Klausur Datenbanken
Wintersemester 2012/2013
Prof. Dr. Wolfgang May
6. Februar 2013, 14-16 Uhr
Bearbeitungszeit: 90 Minuten

Vorname:

Nachname:

Matrikelnummer:

Studiengang:

Bei der Klausur sind **keine Hilfsmittel** (Skripten, Taschenrechner, etc.) erlaubt. Handies müssen ausgeschaltet sein. Papier wird gestellt. Benutzen Sie nur die **ausgeteilten**, zusammengehefteten **Blätter** für Ihre Antworten. Schreiben Sie mit blauem/schwarzem Kugelschreiber, Füller, etc.; Bleistift ist nicht erlaubt.

Zum **Bestehen** der Klausur sind **45** Punkte hinreichend.

☐ meine Note soll mit Matrikelnummer so bald wie möglich auf der Vorlesungs-Webseite veröffentlicht werden.

☐ meine Note soll nicht veröffentlicht werden; ich erfahre sie dann aus FlexNever oder beim zuständigen Prüfungsamt.

	Max. Punkte	Schätzung für "4"
Aufgabe 1 (ER-Modell)	15	11
Aufgabe 2 (Transformation in das Relationale Modell)	17	12
Aufgabe 3 (SQL und Relationale Algebra)	45	29
Aufgabe 4 (Verschiedenes)	13	6
Summe	90	58

Note:

Themenstellung: Frachtschiff-Reederei

Alle Klausuraufgaben basieren auf einem gemeinsamen "Auftrag": In der Klausur soll eine Datenbank einer Frachtschiff-Reederei, für deren Schiffsbestand und Routen, entworfen werden. Die Datenbank soll nicht selbständig planen, sondern die Planung macht wie im Speditionsgeschäft üblich ein Disponent, der dann die Daten in der Datenbank einträgt.

1. Die Reederei besitzt viele Schiffe, darunter Massengutfrachter, Containerschiffe, und Tankschiffe; weitere Arten könnten später dazukommen.

2. Jedes Schiff hat einen eindeutigen Namen. Weiterhin sind die folgenden Daten relevant: Tiefgang (beladen, in Metern), Länge und Breite (in Metern), sowie maximale Nutzlast (in Tonnen). Für Containerschiffe wird ausserdem die maximale Anzahl transportierbarer (20-Fuss-)Container angegeben. Tankschiffe können ebenfalls eine (relativ kleine) Anzahl Container transportieren.

   - Die *Corn Flake* ist ein Massengutfrachter mit 11m Tiefgang, 280m Länge, 32m Breite und 70.000t Nutzlast.
   - die *Tropicana* ist ein Containerfrachter mit 11m Tiefgang, 280m Länge, 32m Breite mit 60.000t Nutzlast, die bis zu 4800 Container transportieren kann.
   - Die *Quick & Dirty* ist ein Tankschiff mit 22m Tiefgang, 370m Länge, 64m Breite mit 390.000t Nutzlast und 50 Containerplätzen.

3. Frachthäfen: Jeder Frachthafen hat einen eindeutigen Namen und ist einem Land sowie einem Ozean zugeordnet. Ausserdem ist der maximale erlaubte Tiefgang in der Datenbank abgelegt.

   - *Rotterdam*, *Niederlande*, am *Atlantik*, Schiffe bis 24m Tiefgang,
   - *Honolulu*, *USA*, am *Pazifik*, Schiffe bis 12m Tiefgang,
   - *New Orleans*, *USA*, am *Atlantik*, Schiffe bis 12m Tiefgang,
   - *Dubai*, *Arabische Emirate*, am *Indischen Ozean*, Schiffe bis 23m Tiefgang.

4. Frachtschiffe haben keinen festen Fahrplan, sondern ihre Fahrten werden vom Disponenten nach Nachfrage geplant. Eine Fahrt geht von einem Hafen zu einem anderen, wobei der Abfahrtstag und der Ankunftstag gespeichert werden.

5. Ausserdem wird gespeichert, welche Frachtaufträge (Bezeichnung der Ladung, Masse in Tonnen, ggf. Anzahl Container, Name des Auftraggebers) bei welchen Fahrten transportiert werden.

   - Die *Tropicana* fährt am 8.2.2013 von *Honolulu* nach *New Orleans* (wo sie am 19.2. ankommt) und transportiert u.a. 2000 Tonnen Ananas in 400 Containern für die Firma *Tutti Frutti*.
   - Die *Tropicana* fährt am 21.2.2013 von *New Orleans* nach *Rotterdam* (wo sie am 29.2. ankommt) und transportiert u.a. 1000 Tonnen Ananas in 200 Containern für die Firma *Tutti Frutti* sowie 1000t Erdnüsse in 150 Containern für die Firma *NutShell*.
   - Die *Quick & Dirty* fuhr am 2.2.2013 in *Dubai* los und wird am 25.2. in *Rotterdam* ankommen, und transportiert 390000t Rohöl für die Firma *Peak Oil*.

- Die *Quick & Dirty* fährt am 28.2.2013 wieder von *Rotterdam* nach *Dubai* (wo sie am 15.3.2013 ankommt) und transportiert 2 Container mit 10t Ananas für *Tutti Frutti* sowie 20 Container mit Autos (Gesamtgewicht 40t) für *Daimler Benz*.

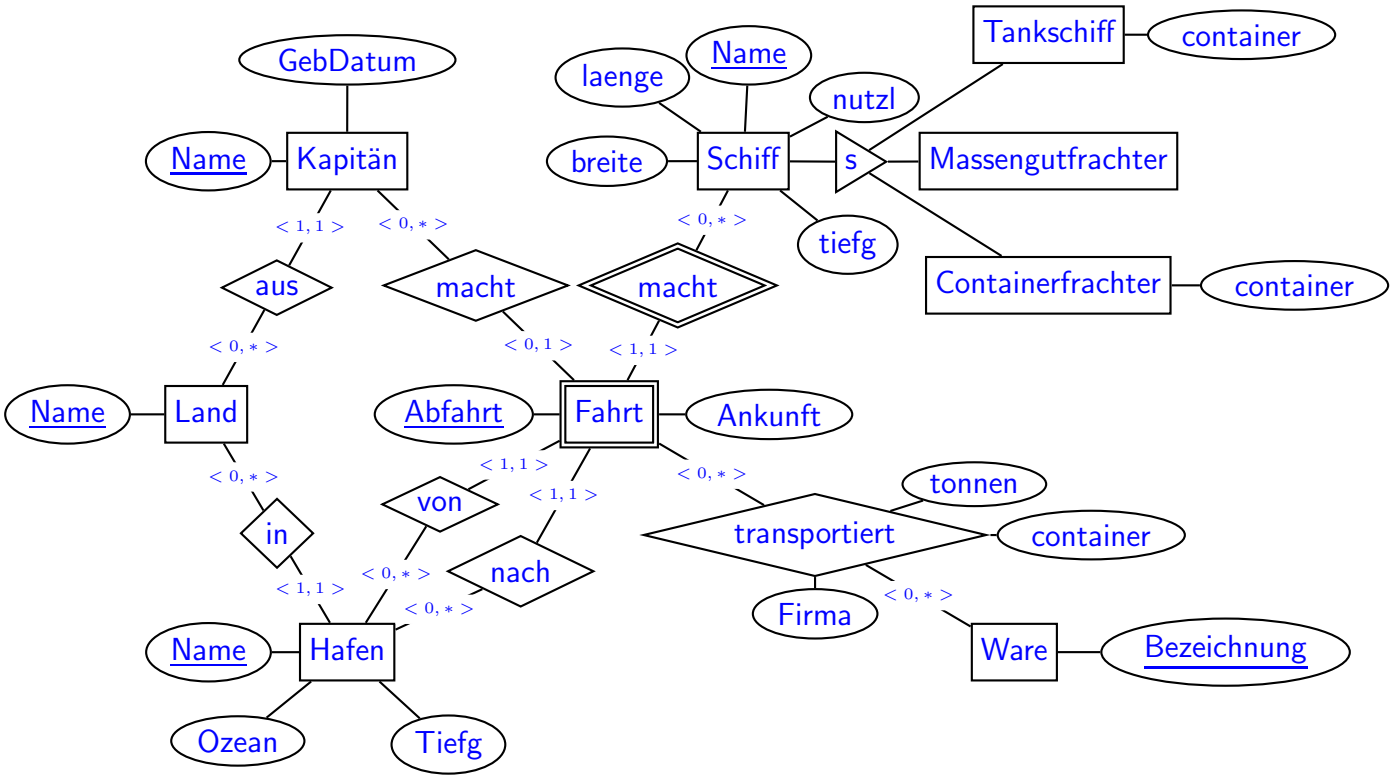
6. Die Diensteinteilung der Kapitäne zu den einzelnen Fahrten ist in der Datenbank abgelegt. Für jeden Kapitän ist ausserdem sein Geburtsdatum und sein Heimatland gespeichert:

   - *Hein Blöd* ist Kapitän auf der *Quick & Dirty* auf der Fahrt vom 2.2.-25.2. von *Dubai* nach *Rotterdam*. Er ist am 29.2.1960 geboren und ist Niederländer.
   - *Hans Dampf* ist Kapitän auf der *Quick & Dirty* auf der Fahrt vom 28.2.-15.3. von *Rotterdam* nach *Dubai*. Er ist am 1.3.1975 geboren und ist Deutscher.

7. Abgeschlossene Fahrten bleiben mit allen ihren Daten in der Datenbank erhalten.

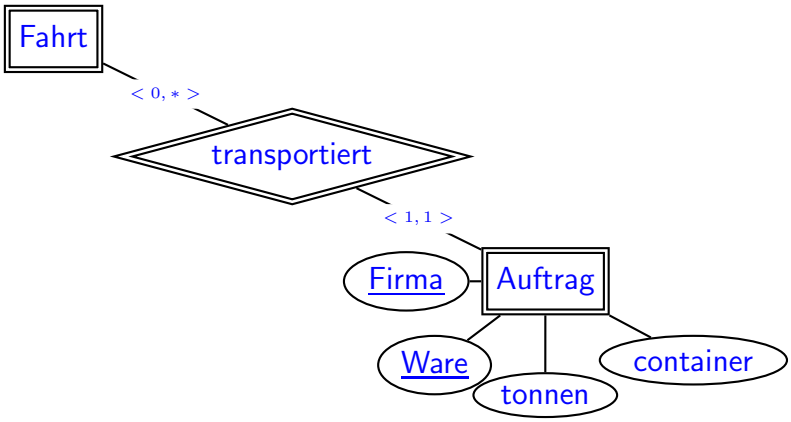
Aufgabe 1 (ER-Modell [15 Punkte])
Entwickeln Sie ein ER-Modell für das Szenario. Geben Sie darin die Schlüsselattribute sowie die Beziehungskardinalitäten an.

Lösung



Alternative Modellierungen und Kommentare:

- die Spezialisierung wurde nur in wenigen Lösungen verwendet (+1 P).
- Wenn man auf die Spezialisierung verzichtet, und stattdessen *Schiff* noch ein Attribut *Art* gibt, und das Attribut *container* bei *Schiff* dazunimmt, ist die Information, dass *Massengutfrachter* keine Containerplätze haben, nicht im Modell enthalten.
- Die Kardinalität von *Fahrt* in der *macht*-Beziehung zu *Kapitän* kann auch mit $< 1, 1 >$ angegeben werden, da jede Fahrt einen Kapitän benötigt. In der Praxis wird aber wohl oft eine Fahrt zur Planung angelegt, und der Kapitän erst später zugeordnet.
- Man könnte versuchen, *Fahrt* einfach nur als 4-stellige Beziehung zwischen (*Schiff, Hafen, Hafen, Kapitän*) zu machen, hätte aber dann keine Möglichkeit die Menge der transportierten Waren zu modellieren.
  Damit ist *Fahrt* praktisch eine aggregierte Beziehung.
- Als Schlüssel für den schwachen Entitätstyp *Fahrt* genügt (*Schiff.Name,Abfahrtsdatum*).
- Alternative Modellierung der rechten unteren Ecke des ER-Diagramms:



- Korrekturvorgabe: Relativ häufig (17 von 46 Teilnehmer) wurde die 1:n Beziehung *transportiert* zwischen *Fahrt* und *Ware/Auftrag* nicht erkannt, und zu einem *Auftrag* verschmolzen. Dies gab noch max. 10 P. (Dies hätte aus dem zweiten Beispiel in Eintrag (5) der Themenstellung erkannt werden müssen.)

Aufgabe 2 (Transformation in das Relationale Modell [17 Punkte])

a) **Lösen Sie diesen Aufgabenteil auf dem *letzten* Blatt und trennen dieses ab (und geben es am Ende mit ab!).** Dann haben Sie dieses Blatt separat zugreifbar um später damit die Aufgaben 2b, und 3 (SQL, Relationale Algebra+SQL) zu lösen.

Geben Sie an, welche Tabellen (mit Attributen, Schlüsseln etc.) Ihre Datenbank enthält (keine SQL `CREATE TABLE`-Statements, sondern einfach grafisch). (10 P)

Markieren Sie dabei auch Schlüssel (durch unterstreichen) und Fremdschlüssel (durch überstreichen).

Geben Sie die Tabellen mit jeweils mindestens zwei Beispieltupeln (z.B. denen, die sich aus dem Aufgabentext ergeben, und weiteren erfundenen) an.

Lösung

Schiff

Name	Art	Nutzlast	Tiefgang	Länge	Breite	Container
Corn Flake	Masseng.	70000	11	280	32	0
Tropicana	Container	60000	11	280	32	4800
Quick& Dirty	Tank	390000	22	370	64	50

Hafen

Name	Land	Ozean	Tiefgang
Rotterdam	NL	Atl.	24
Honolulu	USA	Pac.	12
New Orleans	USA	Atl.	12
Dubai	VAE	Ind.	23

Kapitän

Name	Land	GebDat
Hein Blöd	NL	29.2.1960
Hans Dampf	D	1.3.1975

Fahrt

Schiff	von	nach	Abfahrt	Ankunft	Kapitän
Tropicana	Honolulu	New Orleans	8.2.2013	19.2.2013	NULL
Tropicana	New Orleans	Rotterdam	21.2.2013	29.2.2013	NULL
Quick & Dirty	Dubai	Rotterdam	2.2.2013	25.2.2013	Hein Blöd
Quick & Dirty	Rotterdam	Dubai	28.2.2013	15.3.2013	Hans Dampf

transportiert oder **Auftrag**

Schiff	Abfahrt	Ware	Firma	Tonnen	Container
Tropicana	8.2.2013	Ananas	Tutti Frutti	2000	400
Tropicana	21.2.2013	Ananas	Tutti Frutti	1000	200
Tropicana	21.2.2013	Erdnüsse	NutShell	1000	150
Quick & Dirty	2.2.2013	Rohöl	Peak Oil	390000	0 (oder NULL)
Quick & Dirty	28.2.2013	Ananas	Tutti Frutti	10	2
Quick & Dirty	28.2.2013	Autos	Daimler Benz	40	20

Tabellen: 1+1+1+2+2 P, Tupel jeweils 1/2P pro Tabelle.

- Alternative: 3 Tabellen für die einzelnen Arten von Schiffen. Dann haben nur Tankschiffe und Containerfrachter ein Attribut *Container*. Problem: wenn ein weiterer Typ dazukommt bekommt man eine neue Tabelle, und muss alle existierenden Anfragen umschreiben. Auch Aufgaben 3b und 3c sind dann umständlicher (UNION).

- es ist sinnvoll, bei Massengutfrachtern für Container "0" einzutragen (sie haben keine Containerstellplätze) anstatt NULL (macht auch die Kontrolle in Aufgabe 3b einfacher).
- Die Tabelle für *Land* kann man sich (zumindest für die Aufgabenstellung) sparen, da sie nur die Namen enthält. (wenn man sie hat, hat man zwei refererentielle Integritätsbedingungen von *Kapitän* und von *Hafen*, die dann garantieren, dass nur Werte eingetragen können, die auch erlaubt sind (Schreibweisen etc.)).

b) Geben Sie das `CREATE TABLE`-Statement für diejenige Tabelle (bzw. die Tabellen), in der bei Ihnen die Daten über die transportierten Waren abgespeichert sind, so vollständig wie möglich an (7 P).

Lösung

```
CREATE TABLE transportiert                          Basis 2.5P
( Schiff   VARCHAR2(20),
  Abfahrt  DATE,
  Ware     VARCHAR2(20),
  Firma    VARCHAR2(20),
  Tonnen   NUMBER NOT NULL CHECK (Tonnen >= 0),     1/2 NOT NULL, 1/2 CHECK
  Container NUMBER CHECK (Container >= 0),          1/2 CHECK
CONSTRAINT transpkey PRIMARY KEY (Schiff, Abfahrt, Ware, Firma),
                                                    1P PKEY + 1P Firma dabei
CONSTRAINT transpfkey FOREIGN KEY
  (Schiff, Abfahrt) REFERENCES Fahrt(Schiff, Abfahrt)    1P FKEY
```

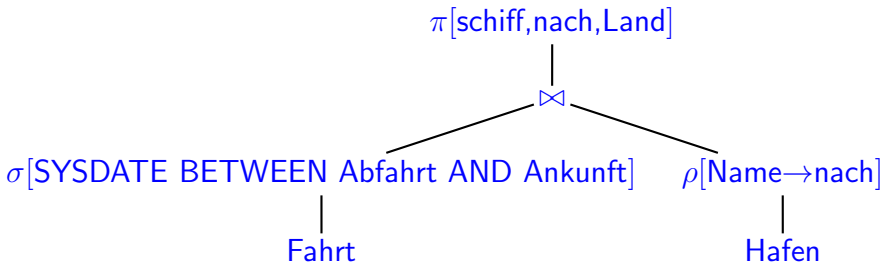
  - Firma ist Teil des Schlüssels, da die *Quick & Dirty* bei der Fahrt nach Dubai z.B. auch noch ein paar Container mit Porsches mitnehmen könnte.
  - es ist nicht notwendig, die Bedingung transportiert.Schiff→Schiff.Name explizit anzugeben, da sie von transportiert.(Schiff,Abfahrt)→Fahrt(Schiff,Abfahrt) und Fahrt.Schiff→Schiff.Name impliziert wird.

Aufgabe 3 (SQL und Relationale Algebra [45 Punkte])
Verwenden Sie für diese Aufgabe die von Ihnen entworfene relationale Datenbasis. Keine der Antworten soll Duplikate enthalten.

a) Geben Sie **eine SQL-Anfrage und einen Algebra-Ausdruck** an, die für alle Schiffe, die gerade irgendwo auf See unterwegs sind, die Namen des Schiffes sowie den Namen und das Land des Hafens, der als nächster angesteuert wird, ausgeben. (2+2 P)

Lösung

```
SELECT schiff, nach, Land
FROM Fahrt, Hafen
WHERE Fahrt.nach = Hafen.Name
  AND Abfahrt < SYSDATE
  AND Ankunft > SYSDATE
```



b) Geben Sie eine **SQL-Anfrage** an, die die Schlüsselattribute derjenigen Fahrten ausgibt, auf denen mehr Container zum Transport eingeplant sind, als das Schiff Containerstellplätze hat. (4 P)

Lösung

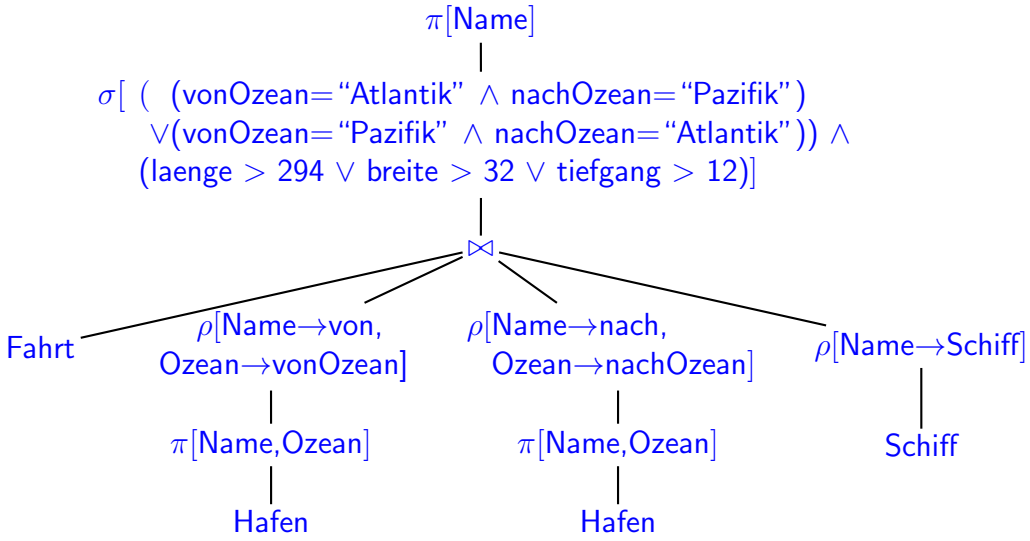
```
SELECT t.schiff, t.abfahrt
FROM transportiert t, schiff s
WHERE t.Schiff = s.Name
GROUP by t.schiff, t.abfahrt, s.Container
HAVING SUM(t.Container) > s.Container ### deshalb muss es ins GROUP BY

SELECT f.schiff, f.abfahrt
FROM fahrt f, schiff s
WHERE f.Schiff = s.Name
  AND s.Container < ( SELECT SUM(container)
                      FROM transportiert t
                      WHERE t.schiff = f.schiff
                        AND t.abfahrt = f.abfahrt)
```

c) Der Panama-Kanal erlaubt die Durchfahrt von Schiffen, die höchstens 294 m lang und höchstens 32 m breit sind, und einen Tiefgang von höchstens 12 m haben.
Geben Sie **eine SQL-Anfrage und einen Algebra-Ausdruck** an, die die Namen aller Schiffe ausgeben, für die eine Fahrt gespeichert ist, die im Atlantik oder im Pazifik startet, und im jeweils anderen Ozean endet, und die *nicht* durch den Panama-Kanal fahren dürfen. (4+4 P)

Lösung

```
SELECT distinct schiff.name
FROM fahrt, schiff, hafen von, hafen nach
WHERE schiff.name = fahrt.schiff
  AND fahrt.von=von.Name AND fahrt.nach=nach.Name
  AND (    (von.Ozean='Atlantik' AND nach.Ozean='Pazifik')
        OR (von.Ozean='Pazifik' AND nach.Ozean='Atlantik')
  AND (   schiff.laenge > 294 OR schiff.breite > 32
       OR schiff.tiefgang > 12)
```



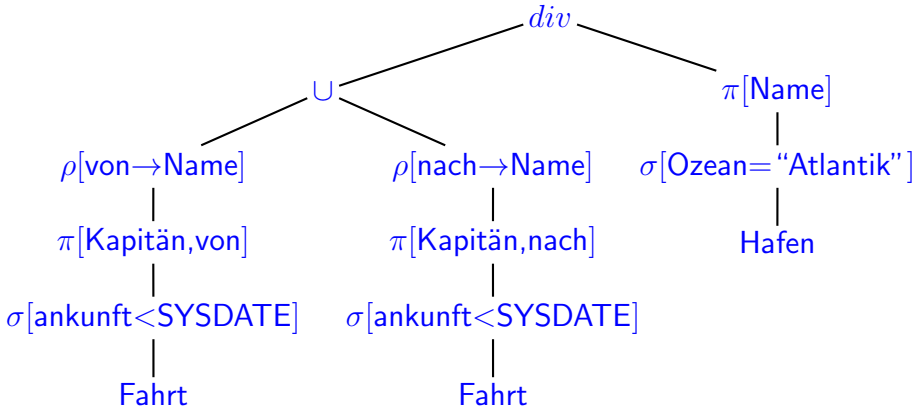
d) Geben Sie **eine SQL-Anfrage und einen Algebra-Ausdruck oder -Baum** an, der die Namen aller Kapitäne ausgibt, die auf ihren *bisher abgeschlossenen* Fahrten schon in jedem der am Atlantik gelegenen gespeicherten Häfen waren. (5+5 P)

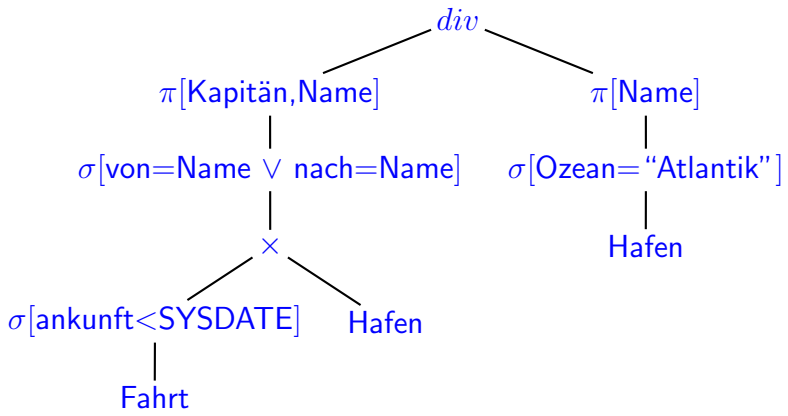
Lösung Es müssen sowohl Start als auch Ziel der Fahrten berücksichtigt werden, falls der Kapitän dort per Flugzeug eingetroffen ist und das Schiff übernommen hat (bzw. abgegeben hat und nach Hause geflogen ist).

```
SELECT name
FROM kapitaen k
WHERE NOT EXISTS
  (SELECT *
   FROM Hafen h
   WHERE Ozean = 'Atlantik'
```

```
      AND NOT EXISTS
      (SELECT *
        FROM Fahrt
        WHERE ankunft < SYSDATE    -- Fahrt ist abgeschlossen
        AND kapitaen = k.name
        AND (von = h.name OR nach = h.name)))   -- startete oder endete in dem Hafen

SELECT name
FROM kapitaen k
WHERE NOT EXISTS
  (SELECT *
   FROM Hafen
   WHERE Ozean = 'Atlantik'
     AND Name NOT IN
     ((SELECT von                  -- Fahrt startete in dem Hafen
       FROM Fahrt
       WHERE ankunft < SYSDATE    -- Fahrt ist abgeschlossen
       AND kapitaen = k.name)
      UNION
      (SELECT nach                 -- Fahrt endete in dem Hafen
       FROM Fahrt
       WHERE ankunft < SYSDATE
       AND kapitaen = k.name)))
```



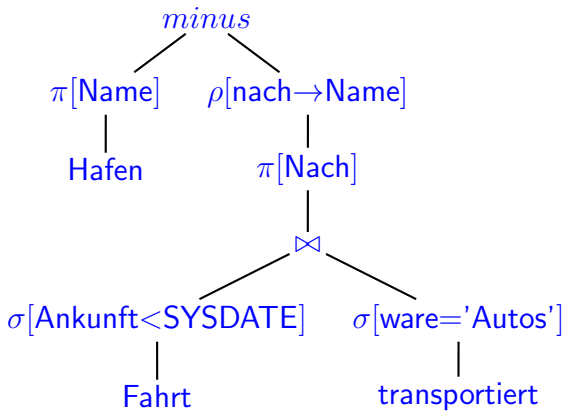


e) Geben Sie **eine SQL-Anfrage und einen Algebra-Ausdruck oder -Baum** an, der die Namen aller Häfen ausgibt, die noch nie von einem Schiff mit Autos an Bord angefahren wurden. (4+4 P)

Lösung

```
SELECT name
FROM Hafen
WHERE NOT EXISTS
(SELECT *
 FROM Fahrt, transportiert t
 WHERE Fahrt.Schiff = t.Schiff
   AND Fahrt.Abfahrt = t.Abfahrt
   AND Ankunft < SYSDATE
   AND Ware = 'Autos'
   AND Hafen.Name = Fahrt.nach)
```

```
(SELECT name
 FROM Hafen )
MINUS
(SELECT nach AS Hafen
 FROM Fahrt, transportiert t
 WHERE Fahrt.Schiff = t.Schiff
   AND Fahrt.Abfahrt = t.Abfahrt
   AND Ankunft < SYSDATE
   AND Ware = 'Autos')
```



f) Geben Sie eine SQL-Anfrage an, die für jeden "Stop" eines Schiffes in einem Hafen zwischen zwei Fahrten ergibt, von welchen Waren zumindest ein Teil *nicht* abgeladen werden dürfen, sondern weitertransportiert werden (vgl. der Transport eines Teils der Ananas-Ladung für *Tutti Frutti* durch New Orleans.) (5 P)

Lösung

```
SELECT t1.ware, t1.firma
FROM transportiert t1, transportiert t2
WHERE t1.schiff = t2.schiff
  AND t2.abfahrt = ( SELECT MIN(abfahrt)
                     FROM fahrt f3
                     WHERE f3.schiff = t2.schiff
                       AND f3.abfahrt > t1.ankunft )
  AND t1.ware = t2.ware AND t1.firma = t2. firma     -- weitertransportiere

SELECT t1.ware, t1.firma
FROM transportiert t1, transportiert t2
WHERE t1.schiff = t2.schiff
```

```
    AND t2.abfahrt > t1.ankunft
    AND t1.ware = t2.ware AND t1.firma = t2. firma     -- weitertransportieren
    AND NOT EXISTS ( SELECT *
                     FROM fahrt f3
                     WHERE f3.schiff = t2.schiff
                     AND f3.abfahrt > t1.ankunft
                     AND f3.abfahrt < t2.abfahrt )

-- vgl. der Transport eines Teils der Ananasse durch New Orleans.
-- ebenfalls zu berucksichtigen: wenn von Produkt A bereits x Einheiten
-- an Bord sind, und y>x weitertransportiert werden, darf auch nichts
-- abgeladen werden.
```

g) **Etwas Theorie:** Gegeben sind zwei beliebige Relationen $R(A, B, C)$ und $S(B, C)$.
Sei $\psi := (\pi[A, B](R) \div \pi[B](S)) \cap (\pi[A, C](R) \div \pi[C](S)$.
Welche der folgenden Aussagen ist zutreffend (MIT BEGRÜNDUNG): (6 P)

- ☐ $R \div S = \psi$,
- ☐ $R \div S \subseteq \psi$,
- ☐ $R \div S \supseteq \psi$.

Lösung Die zweite Aussage ist richtig, die beiden anderen sind falsch.

$\pi[A, B](R) \div \pi[B](S)$ ergibt alle A-Werte, die mit allen B-Werten aus S in R vorkommen.

$\pi[A, C](R) \div \pi[C](S)$ ergibt alle A-Werte, die mit allen C-Werten aus S in R vorkommen.

Die Schnittmenge ergibt somit alle alle A-Werte, die beides erfüllen. Dies ist sicher eine Obermenge derjenigen A-Werte, die mit jeder der in S enthaltenen (A, B)-*Kombinationen* in R vorkommen. Im allgemeinen Fall ist es aber eine *echte* Obermenge, wie folgende Beispiele zeigen:

R		
A	B	C
a	1	1
a	2	2

S	
B	C
1	2
2	1

$R \div S = \emptyset, \quad \psi = A : \{a\}$.

R		
A	B	C
a	1	2
a	2	1

S	
B	C
1	1

$R \div S = \emptyset, \quad \psi = A : \{a\}$.

Aufgabe 4 (Verschiedenes [13 Punkte])

a) Wie muss das ER-Modell und das relationale Schema geändert werden, wenn (i) der vereinbarte Preis für einen Transport und (ii) ob dieser bereits bezahlt wurde, gespeichert werden soll? (4 P)

Lösung

- ER-Modell: Attribute *preis* und *bezahlt* zu der *transportiert*-Beziehung.
- Relationales Modell: diese beiden Spalten zu der Tabelle *transportiert* hinzunehmen.

b) Gemäß Aufgabenstellung führt der Disponent die eigentliche Planung durch und trägt die Daten dann in die Datenbank ein.
Wie kann man datenbankseitig in der Praxis garantieren, dass er dabei nicht mehr Container für eine Fahrt eintragen kann, als das Schiff Stellplätze hat? (Hier ist kein vollständiger SQL-Code gefragt, sondern eine Beschreibung des Konzepts und des Ablaufs) (5 P)

Lösung Einen Trigger, der für jedes INSERT oder UPDATE auf der Relation transportiert überprüft, ob für die neu eingetragenen Container noch Kapazität frei ist. Z.B. ein Trigger auf `BEFORE INSERT OR UPDATE ON transportiert` die Anfrage in Aufgabe 3b (für das jeweilige Schiff) auswertet und ggf. eine Fehlermeldung ausgibt.

Hinweis: ein einfaches CONSTRAINT geht nicht, da dieses nur auf ein einziges Tupel der Tabelle auf der es definiert ist, zugreifen darf,

c) Annahme: die Reederei beschäftigt nur einen Disponenten, der schreibend auf die Datenbank zugreift. Alle anderen Benutzer greifen nur lesend darauf zu (aber beliebig viele gleichzeitig, evtl. auch gleichzeitig mit dem Disponenten).
Geben Sie 2 Gründe an, dass ein Datenbanksystem mit Transaktionsunterstützung dennoch für die Anwendung sehr sinnvoll ist. (4 P)

Lösung

- Dirty Reads (d.h., Werte. die von einer noch nicht beendeten Transaktion geschrieben wurden, und später via ROLLBACK wieder rückgängig gemacht werden) können auch bei nur einem Schreiber und mindestens einem Leser auftreten.
- Phantom: Ein Leser bildet eine Summe über irgendwas, während der Disponent am Einfügen/Updaten bzgl. der summierte Menge ist.
- spitzfindig: der Disponent könnte über zwei verschiende Tools (oder über eines zweimal) schreibend zugreifen und sich somit selber stören.
- Physikalische Sicherheit: wenn nach oder sogar während einer Schreibtransaktion ein Problem (temporärer Stromausfall, Festplattencrash) auftritt, kann der Zustand korrekt wiederhergestellt werden.
- Konsistenzbedingungen werden nach zum Ende der Transaktion überprüft, und ggf die *gesamte* Transaktion zurückgenommen (z.B. wenn im Beispiel weitere 30 Container mit Ananas von Honolulu über New Orleans und Rotterdam nach Dubai duchgebucht würden, was auf dem letzten Schritt die Containerkapazität der *Quick & Dirty* übersteigen würde, würden alle 3 Teilschritte nicht akzeptiert).

Hinweis: Konsistenz nach Abschluss von Updates (d.h., Erfülltsein aller Constraints) und Persistenz ist auch von nicht-transaktionsunterstützenden DBSen gewährleistet – dort wird einfach jedes atomare Update als Transaktion angesehen und sofort auf Konsistenz geprüft, allen sichtbar gemacht, und persistiert.