4. 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 4.1 (Rekursive Funktion; 10 P)
Schreiben Sie eine rekursive Funktion, die die Länge eines Flusssystems (d.h. eines Flusses sowie aller Flüsse, die direkt oder indirekt in ihn hineinfliessen) berechnet.

a) Berücksichtigen Sie dabei erst einmal Seen *nicht* (also ignorieren Sie, dass ein Fluss in einen See fließt, dessen Abfluss ein anderer Fluss ist).
b) Überlegen Sie, wie Sie Ihre Lösung am besten auf Flüsse, die in einen See fließen, dessen Abfluss ein anderer Fluss ist, erweitern. Setzen Sie Ihre Überlegung praktisch um.

Aufgabe 4.2 (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 4.3 (Trigger; 10 P)
In Aufgabe 3.2 wurden bei der Löschung eines Flusses/Sees/Meeres die entsprechenden Tupel in *geo_River/geo_Lake/geo_Sea* ebenfalls gelöscht. Überlegen Sie sich, welche Updates bei Löschung eines Flusses/Sees/Meeres bezüglich der Relation *located* sinnvoll ist. Implementieren Sie diese durch Trigger und entfernen Sie die entsprechende referentielle Aktion aus Ihrer Lösung zu Aufgabe 3.2.

Aufgabe 4.4 (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 auch in der Tabelle *CountryPops* abgelegt, und der Wert für das Bevölkerungswachstum (*Population_Growth* in der Tabelle *Population*) neu berechnet.

Aufgabe 4.5 (Flugstreckennetz; 50 P.)

Die Basisdaten zu den Flughäfen sind in der Tabelle "Airport" zu finden.

Sie wollen ein kleines Flugzeug kaufen, mit dem Sie – ausgehend von Frankfurt (FRA), was aber eigentlich egal ist – alle Flughäfen der Welt erreichen können. Nehmen Sie dazu an, dass Sie an jedem beliebigen Flughafen 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. Das Tabellenschema wurde folgendermaßen erzeugt:

```
CREATE TABLE Distances
(Code1 VARCHAR2(50),
 Code2 VARCHAR2(4),
 dist NUMBER);
CREATE INDEX Dist_dist ON Distances (dist);
CREATE INDEX Dist_A1 ON Distances (Code1);
CREATE INDEX Dist_A2 ON Distances (Code2);
CREATE PUBLIC SYNONYM DISTANCES FOR dbis.distances;
-- Note: the table is not stored symmetrically
```

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

Frage: Überlegen Sie, wie sich das Ergebnis verändert, wenn mehr Flugplätze in der Datenbank enthalten sind (Mondial enthält "nur" ca. 1400 Verkehrsflughäfen). Versuchen Sie, Ihre Aussage anhand der Daten zu validieren.

Aufgabe 4.6 (Meta-Anfragen und Dynamic SQL; 30 P.)
Mit Dynamic SQL kann man Anfragen dynamisch als Strings zusammensetzen und ausführen. Dies kann insbesondere verwendet werden, um auf Basis der Informationen des Data Dictionaries SQL-Befehle über Tabellen, Spalten oder Constraints laufen zu lassen.

Schreiben Sie ein PL/SQL-Programm (also einen anonymen Block), der hilft, bei einem Entwurf zu validieren, ob man alle Foreign Key-Bedingungen definiert hat:

- Für jedes Paar von (tab_1, col_1) und (tab_2, col_2), von *VARCHAR2*-wertigen Spalten, wenn die Menge der Werte von (tab_1, col_1) eine Teilmenge der Werte von (tab_2, col_2) ist, und die beiden Spalten von keiner (möglicherweise mehrspaltigen!) referentiellen Integritätsbedingung abgedeckt wird, dann gebe das Paar aus.
- Solche "fehlenden", aber nicht benötigten Paare kann es aus verschiedenen Gründen (welche fallen Ihnen auf?) geben.
  Erweitern Sie die obige Lösung so, dass zumindest einige dieser Fälle nicht ausgegeben werden.

Verwenden Sie die folgenden Tabellen des Data Dictionaries: `tabs`, `cols`, `user_constraints` und `user_cons_columns` (Die Dokumentation dazu finden Sie am besten im Web)

Das Programm können Sie dann gleich verwenden, um zu validieren, ob Sie in Aufgabe 3.2 alle sinnvollen Integritätsbedingungen definiert haben.

Abgabe bis 14.6.2024