[4131658] Jan Germann
[4054962] Christian Ratz
Übungsgruppe 1

Relationale Datenbanken
23. Februar 2013

Aufgabenblatt 9

9.1 Aufgabe

Based on the given conceptual schema, please provide SQL statements to create the according tables described in the schema.

```
1  -- Employee gehort zu einer bestimmten Abteilung.
2  CREATE TABLE Employee (
3    empNr         int(11) NOT NULL AUTO_INCREMENT,
4    name          varchar(255),
5    department    int(10) NOT NULL,
6    departmentnr  int(10) NOT NULL,
7    PRIMARY KEY (empNr));
8
9  -- Telefonnummer kann mehrwertig sein.
10 -- Also Tabelle fur Telefonnummer erstellen
11 CREATE TABLE PhoneNumbers (
12   employeeempnr int(11) NOT NULL,
13   phoneNr       varchar(255));
14   --PRIMARY KEY ( employeeempnr, phoneNr )
15
16 -- Die Abteilung hat einen Leiter
17 CREATE TABLE Department (
18   house    --CHAR(1)-- NOT NULL,
19   nr       int(10) NOT NULL,
20   manager  int(11) NOT NULL --REFERENCES employee,
21   budget   --float(10),
22   PRIMARY KEY (house, nr));
23
24 -- Das Projekt wird uber eine Relationentabelle mit den
25 -- Mitarbeitern verknuepft.
26 CREATE TABLE Project (
27   name         varchar(255) NOT NULL,
28   description  varchar(255),
29   PRIMARY KEY (name));
30
31 -- Relationentabelle um Mitarbeiter mit einem Projekt zu verknuepfen
32 CREATE TABLE Employee_Project (
33   employeeempnr int(11) NOT NULL --REFERENCES employee ON DELETE CASCADE,
34   projectname   varchar(255) NOT NULL --REFERENCES project ON DELETE CASCADE,
35   PRIMARY KEY (employeeempnr, projectname));
36
37 -- Die offenen Tickets werden hier gespeichert.
38 CREATE TABLE Ticket (
```

```
39    nr           int(10) NOT NULL AUTO_INCREMENT,  ---abhaengig vom Projekt
40    --project varchar(255) NOT NULL REFERENCES project on DELETE CASCADE,
41    title        varchar(255),
42    description  varchar(255),
43    importance   --varchar(255) CHECK(importance IN('minor', 'medium', 'major')),
44    PRIMARY KEY (nr,--project));
45
46  -- Relationentabelle um Tickets einem Projekt zuzuordnen
47  CREATE TABLE Project_Ticket (
48    projectname varchar(255) NOT NULL,
49    ticketnr    int(10) NOT NULL,
50    PRIMARY KEY (projectname,
51    ticketnr));
52
53  -- Relationentabelle um die Arbeit eines Mitarbeiters
54  -- an einem Ticket zu dokumentieren.
55  CREATE TABLE Employee_Ticket (
56    employeeempnr int(11) NOT NULL,
57    ticketnr      int(10) NOT NULL,
58    start         timestamp NOT NULL,
59    end           timestamp NULL,
60    --project varchar(200) NOT NULL REFERENCES Project
61    PRIMARY KEY (employeeempnr, ticketnr,--project));
62
63  -- Fremdschluesselabhaengigkeiten erstellen
64  ALTER TABLE PhoneNumbers
65    ADD INDEX FKPhoneNumbe658372 (employeeempnr),
66    ADD CONSTRAINT FKPhoneNumbe658372
67    FOREIGN KEY (employeeempnr) REFERENCES Employee (empNr);
68  ALTER TABLE Employee_Project
69    ADD INDEX FKEmployee_P990046 (employeeempnr),
70    ADD CONSTRAINT FKEmployee_P990046
71    FOREIGN KEY (employeeempnr) REFERENCES Employee (empNr);
72  ALTER TABLE Employee_Project
73    ADD INDEX FKEmployee_P943600 (projectname),
74    ADD CONSTRAINT FKEmployee_P943600
75    FOREIGN KEY (projectname) REFERENCES Project (name);
76  ALTER TABLE Department
77    ADD INDEX FKDepartment689529 (manager),
78    ADD CONSTRAINT FKDepartment689529
79    FOREIGN KEY (manager) REFERENCES Employee (empNr);
80  ALTER TABLE Employee
81    ADD INDEX FKEmployee366382 (department, departmentnr),
82    ADD CONSTRAINT FKEmployee366382
83    FOREIGN KEY (department, departmentnr) REFERENCES Department (house, nr);
84  ALTER TABLE Project_Ticket
85    ADD INDEX FKProject_Ti16548 (projectname),
86    ADD CONSTRAINT FKProject_Ti16548
87  FOREIGN KEY (projectname) REFERENCES Project (name);
88  ALTER TABLE Project_Ticket
89    ADD INDEX FKProject_Ti360860 (ticketnr),
```

```
90    ADD CONSTRAINT FKProject_Ti360860
91    FOREIGN KEY (ticketnr) REFERENCES Ticket (nr);
92  ALTER TABLE Employee_Ticket
93    ADD INDEX FKEmployee_T249102 (employeeempnr),
94    ADD CONSTRAINT FKEmployee_T249102
95    FOREIGN KEY (employeeempnr) REFERENCES Employee (empNr);
96  ALTER TABLE Employee_Ticket
97    ADD INDEX FKEmployee_T659766 (ticketnr),
98    ADD CONSTRAINT FKEmployee_T659766
99    FOREIGN KEY (ticketnr) REFERENCES Ticket (nr);
```

9.2 Aufgabe

Based on the given statements and data, explain the consequences of the following operations:

a)
```
INSERT INTO connection VALUES(2, 5, 'sequel')
```

Der Primärschlüssel der aus `from_movie, to_movie` besteht, ist bereits vorhanden und mit dem `connection_type` 'parody' belegt. Daher wird bei diesem INSERT-Statement nichts verändert.

b)
```
DELETE FROM actor WHERE role = 'forest ranger'
```

Alle Zeilen in der Relation **actor** in denen ein Schauspieler die `role` 'forest ranker' besitzt, werden gelöscht. Alle anderen Relationen werden nicht verändert.

c)
```
DELETE FROM movie WHERE title = 'Adventures with RDB'
```

Der Eintrag über den Film 'Adventures with RDB' wird nicht entfernt, da die Standardeinstellung (siehe Vorlesungsfolie 20) bei Löschung keine Aktion ausgeführt werden soll, sowie bei Änderung eines Eintrag ebenfalls nichts ausgeführt werden soll.

d)
```
INSERT INTO actor VALUES(6, 85, 'important looking man')
```

Das Einfügen der Werte (6, 85, 'important looking man') in die Relation **actor** funktioniert nicht, da die `id` '6' in der Relation **person** und in der Relation **movie** die `id` '85' nicht existiert. Daher bleibt die Datenbank unverändert.

e)
```
DROP TABLE person
```

Es passiert nichts, alle Relationen bleiben unverändert, da noch andere Tabellen auf Person referenzieren.