Aufgabe 5.13 (Trigger)
In Aufgabe 3.2 wurden bei der Löschung eines Eintrags in *Mountain* auch die entsprechenden Einträge in *geo_Mountain* gelöscht. Programmieren Sie umgekehrt nun einen Trigger, der bei Löschung des letzten Eintrages für einen Berg in *geo_Mountain* den entsprechenden Berg auch in *Mountain* löscht.

Aufgabe 5.14 (Trigger)
Lösen Sie Aufgabe 2.6 mit Hilfe einer Prozedur, die eine geeignete Hilfsrelation füllt. Wer will, kann es auch mal mit Triggern versuchen ...

Aufgabe 5.15 Erzeugen Sie eine Relation, die für jeden Fluss die folgenden Daten enthält:

- Denjenigen Fluss, in dem das Wasser schlussendlich in ein Meer/See fließt, sowie
- dieses/n Meer/Lake.

Aufgabe 5.16 (Schiffsverbindungen)
Bestimmen Sie alle Häfen, die Sie nicht erreichen können, wenn Sie in Hamburg mit einem Schiff losfahren. Nehmen Sie dazu an, dass jede Stadt, die an einem Gewässer liegt, einen Hafen hat.

Aufgabe 5.17 (Flugstreckennetz; 50 P.)
Sie wollen ein kleines Flugzeug kaufen, mit dem Sie – ausgehend von Berlin, was aber eigentlich egal ist – alle Städte der Welt erreichen können. Nehmen Sie dazu an, dass Sie in jeder beliebigen Stadt zwischenlanden und nachtanken können. Wie groß muss die Reichweite des Flugzeugs mindestens sein? Verwenden Sie dazu die Tabelle *dbis.Distances*, die public lesbar sein sollte und zu der es bereits ein systemweites Synonym *distances* gibt. Diese wurde folgendermaßen erzeugt:

```
CREATE TABLE Distances
(City1    VARCHAR2(35),
 Country1 VARCHAR2(4),
 City2    VARCHAR2(35),
 Country2 VARCHAR2(4),
 dist     NUMBER);
CREATE INDEX Dist_dist  ON Distances (dist);
CREATE INDEX Dist_City1 ON Distances (City1, Country1);
CREATE INDEX Dist_City2 ON Distances (City2, Country2);
```

Verwenden Sie auf Tabellen, die Sie selbst erstellen, ebenfalls Indexe.