Implementierung kleiner Datenbanken unter UNIX I (SS 1999)

Abgabetermin 19. Mai 1999

1 Massen-eMails, aber persönlich ! (7 Punkte)

Schreiben Sie mit einem Shell-Skript einen Mailverteiler, der aus einer Datei die email-Adressen der Empfänger und aus einer zweiten den zu verschickenden Mailtext liest. Damit der Empfänger glaubt, man würde sich persönlich für ihn interessieren, sollte auch eine Zeile mit seinem Namen auftauchen, etwa: *Hallo Michael Lehn,* und natürlich kann man auch das Subject als Argument übergeben. Das könnte dann etwa so aussehen:

```
thales$ rundmail passwd cafu.txt "Am 11.5.99 ist Cafu-Nacht!"
sent to wjeberle
sent to s_aberge
.
.
sent to lehn
.
.
```

Und ich lese dann:

```
Hallo Michael Lehn,

am 11. Mai findet im H4/5 die Cafu-Nacht statt,
und damit Du's nicht vergisst, lade ich Dich hiermit
pers"onlich ein.

        mfg
               Captain Future
```

ACHTUNG: Diese Mail könnte teuer werden...
Falls Sie diese Aufgabe testen und dabei hunderte von e-mails an Leute schicken (die Sie nicht kennen, und die sich dann beim Admin. beschweren) könnte Ihnen diese Aufgabe Ihr Account und Studienplatz kosten. Also bitte zum Testen Dateien mit wenigen, Ihnen bekannten e-mail-Adressen verwenden. NO SPAM!

2 Wie kommen die Daten in die Tabelle: INSERT (7 Punkte)

Wie im letzten Übungsblatt sollen Tabellen durch zwei Dateien definiert werden: einer Headerdatei und einer Datei, die die eigentlichen Daten enthält.
Um allerdings ein elegantes Skript zu schreiben, das Zeilen in die Tabelle einfügen kann, muß die Headerdatei zusätzliche Informationen enthalten. Eine Zeile darf nur dann eingefügt, wenn keine andere Zeile den gleichen Schlüssel besitzt. Deshalb soll in der Headerdatei in einer zusätzlichen Zeile eine Liste aller Schlüsselattribute stehen. *passwd.header* würde also so aussehen:

```
:
login:user_id:group_id:Name:Home:Shell
login
```

Ein weiteres Problem taucht auf, wenn mehrere User auf die gleiche Tabelle zugreifen wollen, deshalb sollte man darüber nachdenken, wie man dies synchronisieren könnte...
Schreiben Sie unter dieser Voraussetzung ein Shell-Skript, das einen Datensatz in eine Tabelle einfügt, oder mit einem eindeutigen Exit-Status einen Fehler anzeigt.
Fehler könnten sein:

- Exit 1: Falsche Anzahl der Attribute
- Exit 2: Datensatz mit gleichem Schlüssel existiert
- Exit 3: Tabelle ist ge*lock*t

3 Tabellen anlegen: CREATE (7 Punkte)

Da die Sache mit der Headerdatei anscheinend immer komplizierter wird, sollte man sich langsam überlegen, ob man nicht auch dafür ein Skript schreiben könnte. Aber natürlich müssen Sie nicht lange überlegen, denn Sie wissen: Man kann!
Dem Skript sollte der gewünschte Tabellenname, der Spaltentrenner und eine Liste der Attribute übergeben werden. Fängt ein Attributname mit einem besonderen Metazeichen z.B. − an, wird das Attribut als Teil des Schlüssels interpretiert. Und so sieht's aus, wenn's funktioniert:

```
thales$ ./CREATE passwd \: -login user_id group_id Name Home Shell
thales$ cat passwd.header
:
login:user_id:group_id:Name:Home:Shell
login
thales$
```

4 Selektion aus mehreren Tabellen (7 Punkte)

Schreiben Sie ein neues SELECT, das zwei Tabellen verknüpfen kann. Die Handhabung soll so aussehen:

```
thales$ cat person.header
#
login#Name
login
thales$
thales$ cat person.data
lehn#Michael Lehn
schwedi#Stefan Geschwentner
karmann#Juergen Karmann


thales$ cat ist_hiwi.header
#
Vorlesung#login
Vorlesung#login
thales$
thales$ cat ist_hiwi.data
soft#karmann
prog#schwedi
udb#lehn
stat#schwedi


thales$ SELECT2 person.login=ist_hiwi.login \# ist_hiwi.Vorlesung person.Name
soft#Juergen Karmann
udb#Michael Lehn
prog#Stefan Geschwentner
stat#Stefan Geschwentner
```

Im Detail heißt das:

- SELECT2 erwartet als erstes Argument die Namen der Spalten, über die ein JOIN durchgeführt werden soll (Spalte *login* aus der Tabelle *person* und Spalte *login* aus der Tabelle *ist_hiwi*)
- das zweite Argument bestimmt den Spaltentrenner für die Ausgabe
- alle weiteren Argumente bestimmen die Spalten der erzeugten Tabelle