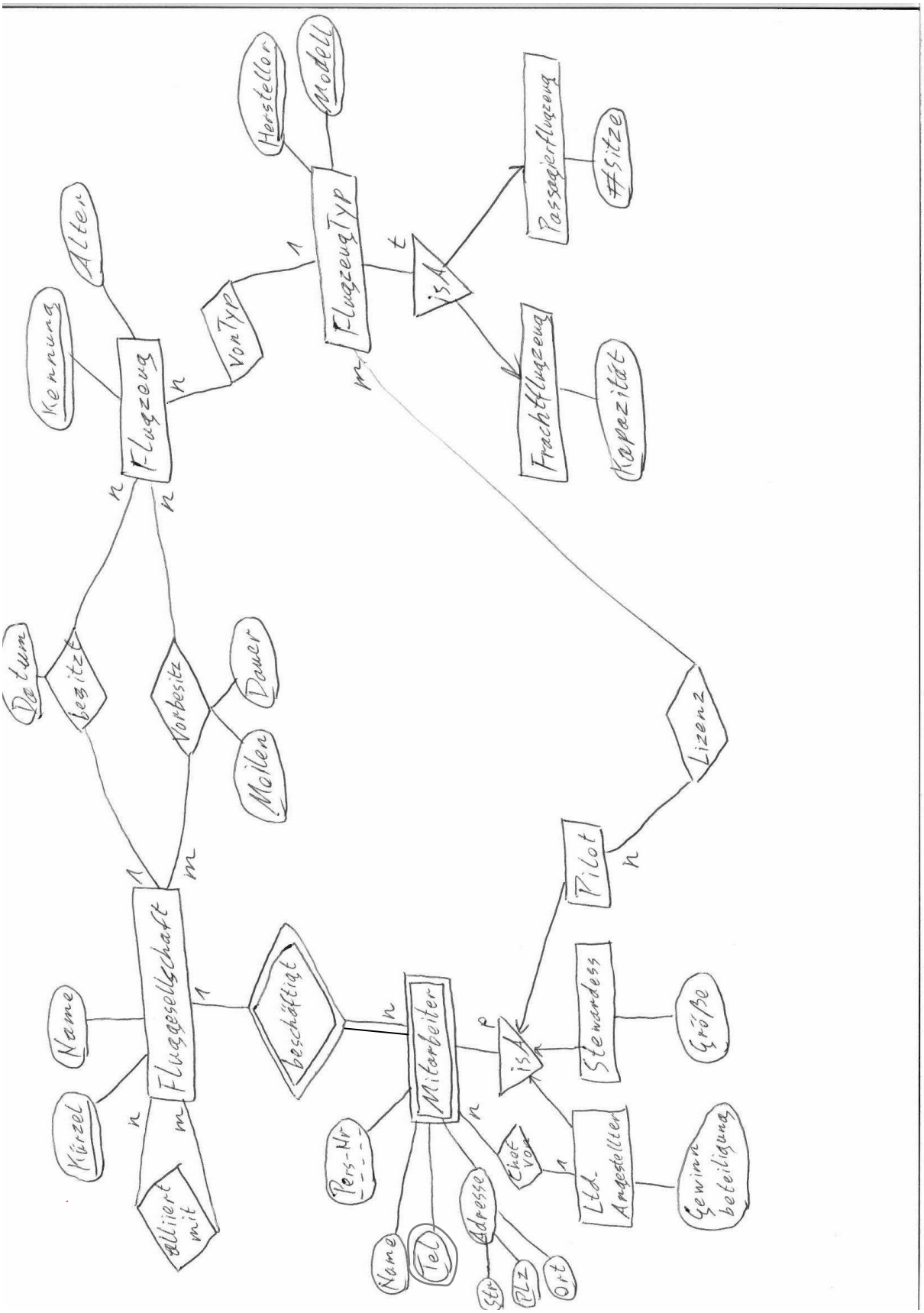
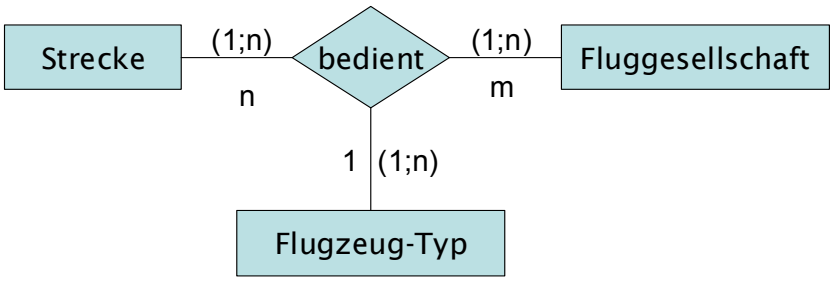
Aufgabe 1:

a)


Datum
besitzt
Vorbesitz
Dauer
Meilen
Kennung
Alter
Flugzeug
VonTyp
Hersteller
Modell
FlugzeugTyp
isA
Passagierflugzeug
#Sitze
Frachtflugzeug
Kapazität
Name
Kürzel
Fluggesellschaft
alliiert mit
beschäftigt
Mitarbeiter
Pers-Nr
Name
Tel
Adresse
Str
PLZ
Ort
Chef von
Ltd. Angestellter
isA
Stewardess
Größe
Pilot
Lizenz
Gewinnbeteiligung

b)



Ausdrucksstärker ist hier die 1:n Notation, da nur sie die Bedingung für die Beziehung zwischen (Strecke, FG) und FT beschreiben kann (Aussage 2)

Aufgabe 2:

a) Abbildung auf ein Relationales Schema

```
Verlag = ({name, plz, ort, str}, {name → ALL})
Cd = ({asin, titel, verlagName, erscheinungsJahr}, {asin → ALL})
SoloCd = ({asin, aufgenommenVon}, {asin → ALL})
GruppenCd = ({asin, aufgenommenVon}, {asin → ALL})
Musiker = ({mid, name, tel}, {mid → ALL})
MusikerEmail  = ({mid, email},{{mid, email} → ALL})
MusikGruppe = ({gid, name, gründungsJahr, sänger}, {gid → ALL})
MitgliedVon = ({mid, gid}, {{mid, gid} → ALL})
```

Interrelationale Abhängigkeiten:

```
CD[verlagName] ⊆ Verlag[name]
SoloCd[asin] ∪ GruppenCD[asin] = Cd[asin]
SoloCd[asin] ∩ GruppenCd[asin] = ∅
SoloCd[aufgenommenVon] ⊆ Musiker[mid]
GruppenCd[aufgenommenVon] ⊆ MusikGruppe[gid]
MusikerEmail[mid] ⊆ Musiker[mid]
MusikGruppe[sänger] ⊆ Musiker[mid]
MitgliedVon[mid] ⊆ Musiker[mid]
MitgliedVon[gid] ⊆ MusikGruppe[gid]
```

b) Tupelkalkül

```
{m | (∃ b) (∃ sa) (∃ sa_a) (∃ rl)
     Musiker(m) ∧ MusikGruppe(b) ∧ b[sänger] = m[mid] ∧
     SoloCD(sa) ∧ sa[aufgenommenVon] = m[mid] ∧
     CD(sa_a) ∧ sa_a[asin] = sa[asin] ∧
     Verlag(rl) ∧ sa_a[verlagName] = rl[name] ∧
     rl[stadt] = 'Aachen'
}
```

c) Relationale Algebra

Gruppen, die eine CD bei Rockstar Records haben – Gruppen deren Mitglieder eine SoloCd haben. Verbund mit Musikgruppe um deren Namen zu erhalten. Dann Projektion auf Name.

$$\Pi_{Name}\Big(\text{MusikGruppe} \underset{gid}{\bowtie}_{gid} \big(\Pi_{gid}(\text{GruppenCD} \underset{asin}{\bowtie}_{asin} \sigma_{verlagName='Rockstar\ Records'}(\text{CD})) - \Pi_{gid}(\text{MitgliedVon} \underset{mid}{\bowtie}_{aufgenommenVon} \text{SoloCD})\big)\Big)$$

oder:

$$\Pi_{Name}\Big(\Pi_{Name,\ gid}(\text{MusikGruppe} \underset{gid}{\bowtie}_{aufgenommenVon} \text{GruppenCD} \underset{asin}{\bowtie}_{asin} \sigma_{verlagName='Rockstar\ Records'}(\text{CD})) - \Pi_{Name,\ gid}(\text{MusikGruppe} \underset{gid}{\bowtie}_{gid} \text{MitgliedVon} \underset{mid}{\bowtie}_{aufgenommenVon} \text{SoloCD})\Big)$$

oder …

c) SQL

```
SELECT cd.titel, v.name, v.ort
FROM SoloCd scd, CD cd, Verlag v, Musiker m
WHERE
  m.name = ,Keith Caputo' AND
  m.mid = scd.aufgenommenVon AND
  scd.asin = cd.asin AND
  cd.verlagName = v.name AND
  EXISTS ( SELECT saenger FROM MusikGruppe WHERE saenger = m.mid )
ORDER BY cd.erscheinungsjahr DESC;
```

```
SELECT v.name, g.name, COUNT(*)
FROM
  CD cd, GruppenCd gcd, Gruppe g, Verlag v
WHERE
  (v.name = cd.verlagName) AND
  (gcd.asin = cd.asin) AND
  (gcd.aufgenommenVon = g.gid) AND
  (cd.erscheinungsJahr < 2006) AND
  (g.saenger NOT IN (SELECT aufgenommenVon FROM SoloCd))
GROUP BY v.name, g.name
```

Aufgabe 3:

a) I kommt auf keiner rechten Seite vor, muss also Teil jedes Schlüssels sein.
H kann nur abgeleitet werden, wenn D bekannt ist.
D kann nur abgeleitet werden, wenn H bekannt ist.

   a. HI → DHI → BCDHI → BCDEGHI → ABCDEGHI (also Superschlüssel, minimal siehe oben)
   b. DI → DHI → … (also Superschlüssel, minimal siehe oben)

Es gibt keine anderen Schlüsselkandidaten, denn:

- A und E stehen nie links
- BCI → ABCI (es ist keine weitere FD anwendbar.
  Hinzufügen von H oder D führt zu einem nicht minimalen Schlüsselkandidaten.

b) 1NF: klar
2NF: nein, da HI → BC nicht links-minimal ist.

c)

   a. <u>R1</u> = (CDEG, {CD→EG})
      R2 = (ABCDHI {BC→A, HI→D, D→H, H→BC})
   b. <u>R21</u> = (ABC, {BC→A})
      R22 = (BCDHI, {HI→D, D→H, H→BC})
   c. <u>R221</u> = (BCH, {H→BC})
      R222 = (DHI, {HI→D, D→H})
   d. <u>R2211</u> = (DH, {D→H})
      <u>R2212</u> = (DI, {})

D = {R1, R21, R221, R2211, R2212}

Aufgabe 4:

1) mache F r-minimal:
   B1 = { A → D, A → G, C → B, AC → D, AC → E, B → E, G → A, G → B, G → E }
2) mache B1 l-minimal:
   a. AC → D ist nicht l-minimal wegen A → D,
      AC → D kann entfernt werden da A→D bereits in B1
   b. AC → E ist nicht l-minimal:
      A → G → E (also, A → E)
      (oder auch C → B → E (also C → E)
      (A → E ist redundant (siehe 3) aber der 2. Schritt erzeugt dennoch zunächst die FD)

B2 = { A → D, A → G, B → E, C → B, A → E, G → A, G → B, G → E }

3) redundante FDs entfernen
   A → E: A → G → E (entferne A → E)
   G → E: G → B → E (entferne G → E)

Ergebnis von BASIS(F): B = { A→D, A→G, B→E, C→B, G→A, G→B }

Aufgabe 5:

-0,5 pro falscher Antwort, insgesamt nicht weniger als 0 Punkte

1. Bei binären Relationship-Typen ist die (min, max)-Notation für Kardinalitätsrestriktionen präziser und damit ausdrucksstärker als die 1:n-Notation.
   - richtig, bei binären Relationship-Typen entspricht max in der (min, max)-Notation dem Wert in 1:n am gegenüberliegenden Ende des Relationship-Typs
2. Bei der Anwendung des in der Vorlesung vorgestellten Dekompositionsalgorithmus' gehen stets funktionale Abhängigkeiten verloren.
   - falsch, wenn die FDs die BCNF nicht verletzen werden auch keine Abhängigkeiten verloren gehen.
3. Die vertikale Fragmentierung beim Entwurf verteilter Datenbanken entspricht einer Projektion.
   - Richtig
4. Ein serialisierbarer Schedule heißt auch seriell.
   - Falsch
5. Strikte Schedules vermeiden Dirty-Read-Situationen.
   - richtig, dirty read wird schon von RC vermieden
6. Zwei Datenoperationen in einem Schedule stehen in Konflikt wenn sie zu verschiedenen Transaktionen gehören und auf demselben Objekt arbeiten.
   - falsch, eine der Operationen muss außerdem eine Schreiboperation sein
7. Im Falle eines Systemfehlers führt der Recovery Manager ein REDO für alle Transaktionen durch, die zum Zeitpunkt des Fehlers noch aktiv waren.
   - falsch, für diese TXen muss UNDO durchgeführt werden
8. Ein valides XML Dokument ist immer einem DTD oder einem XML Schema zugeordnet.
   - richtig, Validitität ist immer in Bezug zu einem Schema

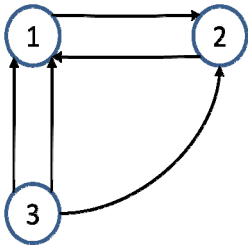
Aufgabe 6:

a) conf(s1) = { (r1(x), w3(x)), (r1(x), w2(x))
   (r2(y), w1(y)),
   (w3(x), w1(x)), (w3(x), r2(x)), (w3(x), w2(x)),
   (w1(x), r2(x)), (w1(x), w2(x)) }
   conf(s2) = { (r1(x), w3(x)), (r1(x), w2(x)),
   (w3(x), w1(x)), (w3(x), r2(x)), (w3(x), w2(x)),
   (r2(y), w1(y)), (w1(x), r2(x)), (w1(x), w2(x)) }
   op(s1) = op(s2) und conf(s1) = conf(s2) also sind s1 und s2 konfliktäquivalent

b) Die beiden Schedules haben den gleichen Konfliktgraphen, dieser enthält aber Zykel, also sind die Schedules nicht Konfliktserialisierbar.

Der Graph ist

E = { (1, 2), (2, 1), (1, 3), (3, 1), (3, 2) }



c) s1 ist in ST, s2 ist in ACA

a. s ist in RC, wenn keine TX freigegeben wird, bis alle TXen von denen sie gelesen hat freigegeben sind.

i. s1: (w3(x), r2(x)) $\in$conf(s1) und c3 $<_{s1}$ c2
(w1(x), r2(x)) $\in$conf(s1) und c1 $<_{s1}$ c2
also ist s1 in RC

ii. s2: (w3(x), r2(x)) $\in$conf(s2) und c3 $<_{s2}$ c2
(w1(x), r2(x)) $\in$conf(s2) und c1 $<_{s2}$ c2
also ist s2 in RC

b. s ist in ACA, wenn eine TX nur Werte von erfolgreich abgeschlossenen TXen liest. Die gleichen Konflikte sind relevant.

i. s1: c3 $<_{s1}$ r2(x)
c1 $<_{s1}$ r2(x)
also ist s1 in ACA

ii. s2: c3 $<_{s2}$ r2(x)
c1 $<_{s2}$ r2(x)
also ist s2 in ACA

c. s ist in ST, wenn eine TX nur Werte von erfolgreich abgeschlossenen TXen liest oder schreibt.

i. s1: (w1(x), w2(x)) $\in$conf(s1) und c1 $<_{s1}$ w2(x)
(w3(x), w1(x)) $\in$conf(s1) und c3 $<_{s1}$ w1(x)
(w3(x), w2(x)) $\in$conf(s1) und c3 $<_{s1}$ w2(x)
also ist s1 in ST

ii. s2: (w3(x), w1(x)) $\in$conf(s2) und c3 $<_{s2}$ w1(x) !
also ist s2 nicht in ST

Aufgabe 7:

a) /pubdb/papier[@jahr="2000"]/titel

/pubdb/papier/authoren/person[@name="Alon Halevy"]/parent::node()/parent::node()
/pubdb/papier/authoren/person[@name="Alon Halevy"]/../..
/pubdb/papier/[authoren/person[@name="Alon Halevy"]]

```
<publikationen>
  { for $p in doc(„pubdb.xml“)/pubdb/papier
    where $p/@jahr = „2007“
    return
      <papier konferenz=“{$p/konferenz}“ jahr=“2007“>
        <titel wert=“{$p/titel}“ />
        <erstauthor>
          {$p/authoren/person[1]/@name}
        </erstauthor>
      </papier>
  }
</publikationen>
```