5. Versuch: PL/SQL, Prozeduren, Funktionen, Trigger

Verwenden Sie für die folgenden Aufgaben eine Datenbasis *ohne* Referentielle Integritätsbedingungen; d. h. lassen Sie `create.sql` einmal laufen.

Aufgabe 5.1 (Rekursive Funktion; 10 P)
Schreiben Sie eine rekursive Funktion, die die Länge eines Flusssystems berechnet.

Aufgabe 5.2 (Transitive Hülle; 10 P.)
Schreiben Sie eine Prozedur, die ein Land L und eine Organisation O als Argument hat, die ausgehend von L alle Länder bestimmt, die sich über eine Landverbindung erreichen lassen, ohne dabei durch ein Land reisen zu müssen, das Mitglied in O ist. Schreiben Sie das Ergebnis in eine einspaltige Tabelle.

Aufgabe 5.3 (Cursor; 10 P.)
Berechnen Sie, welcher Anteil der Weltbevölkerung mindestens notwendig ist, um 50% des Welt-Bruttoinlandsproduktes zu erzeugen. Nehmen Sie dabei an, dass sich das Bruttoinlandsprodukt eines Landes gleichmäßig auf alle Einwohner verteilt. Geben Sie an, wieviele Einwohner aus welchen Ländern beteiligt sind. Verwenden Sie einen Cursor über ein geeignet gewähltes `SELECT`-Statement.

Aufgabe 5.4 (Berechnung abhängiger Werte; 10 P.)
Erweitern Sie die Relation *Country* um eine Spalte *Density*, die immer den aktuellen Wert *Population/Area* enthält. Schreiben Sie dafür eine Prozedur, die die entsprechende Spalte füllt. Schreiben Sie zusätzlich zwei Trigger, die das folgende leisten:

1. Wird die Einwohnerzahl oder Fläche verändert, so wird der neue Wert der Bevölkerungsdichte berechnet.
2. Wird die Einwohnerzahl verändert, so wird der Wert für das Bevölkerungswachstum (*Population_Growth* in der Tabelle *Population*) neu berechnet (Gegebene Werte: Stand 1996).

Aufgabe 5.5 (Referentielle Aktionen über Trigger; 10 P.)
In Aufgabe 3.2 haben Sie referentielle Integritätsbedingungen und entsprechende löschende Aktionen über Constraints implementiert. Schreiben Sie nun Trigger, die bei der Umbenennung eines Landes (Country.Code) oder einer Provinz (Province.Name) diese Umbenennungen in der gesamten Datenbasis ausführen.

Aufgabe 5.6 (Flugstreckennetz; 50 P.)
Sie besitzen ein Flugzeug mit x km Reichweite.

a) Schreiben Sie eine Prozedur (mit Aufrufparametern x für die Reichweite sowie s1 und s2 für zwei Städte (inklusive Namen und Code), die folgendes berechnet und in eine von Ihnen vorher erstellte Hilfstabelle schreibt:
   - Die Länge des kürzesten Reiseweges in km von s1 nach s2,
   - Die Länge desjenigen Weges in km, der die wenigsten Zwischenlandungen benötigt.

   Schreiben Sie alle Zwischenlandungen auf den berechneten Wegen in jeweils eine Tabelle.

b) Nehmen Sie zwei Parameter hinzu:
   - t1: Zeit, die Sie pro Zwischenlandung benötigen,
   - v: Reisegeschwindigkeit Ihres Flugzeuges

und berechnen Sie unter diesen Bedingungen jeweils den *schnellsten* Weg von s1 nach s2.

c) Erweitern Sie die Berechnung um einen weiteren Parameter:
   - tc: Zollabfertigungszeit bei grenzüberschreitenden Flügen (Annahme: Bei jedem Zwischenstop in einem anderen Land sind zeitraubende Formalitäten zu erledigen)

   und untersuchen Sie, ob sich die berechneten Wege ändern.

Experimentieren Sie mit unterschiedlichen Start-Ziel-Kombinationen und Reichweiten, z.B. um zu sehen, wie sich die Strecke aufgrund geringerer Reichweite ändert, oder wie sich die optimale Strecke aufgrund unterschiedlicher Geschwindigkeiten ändert.

Wie verhält sich Ihr Algorithmus, wenn die Reichweite zu klein ist, um das Ziel überhaupt zu erreichen (z.B. Honolulu)?

Nehmen Sie an, dass Sie in jeder beliebigen Stadt zwischenlanden und nachtanken können. 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.

Abgabe bis 01.04.2005