	Lehrveranstaltung	Grundlagen von Datenbanken			WS 2012/13
	Aufgabenzettel	4 (Lösungsvorschläge)			
	Gesamtpunktzahl	40			
	Ausgabe	Do. 22.11.12	Abgabe	Do. 06.12.12	

Aufgabe 1: Schemadefinition

(18 Punkte)

Wir verwenden das gleiche Datenbankschema wie in der dritten Aufgabe von Aufgabenblatt 3:

Personal	<u>PID</u>	Vorname	Nachname	Geburt	Wohnort	<u>Abteilung</u>
	4	Peter	Müller	1962-07-25	Hamburg	2
	8	Bianca	Lohse	1982-01-13	Kiel	4
	11	Murat	Sahir	1990-03-16	Hamburg	2
	21	Frank	Siebenstein	1975-12-02	Norderstedt	1
	22	Bernd	Schmidt	1973-11-26	Norderstedt	1
	24	Ulrike	Müller	1963-10-07	Hamburg	2
	31	Jochen	Fuhrmann	1958-05-09	Stade	2

Abteilung → Abteilungen.AID

Projekte	<u>PrID</u>	Name	<u>Leiter</u>	Budget
	15	Prozessoptimierung	22	10.000
	36	B.L.I.C.K.F.A.N.G	8	7.500

Leiter → Personal.PID

ProjektArbeiter	<u>PrID</u>	<u>PID</u>
	36	8
	15	21
	36	11
	15	22
	36	31

PrID → Projekte.PrID, PID → Personal.PID


Abteilungen	<u>AID</u>	Name
	1	Controlling
	4	Marketing
	2	Einkauf

Um die Konsistenz der Daten sicherzustellen, sollen folgende Integritätsbedingungen gelten:

IB1: Das Projektbudget muss zwischen 0 und 200.000 liegen.

IB2: Abteilungsamen sind eindeutig.

IB3: Alle Felder bis auf *Personal.Wohnort* und *Personal.Geburt* sind Pflichtfelder.

	Lehrveranstaltung	Grundlagen von Datenbanken			WS 2012/13
	Aufgabenzettel	4 (Lösungsvorschläge)			
	Gesamtpunktzahl	40			
	Ausgabe	Do. 22.11.12	Abgabe	Do. 06.12.12	

- a) Definieren Sie das angegebene Schema mithilfe von Befehlen der SQL DDL (Data Definition Language). Zur Prüfung Ihrer Lösung führen Sie die DDL-Befehle bitte in MySQL aus.
Hinweis: Legen Sie Fremdschlüssel bitte als benannte Constraints an. Legen Sie die Prüfung des Budgets als eine Check-Klausel an, auch wenn MySQL diese (ohne Fehler) ignoriert. (8 Punkte)

Lösungsvorschlag:

```


CREATE TABLE Abteilungen(
  AID          int          PRIMARY KEY,
  Name         varchar(50) UNIQUE NOT NULL
);

CREATE TABLE Personal(
  PID          int          PRIMARY KEY,
  Vorname      varchar(50) NOT NULL,
  Nachname     varchar(50) NOT NULL,
  Geburt       date,
  Wohnort      varchar(50),
  Abteilung    int          NOT NULL,
  CONSTRAINT fk_pr_abteilung FOREIGN KEY (Abteilung) REFERENCES Abteilungen (AID)
);

CREATE TABLE Projekte(
  PrID         int          PRIMARY KEY,
  Name         varchar(50) NOT NULL,
  Leiter       int          NOT NULL,
  Budget       int          NOT NULL CHECK(Budget > 0 AND Budget <200000),
  CONSTRAINT fk_pr_leiter FOREIGN KEY (Leiter) REFERENCES Personal (PID)
);

CREATE TABLE ProjektArbeiter(
  PrID         int,
  PID          int,
  CONSTRAINT pk_pa PRIMARY KEY (PrID, PID),
  CONSTRAINT fk_pa_proj FOREIGN KEY (PrID) REFERENCES Projekte (PrID),
  CONSTRAINT fk_pa_pers FOREIGN KEY (PID) REFERENCES Personal (PID)
);

```

	Lehrveranstaltung	Grundlagen von Datenbanken			WS 2012/13
	Aufgabenzettel	4 (Lösungsvorschläge)			
	Gesamtpunktzahl	40			
	Ausgabe	Do. 22.11.12	Abgabe	Do. 06.12.12	

- b) Erklären Sie knapp, was es für Transaktionen bedeutet, dass in MySQL die referentielle Integrität von Fremdschlüsseln nicht verzögert am Ende der Transaktion (*deferred*) geprüft werden kann, sondern stets direkt.

Erläutern Sie, was passieren würde, wenn *Abteilungen* das Feld *Leiter* erhalten würde, welches ein Fremdschlüssel auf *Personal.PID* ist. Was müsste man bei der Definition des Schemas in SQL DDL beachten? (4 Punkte)

Lösungsvorschlag:

Laut Dokumentation unterstützt MySQL das deferred Checking von Fremdschlüsseln nicht:


Deviation from SQL standards: Like MySQL in general, in an SQL statement that inserts, deletes, or updates many rows, InnoDB checks UNIQUE and FOREIGN KEY constraints row-by-row. When performing foreign key checks, InnoDB sets shared row-level locks on child or parent records it has to look at. InnoDB checks foreign key constraints immediately; the check is not deferred to transaction commit. According to the SQL standard, the default behavior should be deferred checking. That is, constraints are only checked after the entire SQL statement has been processed. Until InnoDB implements deferred constraint checking, some things will be impossible, such as deleting a record that refers to itself using a foreign key.

Dies hat Auswirkungen auf Transaktionen, z.B:

- Einträge, die sich gegenseitig oder selbst referenzieren, können nicht gelöscht werden.
- Beim Löschen von Einträgen muss auf die richtige Reihenfolge geachtet werden, z.B. müssen wegen *Personal.Abteilung* → *Abteilungen.AID* beim Löschen einer Abteilung zuerst alle referenzierenden Personen gelöscht oder geändert werden.
- Beim Einfügen muss ebenfalls auf die richtige Reihenfolge geachtet werden, damit keine nichtexistierenden Tupel referenziert werden.
- Das Löschen ganzer Tabellen erfordert ebenfalls die Einhaltung der Reihenfolge, die durch die Fremdschlüssel vorgegeben ist (z.B. *Projekte* nach *Personal* löschen).

Wenn in *Abteilungen* das Feld *Leiter* eingeführt wird, entsteht eine wechselseitige Abhängigkeit. Da Fremdschlüssel nicht deferred geprüft werden, können neue Einträge dann nur angelegt werden, wenn einer der Fremdschlüssel NULLable ist, d.h. auch NULL-Werte annehmen darf.

Bei der Definition des Schemas muss darauf geachtet werden, zunächst nur einen Foreign Key Constraint anzulegen. Sobald beide Tabellen existieren, kann über ALTER TABLE der zweite Constraint hinzugefügt werden.

	Lehrveranstaltung	Grundlagen von Datenbanken			WS 2012/13
	Aufgabenzettel	4 (Lösungsvorschläge)			
	Gesamtpunktzahl	40			
	Ausgabe	Do. 22.11.12	Abgabe	Do. 06.12.12	

- c) Befüllen Sie die Datenbank mit den in der Tabelle angegebenen Datensätzen. Schreiben Sie die SQL-Befehle auf. (4 Punkte)

Lösungsvorschlag:

```

INSERT INTO Abteilungen (AID, Name) VALUES
(1, "Controlling"), (4, "Marketing"), (2, "Einkauf");

INSERT INTO Personal(PID, Vorname, Nachname, Geburt, Wohnort, Abteilung) VALUES
(4, "Peter", "Müller", "1962-07-25", "Hamburg", 2),
(8, "Bianca", "Lohse", "1982-01-13", "Kiel", 4),
(11, "Murat", "Sahir", "1990-03-16", "Hamburg", 2),
(21, "Frank", "Siebenstein", "1975-12-02", "Norderstedt", 1),
(22, "Bernd", "Schmidt", "1973-11-26", "Norderstedt", 1),
(24, "Ulrike", "Müller", "1963-10-07", "Hamburg", 2),
(31, "Jochen", "Fuhrmann", "1958-05-09", "Stade", 2);

INSERT INTO Projekte (PrID, Name, Leiter, Budget) VALUES
(15, "Prozessoptimierung", 22, 10000),
(36, "B.L.I.C.K.F.A.N.G", 8, 7500);

INSERT INTO ProjektArbeiter(PrID, PID) VALUES
(36, 8), (15, 21), (36, 11), (15, 22), (36, 31);

COMMIT;

```

- d) Geben Sie die SQL Befehle an, um: (2 Punkte)

- alle Tabellen zu löschen (Reihenfolge beachten!)
- alle Personen mit dem Namen "Peter Müller" zu löschen


Lösungsvorschlag:

Alle Tabellen löschen:

```

DROP TABLE ProjektArbeiter;
DROP TABLE Projekte;
DROP TABLE Personal;
DROP TABLE Abteilungen;

```

	Lehrveranstaltung	Grundlagen von Datenbanken			WS 2012/13
	Aufgabenzettel	4 (Lösungsvorschläge)			
	Gesamtpunktzahl	40			
	Ausgabe	Do. 22.11.12	Abgabe	Do. 06.12.12	

Alle Personen mit dem Namen "Peter Müller" löschen:

```
DELETE FROM Personal
WHERE Vorname = "Peter"
AND Nachname = "Müller";
```

Aufgabe 2: SQL

(12 Punkte)

Die folgenden Teilaufgaben basieren auf dem in Aufgabe 1 dargestellten relationalen Datenbankschema. Übersetzen Sie die folgenden umgangssprachlich formulierten Anfragen in SQL-Anfragen und führen Sie die Anfragen mit MySQL aus.

- a) Die Namen aller Projekte die ein Budget von mehr als 5000 Geldeinheiten haben.

Lösungsvorschlag:

```
SELECT DISTINCT PR.Name
FROM Projekte PR
WHERE PR.Budget > 5000
```


Ergebnismenge:

Name
Prozessoptimierung
B.L.I.C.K.F.A.N.G

- b) Die Nachnamen aller Mitarbeiter (alphabetisch aufsteigend), die im Projekt mit dem Namen „B.L.I.C.K.F.A.N.G“ arbeiten.

Lösungsvorschlag:

```
SELECT P.NachName
FROM Personal P, ProjektArbeiter PA, Projekte PR
WHERE PR.Name = "B.L.I.C.K.F.A.N.G"
AND PA.PID = P.PID
AND PA.PrID = PR.PrID
ORDER BY P.NachName
```

	Lehrveranstaltung	Grundlagen von Datenbanken			WS 2012/13
	Aufgabenzettel	4 (Lösungsvorschläge)			
	Gesamtpunktzahl	40			
	Ausgabe	Do. 22.11.12	Abgabe	Do. 06.12.12	

Ergebnismenge:

Nachname

Fuhrmann

Lohse

Sahir

c) Zu jedem Projekt der Name des Projekts und das Geburtsdatum des ältesten Projektmitarbeiters.

Lösungsvorschlag:

```
SELECT PR.Name, MIN(P.Geburt) AS ältester
FROM Personal P, ProjektArbeiter PA, Projekte PR
WHERE PA.PID = P.PID
AND PA.PrID = PR.PrID
GROUP BY PA.PrID, PR.Name
```

Ergebnismenge:

<u>Name</u>	ältester
Prozessoptimierung	1973-11-26
B.L.I.C.K.F.A.N.G	1958-05-09


d) Nachnamen aller Mitarbeiter in der Abteilung „Controlling“ alphabetisch absteigend sortiert.

Lösungsvorschlag:

```
SELECT Nachname
FROM Personal P, Abteilungen A
WHERE P.Abteilung = A.AID
AND A.Name = "Controlling"
ORDER BY Nachname DESC
```

Ergebnismenge:

<u>Nachname</u>
Siebenstein
Schmidt

	Lehrveranstaltung	Grundlagen von Datenbanken			WS 2012/13
	Aufgabenzettel	4 (Lösungsvorschläge)			
	Gesamtpunktzahl	40			
	Ausgabe	Do. 22.11.12	Abgabe	Do. 06.12.12	

e) Für jede Abteilung den Abteilungsnamen und die Anzahl der Mitarbeiter.

Lösungsvorschlag:

```
SELECT A.Name, COUNT(*) AS Mitarbeiter
FROM Personal M, Abteilungen A
WHERE M.Abteilung = A.AID
GROUP BY A.AID, A.Name
```

Ergebnismenge:

Name	Mitarbeiter
Controlling	2
Einkauf	4
Marketing	1


f) Alle Informationen zu Mitarbeitern, die an keinem Projekt mitarbeiten.

Lösungsvorschlag:

```
SELECT *
FROM Personal P
WHERE NOT EXISTS (SELECT *
                  FROM ProjektArbeiter PA
                  WHERE PA.PID = P.PID)
```

Ergebnismenge:

PID	Vorname	Nachname	Geburt	Wohnort	Abteilung
4	Peter	Müller	1962-07-25	Hamburg	2
24	Ulrike	Müller	1963-10-07	Hamburg	2

	Lehrveranstaltung	Grundlagen von Datenbanken			WS 2012/13
	Aufgabenzettel	4 (Lösungsvorschläge)			
	Gesamtpunktzahl	40			
	Ausgabe	Do. 22.11.12	Abgabe	Do. 06.12.12	

- g) Den Nachnamen, den Namen der Abteilung, in der er arbeitet, sowie die Summe des von ihm insgesamt verantworteten Projektbudgets eines jeden Projektleiters.

Lösungsvorschlag:

```
SELECT A.Name, P.Nachname, B.GesamtBudget
FROM Personal P,
     Abteilungen A,
     (SELECT Leiter AS PID, SUM(Budget) AS GesamtBudget
      FROM Projekte Proj
      GROUP BY Leiter) B
WHERE P.Abteilung = A.AID
AND P.PID = B.PID
```

Ergebnismenge:

Name	Nachname	GesamtBudget
Controlling	Schmidt	10,000
Marketing	Lohse	7,500


- h) Alle Projektnamen, in denen Personen arbeiten, die zwischen dem 01.01.1970 und dem 01.01.1980 geboren wurden und deren Nachname mit einem „S“ oder einem „L“ beginnt.

Lösungsvorschlag:

```
SELECT DISTINCT PR.Name
FROM Personal P, ProjektArbeiter PA, Projekte PR
WHERE P.PID = PA.PID
AND PA.PrID = PR.PrID
AND Geburt BETWEEN "1970-01-01" AND "1980-01-01"
AND (Nachname LIKE "S%" OR Nachname LIKE "L%")
```

Ergebnismenge:

Name
Prozessoptimierung

	Lehrveranstaltung	Grundlagen von Datenbanken			WS 2012/13
	Aufgabenzettel	4 (Lösungsvorschläge)			
	Gesamtpunktzahl	40			
	Ausgabe	Do. 22.11.12	Abgabe	Do. 06.12.12	

- i) Für jede Abteilung die Wohnorte, aus denen ihre Mitarbeiter kommen, sowie die Anzahl der Mitarbeiter pro Wohnort, wenn mindestens 2 Personen in der Abteilung arbeiten.

Lösungsvorschlag:

```
SELECT P.Abteilung, P.Wohnort, COUNT(*) AS Anzahl
FROM Personal P
GROUP BY P.Abteilung, P.Wohnort
HAVING (SELECT COUNT(*) FROM Personal P2 WHERE P2.Abteilung=P.Abteilung)>=2
```

Ergebnismenge:

Abteilung	Wohnort	Anzahl
1	Norderstedt	2
2	Hamburg	3
2	Stade	1


- j) Der Name von Projekten und ihrem Projektleiter, in denen der Projektleiter jünger ist als alle darin mitarbeitenden Mitarbeiter.

Lösungsvorschlag:

```
SELECT P.Name, L.Nachname
FROM Projekte P, Personal L
WHERE P.Leiter = L.PID
AND L.Geburt > ALL (SELECT Geburt
FROM Personal M, ProjektArbeiter PR
WHERE M.PID = PR.PID
AND PR.PrID = P.PrID
AND M.PID != L.PID)
```

Die Ergebnismenge ist leer:

Name	Nachname

	Lehrveranstaltung	Grundlagen von Datenbanken			WS 2012/13
	Aufgabenzettel	4 (Lösungsvorschläge)			
	Gesamtpunktzahl	40			
	Ausgabe	Do. 22.11.12	Abgabe	Do. 06.12.12	

Aufgabe 3: Anfrageoptimierung

(10 Punkte)

Übersetzen Sie folgende SQL-Anfrage entsprechend dem in der Vorlesung vorgestellten Erklärungsmodell in einen Operatorbaum (wählen Sie einen beliebigen der verschiedenen möglichen Operatorbäume). Führen Sie anschließend eine algebraische Optimierung entsprechend den in der Vorlesung eingeführten Regeln durch. Bewerten Sie beide Operatorbäume mit den Kardinalitäten der Zwischenergebnisse.


```

SELECT DISTINCT P.Vorname, P.Nachname, A.Name
FROM   Personal P,
       Abteilungen A,
       ProjektArbeiter PA,
       Projekte PR
WHERE  P.Abteilung = A.AID
AND    PA.PrID = PR.PrID
AND    PA.PID = P.PID
AND    PR.Budget BETWEEN 5000 AND 15000;
AND    P.Wohnort IN ("Hamburg", "Kiel", "Berlin")

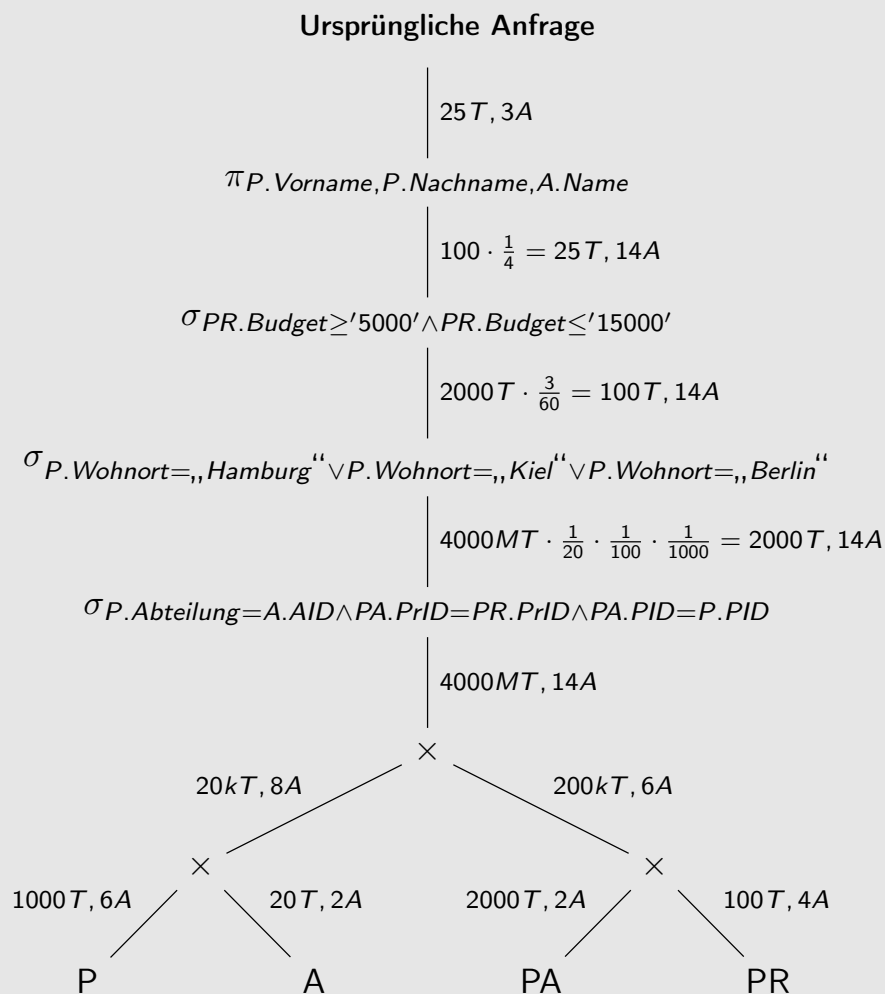
```

Für die zugehörige Datenbank werden folgende Kardinalitäten angenommen:


$\text{Card}(\text{Abteilung}) = 20$, $\text{Card}(\text{Personal}) = 1000$, $\text{Card}(\text{Projekt}) = 100$, $\text{Card}(\text{ProjektArbeiter}) = 2000$. Zudem gibt es insgesamt 60 verschiedene Wohnorte. Hinweis: Beachten Sie, dass über das minimale und maximale Budget eines Projektes nichts bekannt ist. Daher muss in diesem Fall die in der Vorlesung behandelte Abschätzung des Selektivitätsfaktors verwendet werden.

	Lehrveranstaltung	Grundlagen von Datenbanken			WS 2012/13
	Aufgabenzettel	4 (Lösungsvorschläge)			
	Gesamtpunktzahl	40			
	Ausgabe	Do. 22.11.12	Abgabe	Do. 06.12.12	

Lösungsvorschlag:



Erläuterung: T=Tupel, A=Attribute

	Lehrveranstaltung	Grundlagen von Datenbanken		WS 2012/13
	Aufgabenzettel	4 (Lösungsvorschläge)		
	Gesamtpunktzahl	40		
	Ausgabe	Do. 22.11.12	Abgabe	Do. 06.12.12

Optimierte Anfrage

