Datenbanken Ubungsblatt 1¨

Aufgabe 1

Sie m¨ochten in einer Tabelle in einer Datenbank speichern, an welchem Datum Sie sich mit welcher Person an welchem Ort getroffen haben.

1. Geben Sie das Schema dieser Datenbank.
2. Geben Sie ein Beispiel f¨ur eine Instanz dieses Schemas.

solution 1

```
create table meetings (
    n_id integer primary key autoincrement,
    s_name varchar(512),
    s_location varchar(512)
);
insert into meetings (
    s_name,
    s_location
)
values
(
    "Grafted Scion",
    "Chapel of Anticipation"
),
(
    "Margit",
    "Stormhill"
)
```

Aufgabe 2

1. Erweitern Sie das Beispiel aus der letzten Aufgabe: Sie wollen in der selben Tabelle auch noch die Telefonnummer der Personen speichern.
2. Erklären Sie anhand dieses Beispiels Redundanz in einer Datenbank.
3. Erklären Sie, wie aus dieser Redundanz Inkonsistenz entstehen kann.

solution 2

```
alter table meetings
add column n_phone_number int
```

```
insert into meetings (
    s_name,
    s_location,
    n_phone_number
)
values
(
    "Margit",
    "Stormhill",
    0313212121
)
```

```
1|Grafted Scion|Chapel of Anticipation|
2|Margit|Stormhill|
3|Margit|Stormhill|313212121
```

wenn nun die zb die person "margit" wieder getroffen wird, ist sie redundant in der tabelle weil ihr gleicher name 2 mal gespeichert ist, ausserdem sind die daten inkonsistent weil im 2. eintrag ihre telefonnummer vorhanden ist, beim ersten jedoch nicht.

redundanz

wuerde der name margit in einem record geaendert werde, muesste jeder record mit diesem namen geandert werden um konsistenz zu bewahren. das selbe gilt fuer die location "stormhill"

Aufgabe 3

Gegeben seien die Tabellen 'instructor' undde 'partmentals' CSV-Dateien. Implementieren Sie folgende Abfragen einer Ihnen genehmen Programmiersprache:

1. select name from instructor where ID = '22222'
2. select building from instructor, department where instructor.dept_name = department.dept_name and name = 'Einstein'

Warum nutzt man eine deklarative Sprache wie SQL zur Datenbankabfrage? Warum nicht eine prozedurale Programmiersprache wie Java oder Python?

solution 3

instructors.csv

```
ID,name,dept_name
1,GLaDOS,aperture science
22222,Einstein,LHEP
```

department.csv

```
ID,dept_name,building
1,aperture science,Aperture Laboratories
2,LHEP,G6
```

python code 1.

```
select name
    from instructor
    where ID = '22222'
```

```
def index_of(s_haystack, s_searchterm):
    n_index = -1
    try:
        n_index = s_haystack.index(s_searchterm)
        return n_index
    except:
        return -1

a_instructor_records = open("instructor.csv", "r").readlines()
a_result = list(filter(lambda s_record:  index_of(s_record, "22222,") == 0,
a_instructor_records))
print("-----records-----")
print("\n".join(a_result))
```

python code 2.

```
select building
    from instructor, department
    where instructor.dept_name = department.dept_name and
       name = 'Einstein'
```

```
a_instructor_records = open("instructor.csv", "r").readlines()
a_records_instructor = list(filter(lambda s_record: index_of(s_record,
",Einstein,") != -1,a_instructor_records))

a_instructor_columnnames = a_instructor_records[0].strip().split(",")


a_department_records = open("department.csv", "r").readlines()
a_department_columnnames = a_department_records[0].strip().split(",")
a_department_records_results = []
a_instructor_records_results = []


for s_instructor_record in a_records_instructor:
    a_recs =  list(filter(lambda s_record: (index_of(s_record,
```

```
s_instructor_record.split(",")
[a_instructor_columnnames.index("dept_name")]) != -1),
a_department_records))
    if(len(a_recs) > 0):
        a_department_records_results = a_department_records_results +
a_recs
        a_instructor_records_results.append(s_instructor_record)


s_colname = "building"

print("---records instructor ---")
n_col_index = 0
try:
    n_col_index = a_instructor_columnnames.index(s_colname)
    for s_record in a_instructor_records_results:
        print(
            s_record.split(",")[n_col_index]
        )
except:
    print(s_colname+ ": column does not exist in instructor.csv")



# print(n_col_index = a_department_columnnames.index(s_colname))
print("---records department ---")
n_col_index = 0
try:
    n_col_index = a_department_columnnames.index(s_colname)

    for s_record in a_department_records_results:
        # print(s_record)
        print(s_record.split(",")[n_col_index])
except:
    print(s_colname+ ": column does not exist in department.csv")
```

oupyt of python script

```
-----records-----
22222,Einstein,LHEP
---records instructor ---
building: column does not exist in instructor.csv
---records department ---
G6
```

fazit

wie wir sehen ist der python code hochkomplex, es werden staendig filter funktionen mit lambda verwendet und es muessen oft, je nach komplexitaet der abzubildenden sql abfrage, mehrere

verschachtelte loops erstellt werden um realtationen zwischen verschiedenen tabellen gegenzupruefen

sql ist viel kurzer und intuitiver geschrieben

Aufgabe 4

Gegeben sei ein Array in der globalen Variable a mit Kontostaendnden in nicht-fluechtigem Speicher. Es soll die Integritaetsbedingung gelten, dass die Summe aller Werte des Arrays gleich bleibt. Nun soll das erste Konto geleert und auf das zweite Konto ̈uberwiesen werden. Dazu gibt es die Funktion:

```
void transact(int[] a)
    a[1] = a[1] + a[0];
    a[0] = 0;
```

Erklaeren Sie, wie bei einem Systemabsturz die Integrit ̈atsbedingung verletzt wer- den kann.

Erweitern Sie die Funktion auf geeignete Weise, um sicherzustellen, dass die Integrit ̈atsbedingung zu jedem Zeitpunkt gilt.

problem

wenn das programm nach zeile `a[1] = a[1] + a[0];` abstuerzt , dann hat das konto `a[1]` einen groesseren wert und `a[0]` wurde nicht geleert, die summe aller konten ist somit groesser als vor dem funktionscall transact, die integriaets bedingung wurde verletzt.

loesung

wir kopieren das ganze array in ein temporaeres array und nehmen dort alle manipulationen vor, erst ganz am schluss wenn alle manipulationen erfolgt sind, koennen wir das temporaere array zurueck kopieren, wenn der code vorher abstuerzt sind nur die temporaeren daten betroffen, die originaldaten bleiben unveraendert, die integritaet ist gesichert!

```
void transact(int[] a)

    int a_tmp[sizeof(a)];
    // do the manipulation on a temporary copy
    memcpy(a, a_tmp, sizeof(a));
    a_tmp[1] = a_tmp[1] + a_tmp[0];
    a_tmp[0] = 0;

    // if we reached this line , no crash has happened
    memcpy(a_tmp, a, sizeof(a_tmp));
```