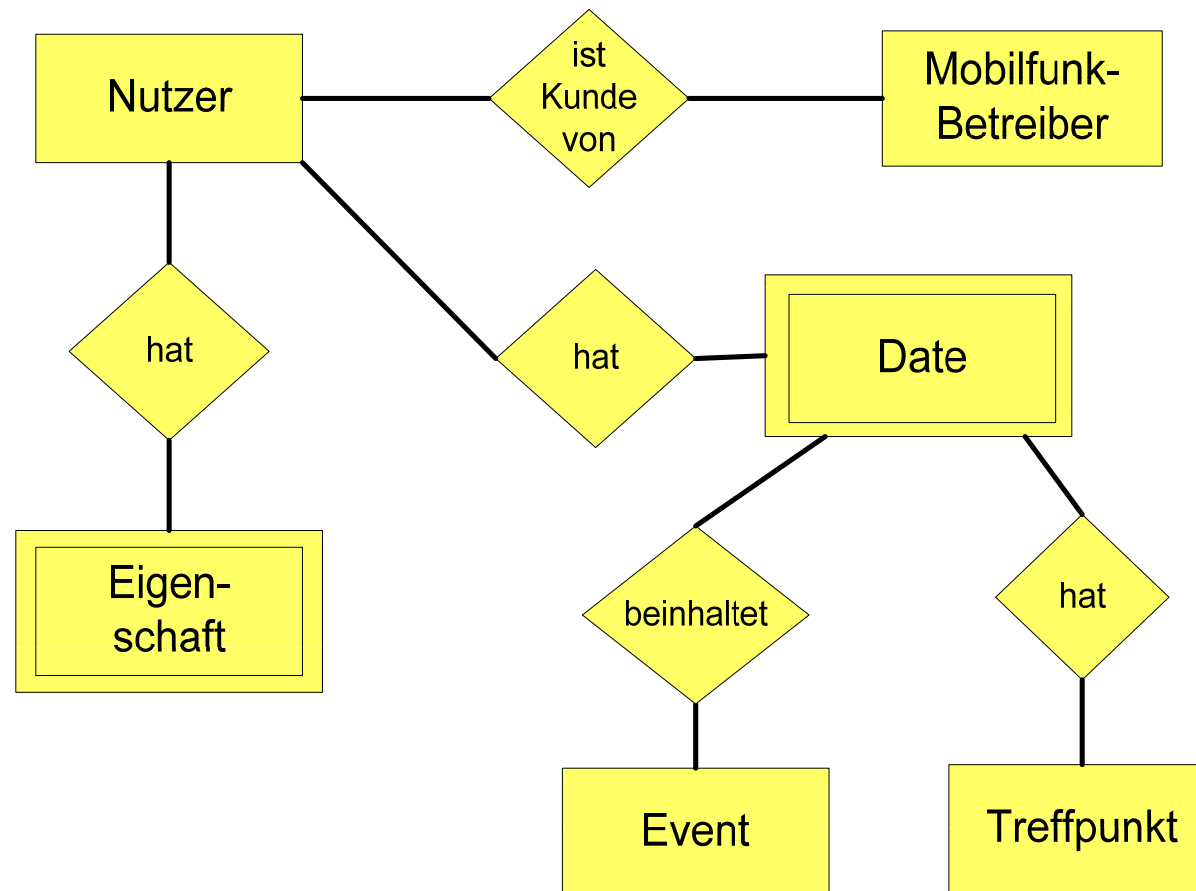
- 1. Wiederholung / Fragen Übung 4
- 2. Normalisierung
- 3. Beschreibungssprache – XML
- 4. SQL Einführung

- Schwache Entitäten

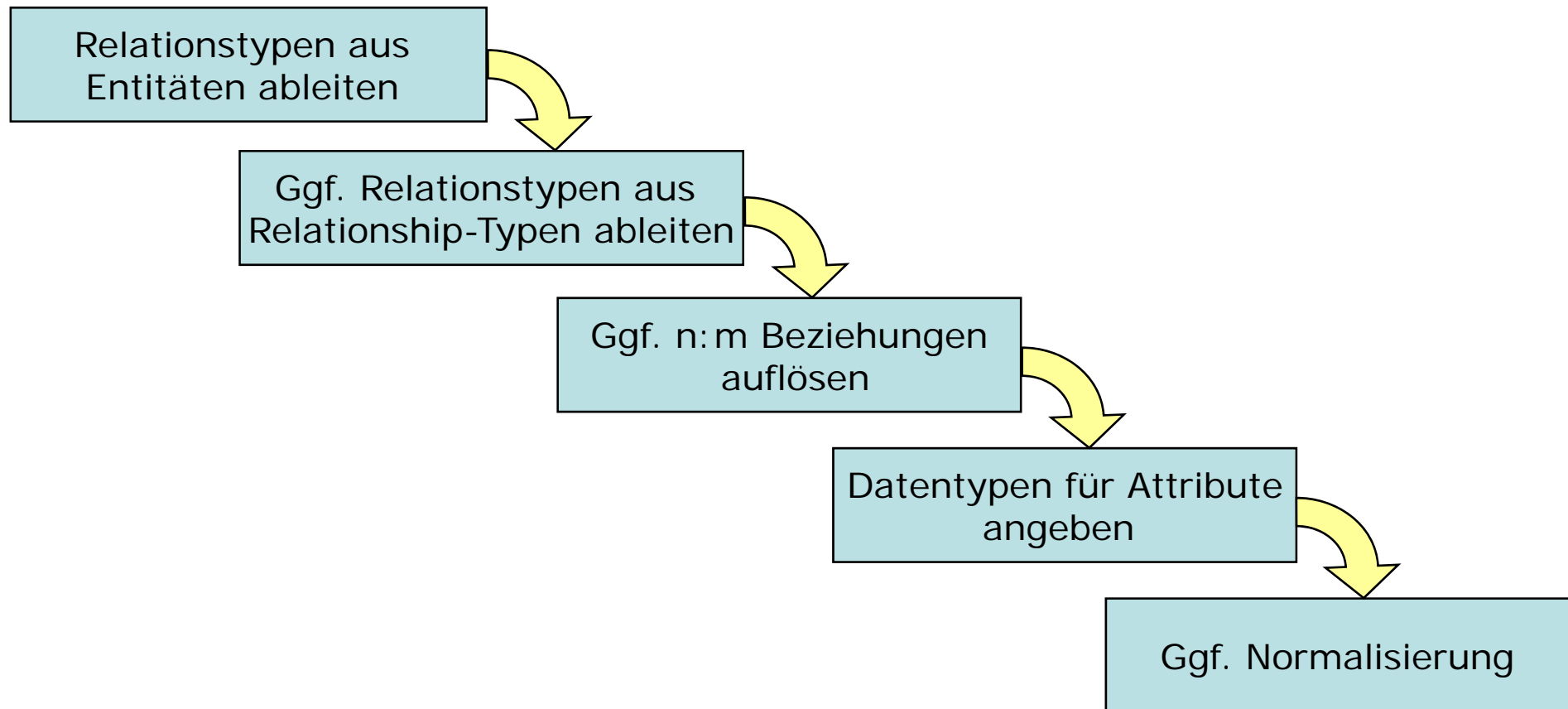
Schwache Entitäten sind abhängig von mindestens einer weiteren Entität und können somit nicht alleine existieren.



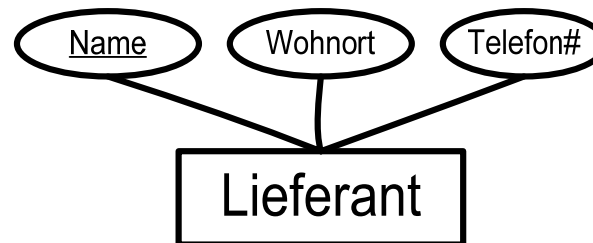
- Schwache Entitäten definieren



- 1. Wiederholung / Fragen Übung 4
- 2. Normalisierung
- 3. Beschreibungssprache – XML
- 4. SQL Einführung



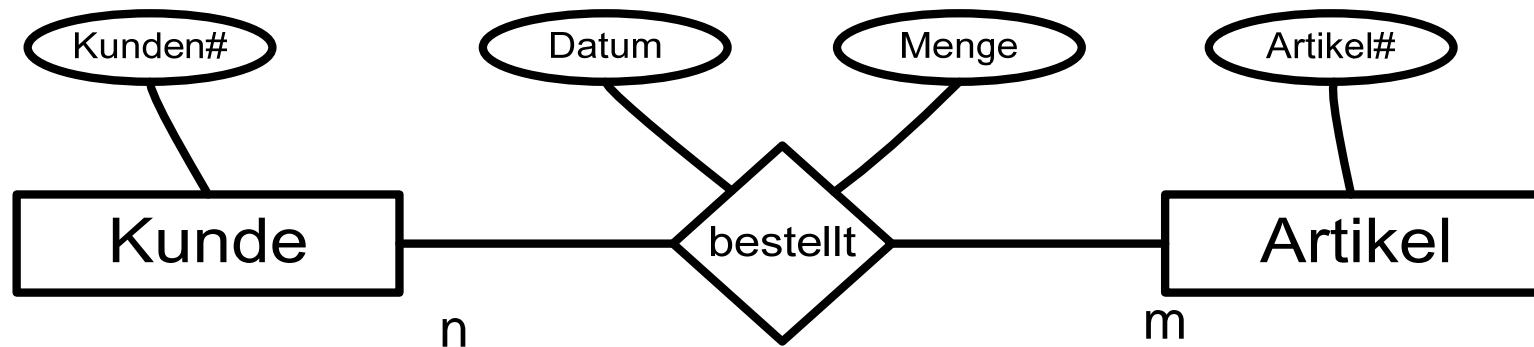
- Aus Entity-Typ wird Relationstyp mit den entsprechenden Attributen



Lieferant	<u>Name</u>	Wohnort	Telefon#

- Aus n:m-Relationship-Typ wird ein zusätzlicher Relationstyp.
 - Relation enthält die Schlüssel der beteiligten Entity-Typen als Attribute und zusätzlich die Attribute des Relationship-Typs.

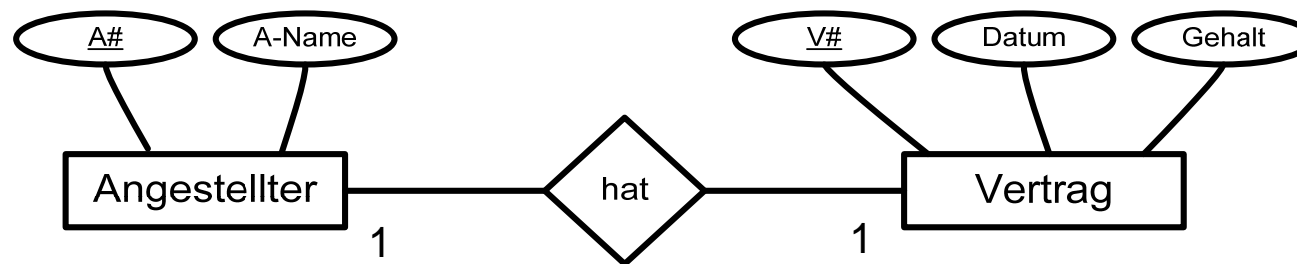
Beispiel:



bestellt	Kunden#	Artikel#	Datum	Menge

- Ein 1:1-Relationship-Typ wird i. allg. nicht zu einer eigenen Relation.
- Information wird an eine der beiden den betroffenen Entity-Typen entsprechenden Relationen „angehängt“.

Beispiel:



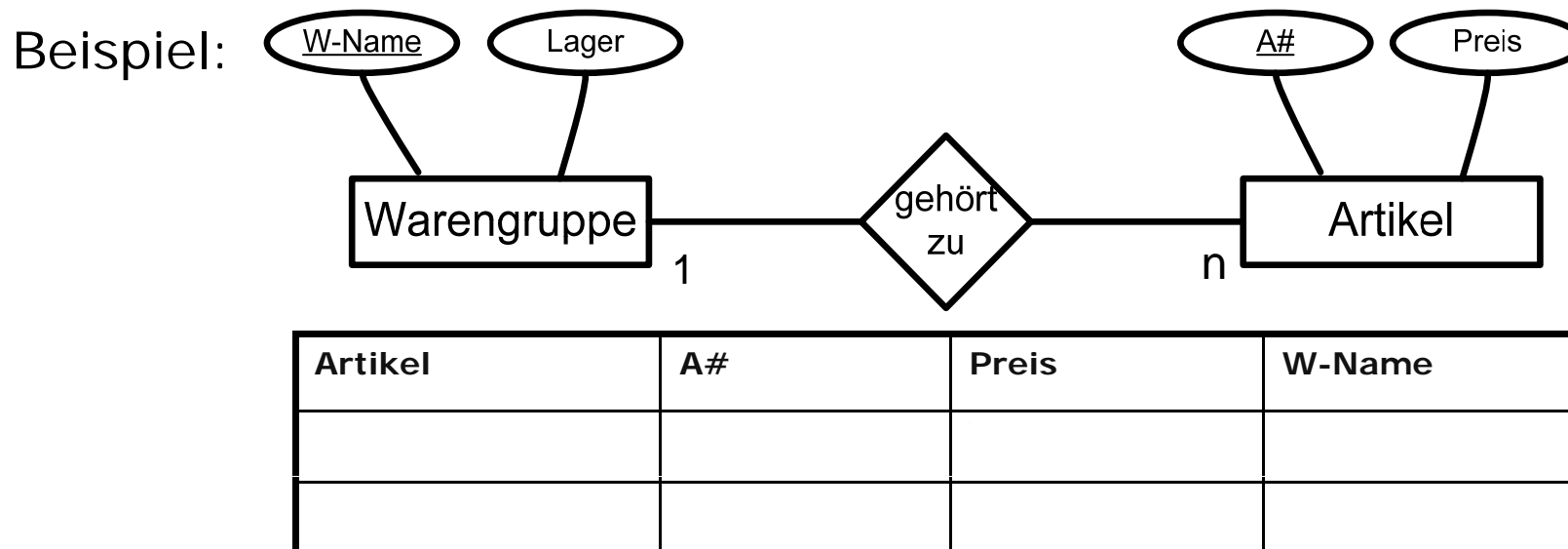
Alternative 1:

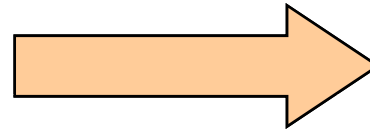
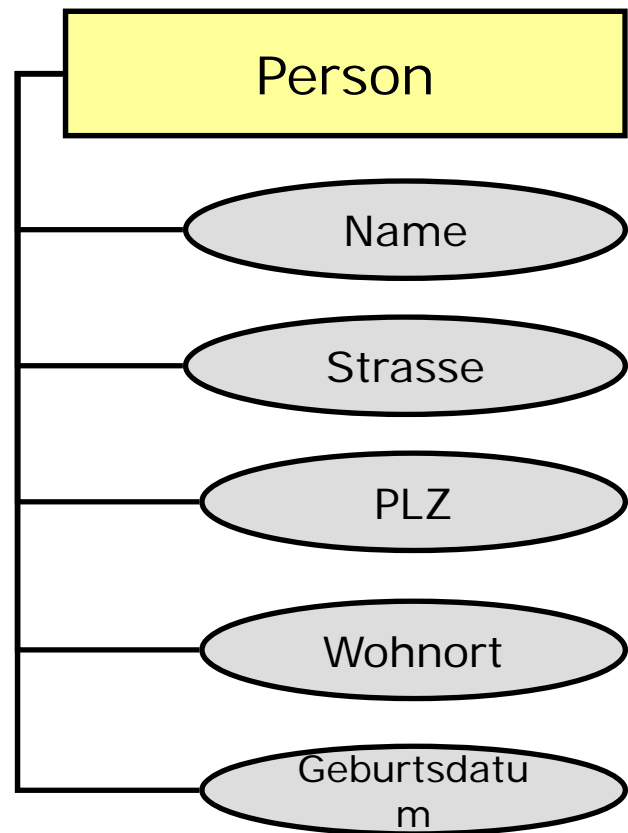
Angestellter	A#	A-Name	V#

Alternative 2:

Vertrag	V#	Datum	Gehalt	A#

- Ein 1:n-Relationship-Typ wird i. allg. nicht zu einer eigenen Relation.
- Information wird an die Relation "angehängt", die dem Entity-Typen an der mit n beschrifteten Kante entspricht.





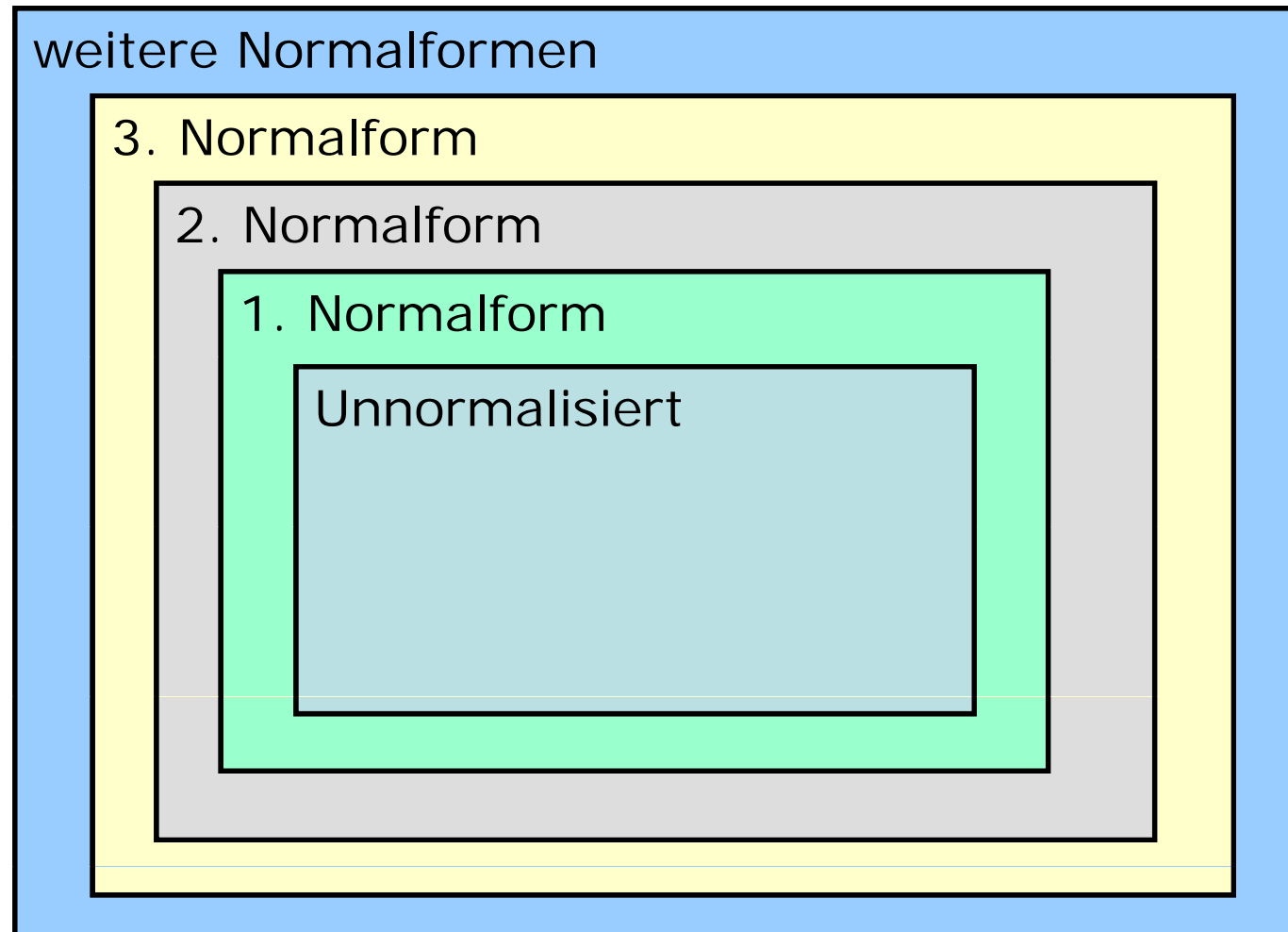
Relation „Person“

- Name: CHAR(30)
- Strasse: CHAR(40)
- PLZ: INT
- Wohnort: CHAR(30)
- Geburtsdatum: DATUM

- Ziele
 - Jede Relation enthält nur Daten einer Bedeutung.
 - Redundanzfreie Speicherung von Daten
 - Vermeidung von Anomalien verursacht durch Redundanzen und Inkonsistenzen

- Normalisierungsformen
 - Unnormalisierte Relationen
 - Normalisierte Relationen in der 1. - 3. Normalform (NF)
 - Weitere Normalformen (z.B. 4.+5. Normalform oder Boyce Codd)

- Vorgehen zur Normalisierung
 - Unnormalisiert -> 1. NF -> 2. NF -> 3. NF
 - Relationen abgeleitet aus ER-Modell sind bereits in 2. NF



Sign	Verfasser	Titel	Jahr	SID	Sachgebiet	Schlagwort
QH1	Ferstl, Sinz	Wirtschafts informatik	1993	GRD	Grundlagen	Informationssysteme
QH1	Ferstl	Wirtschafts -informatik	1993	GRD	Grundlagen	Objektmodellierung
QH1	Sinz	Wirtschafts -informatik	1993	GRD	Grundlagen	Objektmodellierung
QH2	Wirth	Modula-2	1989	PRG	Programmierung	Algorithmus, Modul
QH3	Wirth	Pascal	1990	PRG	Programmierung	Algorithmus
QH3	Wirth	Pascal	1990	PRG	Programmierung	Datenstruktur

Quelle: Ferstl, Sinz 2001

Eine Relation ist unnormalisiert (0. NF), wenn Attribute mit nicht-elementaren Wertebereichen vorhanden sind.

- Redundanz
 - Z.B. Mehrfacheinträge für die Zuordnung des Buches „Wirtschaftsinformatik“ zu der Signatur „QH1“.
- Einfügeanomalie
 - Z.B. das Sachgebiet kann nur aufgenommen werden, wenn ein Buch hierfür vorliegt.
- Änderungsanomalie
 - Z.B. Änderungen am Titel eines Buches sind mehrfach durchzuführen.
- Löschanomalie
 - Z.B. mit dem Löschen des letzten Buches wird auch das Sachgebiet selbst gelöscht.

1. Normalform (1. NF)

Sign	Verfasser	Titel	Jahr	SID	Sachgebiet	Schlagwort
QH1	Ferstl	Wirtschafts-informatik	1993	GRD	Grundlagen	Informationssysteme
QH1	Sinz	Wirtschafts-informatik	1993	GRD	Grundlagen	Informationssysteme
QH1	Ferstl	Wirtschafts-informatik	1993	GRD	Grundlagen	Objektmodellierung
QH1	Sinz	Wirtschafts-informatik	1993	GRD	Grundlagen	Objektmodellierung
QH2	Wirth	Modula-2	1989	PRG	Programmierung	Algorithmus
QH2	Wirth	Modula-2	1989	PRG	Programmierung	Modul
QH3	Wirth	Pascal	1990	PRG	Programmierung	Algorithmus
QH3	Wirth	Pascal	1990	PRG	Programmierung	Datenstruktur

Quelle: Ferstl; Sinz, 2001

Eine Relation ist in 1. NF, wenn die Wertebereiche aller Attribute elementar sind.

- Vereinbarungen von Bezeichnungen
 - $R(A_1, A_2, \dots, A_n)$ Relationstyp
 - A_1, A_2, \dots, A_n Attribute
 - $X, Y, Z \subseteq \{A_1, A_2, \dots, A_n\}$

- Funktionale Abhängigkeit
 - Y heißt *funktional abhängig* von X in R (X bestimmt Y funktional) $X \rightarrow Y$, wenn es in keiner Relation zwei Attribute gibt, die in ihrem Wert zu X , aber nicht in ihrem Wert zu Y übereinstimmen.

- Voll funktionale Abhängigkeit
 - Y heißt *voll funktional abhängig* von X in R (X bestimmt Y voll funktional) $X \bullet \rightarrow Y$, wenn $X \rightarrow Y$ gilt und X minimal ist, d.h. wenn es keine Attributmenge $Z \subset X$ gibt, so dass $Z \rightarrow Y$.

- Schlüsselkandidat
 - X heißt Schlüsselkandidat von R , falls $X \bullet \rightarrow A_1, A_2, \dots, A_n$.
 - Ein Schlüsselattribut ist ein Bestandteil eines Schlüsselkandidaten.

Quelle: Auf Basis von Ferstl; Sinz, 2001

- Funktionale Abhängigkeit (siehe Abbildung 1. Normalform)
 - Beispiele
 - Sign, Verfasser, Schlagwort → Titel, Jahr, SID, Sachgebiet
 - Sign, Verfasser → Titel, Jahr, SID, Sachgebiet
 - Bei Kenntnis von „Sign, Verfasser, Schlagwort“ bzw. „Sign, Verfasser“ lassen sich die zugehörigen Werte „Titel, Jahr, SID, Sachgebiet“ eindeutig bestimmen.

- Voll funktionale Abhängigkeit (siehe Abbildung 1. Normalform)
 - Beispiele
 - Sign •→ Titel, Jahr, SID, Sachgebiet
 - SID •→ Sachgebiet
 - „Sign“ ist die minimale Attributmenge, um „Titel, Jahr, SID, Sachgebiet“ eindeutig zu bestimmen. „Verfasser“ oder „Schlagwort“ allein genommen würden nicht ausreichen, um „Titel, Jahr, SID, Sachgebiet“ eindeutig zu bestimmen.
 - „SID“ ist die minimale Attributmenge, um „Sachgebiet“ eindeutig zu bestimmen.

- Schlüsselkandidaten (siehe Abbildung 1. Normalform)
 - Schlüsselkandidaten sind Attributkombinationen, von denen alle Attribute der Relation voll funktional abhängig sind.
 - Beispiel
 - Sign, Verfasser, Schlagwort •→ Sign, Verfasser, Titel, Jahr, SID, Sachgebiet, Schlagwort
 - Bei Kenntnis von „Sign, Verfasser, Schlagwort“ kann man **alle** Werte der Relation „Titel, Jahr, SID, Sachgebiet“ eindeutig bestimmen.

- Schlüsselattribute (siehe Abbildung 1. Normalform)
 - Ein Attribut, das Bestandteil eines Schlüsselkandidaten ist, heißt Schlüsselattribut.
 - Beispiel
 - Sign, Verfasser, Schlagwort sind Schlüsselattribute
 - Titel, Jahr, SID, Sachgebiet sind Nichtschlüsselattribute

- Aufspaltung der Relation in 1. NF in zwei neue Relationen in 2. NF
 - Die Relation in 1. NF ist so aufzuteilen, dass
 - alle Attribute aus der Relation in 1. NF auch in den Relationen in 2. NF vertreten sind;
 - in jeder der Relationen in 2. NF alle Nichtschlüsselattribute voll funktional vom Schlüsselkandidaten abhängig sind.
 - Es kann dabei Relationen geben, die nur aus Schlüsselattributen bestehen.
 - Die Relationen sind über ein Schlüsselattribut, welches in beiden Relationen vorhanden ist (Fremdschlüssel) miteinander zu verknüpfen.
 - Beispiel
 - Sign •→ Titel, Jahr, SID, Sachgebiet
 - Sign, Verfasser, Schlagwort •→ Sign, Verfasser, Schlagwort
 - „Sign“ ist der Fremdschlüssel.

2. Normalform (2. NF)

- Die Relation R1 ist in 2. Normalform, wenn sie in 1. NF ist und jedes Nichtschlüsselattribut (Titel, Jahr, SID, Sachgebiet) von dem Schlüsselkandidaten („Sign“) voll funktional abhängig ist.
- Die Relation R2 ist in 2. Normalform, wenn sie in 1. NF ist und jedes Nichtschlüsselattribut (-) von dem Schlüsselkandidaten („Sign, Verfasser, Schlagwort“) voll funktional abhängig ist.

- R1 -

<u>Sign</u>	Titel	Jahr	SID	Sachgebiet
QH1	Wirtschaftsinformatik	1993	GRD	Grundlagen
QH2	Modula-2	1989	PRG	Programmierung
QH3	Pascal	1990	PRG	Programmierung

- R2 -

<u>Sign</u>	<u>Verfasser</u>	<u>Schlagwort</u>
QH1	Ferstl	Informationssysteme
QH1	Sinz	Informationssysteme
QH1	Ferstl	Objektmodellierung
QH1	Sinz	Objektmodellierung
QH2	Wirth	Algorithmus
QH2	Wirth	Modul
QH3	Wirth	Algorithmus
QH3	Wirth	Datenstruktur

Anmerkung:
Schlüsselattribute sind unterstrichen.

Quelle: Auf Basis von Ferstl; Sinz, 2001

3. Normalform

Eine Relation R ist in 3. Normalform, wenn sie in 2. NF ist und kein Nichtschlüsselattribut transitiv von einem Schlüsselkandidaten abhängt.

Quelle: Auf Basis von Ferstl; Sinz, 2001

Aufgabe 1: Normalisierung

- Normalisierung hat die redundanzfreie Speicherung von Daten sowie die Vermeidung von Anomalien zum Ziel.
 - a) Nennen Sie eine Anomalie, welche bei einer nicht-normalisierten Datenbank auftreten kann, und geben Sie, mit Bezug auf das InstantONS[®] System, ein Beispiel an.
 - b) Gegeben sei beispielhaft die folgende Nutzer-Datentabelle aus dem InstantONS[®] System.

Nutzer ID	Pseudonym	Alter	Sport ID	Sportart	Leistung
101	Fritz1234	25	1	Fußball	Anfänger
102	Heinz4578	27	1	Fußball	Fortgeschritten
102	Heinz4578	27	2	Surfen	Anfänger
103	Anna1978	26	3	Tennis	Profi

Prüfen Sie, ob sich die Tabelle in der 2. Normalform befindet. Überführen Sie die Tabelle ggf. in die 2. Normalform.

- c) Wieso beschränkt man sich in der Praxis oft auf Normalisierung bis zur 3. Normalform?

- Z. B. Löschanomalie
- Löschanomalie in einer InstantONS[®] Datentabelle

Pseudonym	Alter	Sportart
Karl1234	34	Schwimmen
Willi0987	29	Radfahren
Franz4532	24	Tischtennis

Willi hat durch InstantONS[®] jetzt eine feste Beziehung und meldet sich ab. Sein Eintrag wird gelöscht. Beim Löschen wird auch der Sport „Radfahren“ entfernt und steht anderen Benutzern nicht mehr zur Verfügung.

- Z. B. Änderungsanomalie
- Änderungsanomalie in einer InstantONS[®] Datentabelle

Pseudonym	Date ID	Event	Kosten
Karl1405	1	Kino	6 €
Willi45	2	Theater	20 €
Helga233	1	Kino	6 €
Frieda4444	2	Theater	20 €

Bei Änderung der Kosten für Kino bzw. Theater müssen mehrere Datensätze geändert werden.

- Welche Eigenschaft, die Normalisierung verhindern soll, hat die obige Datentabelle noch?

- Normalisierung hat die redundanzfreie Speicherung von Daten sowie die Vermeidung von Anomalien zum Ziel.
- b) Gegeben sei beispielhaft die folgende Nutzer-Datentabelle aus dem InstantONS[®] System.

<u>Nutzer ID</u>	Pseudonym	Alter	<u>Sport ID</u>	Sportart	Leistung
101	Fritz1234	25	1	Fußball	Anfänger
102	Heinz4578	27	1	Fußball	Fortgeschritten
102	Heinz4578	27	2	Surfen	Anfänger
103	Anna1978	26	3	Tennis	Profi

Prüfen Sie, ob sich die Tabelle in der 2. Normalform befindet. Überführen Sie die Tabelle ggf. in die 2. Normalform.

- Aktuelle Normalform feststellen

Attribute

<u>Nutzer ID</u>	Pseudonym	Alter	<u>Sport ID</u>	Sportart	Leistung
101	Fritz1234	25	1	Fußball	Anfänger
102	Heinz4578	27	1	Fußball	Fortgeschritten
102	Heinz4578	27	2	Surfen	Anfänger
103	Anna1978	26	3	Tennis	Profi

- Datentabelle befindet sich in 1. Normalform, da Wertebereiche aller Attribute elementar sind.

- 2. Normalform
 - Eine Relation R ist in 2. Normalform, wenn sie in 1. NF ist und jedes Nichtschlüsselattribut von dem Schlüsselkandidaten voll funktional abhängig ist.

- Funktionale Abhängigkeit
 - Y heißt *funktional abhängig* von X in R (X bestimmt Y funktional) $X \rightarrow Y$, wenn es in keiner Relation zwei Attribute gibt, die in ihrem Wert zu X , aber nicht in ihrem Wert zu Y übereinstimmen.

- Voll funktionale Abhängigkeit
 - Y heißt *voll funktional abhängig* von X in R (X bestimmt Y voll funktional) $X \bullet \rightarrow Y$, wenn $X \rightarrow Y$ gilt und X minimal ist, d.h. wenn es keine Attributmengung $Z \subset X$ gibt, so dass $Z \rightarrow Y$.

- Schlüsselkandidat
 - X heißt Schlüsselkandidat von R , falls $X \bullet \rightarrow A_1, A_2, \dots, A_n$.
 - Ein Schlüsselattribut ist ein Bestandteil eines Schlüsselkandidaten.

- Aktuelle Normalform feststellen

<u>Nutzer ID</u>	Pseudonym	Alter	<u>Sport ID</u>	Sportart	Leistung
101	Fritz1234	25	1	Fußball	Anfänger
102	Heinz4578	27	1	Fußball	Fortgeschritten
102	Heinz4578	27	2	Surfen	Anfänger
103	Anna1978	26	3	Tennis	Profi

- Datentabelle befindet sich nicht in 2. Normalform, da nicht jedes Nichtschlüsselattribut von dem Schlüsselkandidaten voll funktional abhängig ist.

- Vorgehen zur Überführung in die 2. Normalform
 - Identifizieren der nicht voll-funktional abhängige Attribute
 - Z.B. „Sportart“ bereits durch „Sport ID“ eindeutig identifizierbar
 - Aufteilung der Relation bzw. Tabelle in mehrere kleinere Relationen bzw. Tabellen mit voll-funktional abhängigen Attributen

Aufgabe 1b: Normalisierung (5)

<u>Nutzer ID</u>	Pseudonym	Alter
101	Fritz1234	25
102	Heinz4578	27
103	Anna1978	26

<u>Sport ID</u>	Sportart
1	Fußball
2	Surfen
3	Tennis




<u>Nutzer ID</u>	<u>Sport ID</u>	Leistung
101	1	Anfänger
102	1	Fortgeschritten
102	2	Anfänger
103	3	Profi

- Normalisierung hat die redundanzfreie Speicherung von Daten sowie die Vermeidung von Anomalien zum Ziel.

Wieso beschränkt man sich in der Praxis oft auf Normalisierung bis zur 3. Normalform?

- In der Praxis wird i.d.R. nur die 1.-3. Normalform verwendet.
- Anzahl der Tabellen und die Komplexität eines Relationenmodells steigt mit höherer Normalform.
- Höherere Normalform senkt die Performance eines Datenbanksystems.
- Im praktischen Einsatz wird ein Trade-Off zwischen Normalform und Performance eines Datenbanksystems angestrebt.

- 1. Wiederholung / Fragen Übung 4
- 2. Normalisierung
- 3. Beschreibungssprache – XML
- 4. SQL Einführung

- Erläutern Sie die wesentlichen Unterschiede zwischen HTML und XML!

■ Merkmale von HTML

- HTML ist eine Seitenbeschreibungssprache, keine Datenbeschreibungssprache!
- Beim Erzeugen von HTML-Seiten aus Datenbanken ist erheblicher Aufwand nötig (Skripte, Formatierung, Darstellung).
- HTML eignet sich nicht zur Datenhaltung, da die Sprache sich nicht eignet, Daten zu beschreiben.
- Bei HTML hat sich unsaubere Syntax etabliert Elemente werden oft nicht abgeschlossen:

o ``

Erta Ale is a shield volcano, part of the East African Rift system.

``

- Merkmale von XML
 - Einfache und menschenlesbare Syntax
 - Standardisiert
 - Selbstbeschreibend durch enthaltene Meta-Beschreibung
 - Erweiterbar durch neue Elementbeschreibungen
 - > anwendungsspezifische Datenmodelle
 - Eignet sich zur Datenhaltung

- Geben Sie an, welche der folgenden InstantONS[®] - XML-Dokumente wohlgeformt sind!

- Mögliche Software-Unterstützung
 - Freeware-Software „Peter’s XML Editor“ für Windows
Internet: <http://www.iol.ie/~pxe>
- Färbt XML-Syntax ein, überprüft Syntaxfehler, validiert gegen DTDs.

```
<?xml version="1.0"?>
<Nutzer>
  <Pseudonym>
    Jenny23
  </Pseudonym>
  <Mobilfunker>
    t-mobile
  </Mobilfunker>
  <Eintritt>
    03.02.2007
  </Eintritt>
  <LetzteNutzung>
    29.04.2007
  </LetzteNutzung>
</Nutzer>
```

```
<?xml version="1.0"?>
<Nutzer>
  <Pseudonym>
    Joe1976
  </Pseudonym>
  <Mobilfunker>
    vodafone
  </Mobilfunker>
  <Eintritt>
    03.02.2007
  </Eintritt>
  <LetzteNutzung>
    29.04.2007
  </LetzteNutzung>
</Nutzer>
```

```
<?xml version="1.0"?>
<Nutzer>
  <Pseudonym>
    Jenny23
  </Pseudonym>
  <Mobilfunker>
    t-mobile
  </Mobilfunker>
  <Eintritt>
    03.02.2007
  </Eintritt>
  <LetzteNutzung>
    29.04.2007
  </LetzteNutzung>
</Nutzer>
```



```
<?xml version="1.0"?>
<Nutzer>
  <Pseudonym>
    Joe1976
  </Pseudonym>
  <Mobilfunker>
    vodafone
  </Mobilfunker>
  <Eintritt>
    03.02.2007
  <LetzteNutzung>
    29.04.2007
  </Nutzer>
```

```

<?xml version="1.0"?>
<Date>
<Ort>
    Fressgass 17, Frankfurt
</Ort>
<Zeit>
    25.03.2007, 21:15-0:15
</Zeit>
<Treffpunkt>
    Starbucks
<Personen>
    Gina
</Treffpunkt>
</Personen>
<Personen>
    Jimmy
</Personen>
<Aktivität>
    Cocktails
</Aktivität>
<Kommentar>
    Gina 2 Caipis ausgegeben
</Kommentar>
</Date>

```

```

<?xml version="1.0"?>
<Date>
<Ort>
    Schweizer Strasse, Frankfurt
</Ort>
<Zeit>
    25.03.2007, 21:15-0:15
</Zeit>
<Treffpunkt>
    Apfelwein-Wagner
</Treffpunkt>
<Personen>
    Pit
</Personen>
<Personen>
    Jenny23
</Personen>
<Aktivität>
    Äppler trinken
</Aktivität>
<Kommentar>
    Pit hatte coole Sonnenbrille an!
</Kommentar>
</Date>

```

```
<?xml version="1.0"?>
<Date>
<Ort>
    Fressgass 17, Frankfurt
</Ort>
<Zeit>
    25.03.2007, 21:15-0:15
</Zeit>
<Treffpunkt>
    Starbucks
<Personen>
    Gina
</Treffpunkt>
</Personen>
<Personen>
    Jimmy
</Personen>
<Aktivität>
    Cocktails
</Aktivität>
<Kommentar>
    Gina 2 Caipis ausgegeben
</Kommentar>
</Date>
```

```
<?xml version="1.0"?>
<Date>
<Ort>
    Schweizer Strasse, Frankfurt
</Ort>
<Zeit>
    25.03.2007, 21:15-0:15
</Zeit>
<Treffpunkt>
    Apfelwein-Wagner
</Treffpunkt>
<Personen>
    Pit
</Personen>
<Personen>
    Jenny23
</Personen>
<Aktivität>
    Äppler trinken
</Aktivität>
<Kommentar>
    Pit hatte coole Sonnenbrille an!
</Kommentar>
</Date>
```



- Erklären Sie den Zweck einer XML-DTD!

- Eine Document Type Definition (DTD) beschreibt Struktur und Grammatik von XML-Dokumenten.
- Ist vergleichbar mit einer Variablen-/ Typendeklaration in einer Programmiersprache
- Es wird definiert, welche Werte in Elementen vorkommen dürfen, damit ein „gültiges“ XML-Dokument entsteht.
- DTD „übersetzt“ also ein XML-Dokument in Datentypen für Datenbanken und erzeugt Regeln für Elemente.

- Erklären Sie, was ein validiertes XML-Dokument ist!

- Ein „validiertes“ oder „gültiges“ XML-Dokument ist ein wohlgeformtes XML-Dokument, ...
 - dessen Inhalte vollständig den Vorgaben einer DTD oder eines Schemas entsprechen.
- Validiert wird also immer gegen eine DTD oder gegen ein Schema.

- InstantONS[®] hat einen Wettbewerber gekauft und möchte dessen Daten importieren. Leider sind die Daten geringfügig anders strukturiert.
- Validieren Sie das unten stehende Wettbewerber-Dokument gegen die angegebene DTD von InstantONS[®]. Sollten Fehler auftreten, so verbessern Sie die DTD so, dass Sie die neuen Daten ohne Verluste einlesen können.

- Element-Content:

EMPTY	leeres Element
ANY	beliebiger Inhalt
	Auswahlliste
,	Sequenz
()	Gruppierung
(#PCDATA)	Zeichen- oder Stringdaten

- Kardinalität:

	leer: genau ein Wert nötig
+	mindestens ein Wert
?	Null oder ein Wert
*	Null oder mehr Werte

- Deklaration der Elementregeln in einer DTD:

<!ELEMENT flirt	(name,telefon,email,stadt,kontakte, geburtstag,gebunden) >	
<!ELEMENT name	(&#PCDATA) >	← Text
<!ELEMENT telefon	(mobil zuhause arbeit) + >	
<!ELEMENT mobil	(&#PCDATA) >	↙ Auswahlliste
<!ELEMENT zuhause	(&#PCDATA) >	
<!ELEMENT email	(&#PCDATA) >	
<!ELEMENT stadt	(&#PCDATA) >	
<!ELEMENT kontakte	(&#PCDATA) >	
<!ELEMENT geburtstag	(&#PCDATA) >	
<!ELEMENT gebunden	(&#PCDATA) >	

```

<Date>
  ... (gekürzt)
<Personen>
  Jenny23
</Personen>
<Aktivität>
  Äppler trinken
</Aktivität>
<Erfolg>
  <Kompliment>ja</Kompliment>
  <Einladung>nein</Einladung>
  <Neuesdate>nein</Neuesdate>
</Erfolg>
<Kommentar>
  Pit hatte coole
  Sonnenbrille an!
</Kommentar>

```

```

<?xml version="1.0"?>

<!DOCTYPE Date [
<!ELEMENT Date
(Ort, Zeit, Treffpunkt, Personen+,
Aktivität, Kommentar)>

<!ELEMENT Ort (#PCDATA)>
<!ELEMENT Zeit (#PCDATA)>
<!ELEMENT Treffpunkt (#PCDATA)>
<!ELEMENT Personen (#PCDATA)>
<!ELEMENT Aktivität (#PCDATA)>
<!ELEMENT Kommentar (#PCDATA)>
]>

```

```

<Date>
  ... (gekürzt)
<Personen>
  Jenny23
</Personen>
<Aktivität>
  Äppler trinken
</Aktivität>
<Erfolg>
  <Kompliment>ja</Kompliment>
  <Einladung>nein</Einladung>
  <Neuesdate>nein</Neuesdate>
</Erfolg>
<Kommentar>
  Pit hatte coole
  Sonnenbrille an!
</Kommentar>

```

```

<?xml version="1.0"?>
<!DOCTYPE Date
[
<!ELEMENT Date
(Ort, Zeit, Treffpunkt, Personen+,
Aktivität, Erfolg?, Kommentar
)>
<!ELEMENT Ort (#PCDATA)>
<!ELEMENT Zeit (#PCDATA)>
<!ELEMENT Treffpunkt (#PCDATA)>
<!ELEMENT Personen (#PCDATA)>
<!ELEMENT Aktivität (#PCDATA)>
<!ELEMENT Erfolg
(Kompliment?, Einladung?,
Neuesdate?)
>
<!ELEMENT Kompliment (#PCDATA)>
<!ELEMENT Einladung (#PCDATA)>
<!ELEMENT Neuesdate (#PCDATA)>
<!ELEMENT Kommentar (#PCDATA)>
]>

```

- 1. Wiederholung / Fragen Übung 4
- 2. Normalisierung
- 3. Beschreibungssprache – XML
- 4. SQL Einführung

- a) Was bedeutet SQL?

- b) Stellen Sie die Struktur des SELECT-Befehls und dessen Bedeutung dar.

- SQL steht für = Structured Query Language
 - Mitte der 70er Jahre entwickelt
 - Zurzeit Standard für relationale Datenbanken,
 - o ANSI (American National Standards Institute)
 - o ISO (International Standardization Organization),
 - o Norm für SQL 2 liegt vor, Weiterentwicklung SQL 3

- Struktur der Grundelemente
 - Struktur:
 - o SELECT Attribut(e)
 - o FROM Relation(en)
 - o [WHERE Bedingung]
 - o [GROUP BY Attribut(e)]
 - o [ORDER BY Attribut(e)]
 - Bedeutung:
 - o Suche Attributwerte
 - o in Relation(en)
 - o [wobei gilt: Bedingung]
 - o [Aggregation nach ...]
 - o [Sortierung nach ...]

Offene Fragen ?