3. Versuch: ER-Modell, Integrität, verschachtelte Tabellen, Rekursion

Hinweise

Wir möchten nochmal darauf hinwiesen, dass Sie jederzeit die Möglichkeit haben, uns bei Unklarheiten zu den Aufgaben zu fragen.

Aufgabe 3.1 (ER-Modelle; 6+8+9+17 P.)
In Ihrem Repository finden Sie drei ER-Diagramme. Das zugrundeliegende Szenario ist eine etwas vereinfachte Version des Szenarios aus Aufgabe 1.1, hier nochmals die wichtigsten Angaben:

- In den Bundesligen wird jeder gegen jeden mit Hin- und Rückrunde gespielt.
- An der ersten und zweiten Bundesliga nehmen jeweils 18 Mannschaften teil.
- In den Ligen gibt es direkte Auf- und Absteiger und Relegationsspiele, in denen um den Aufstieg bzw. gegen den Abstieg gespielt wird. Relegationsrunden sind aufgeteilt in Hin- und Rückspiel und haben danach immer einen Sieger.
- Pokalwettbewerbe werden im Einzelausscheidungsverfahren (K.O.-System) ausgetragen.
- Vereine können nicht gegen sich selbst antreten, auch wenn sie mehrere Mannschaften haben.

a) Kann der gewünschte Sachverhalt in jedem der Diagramme ausgedrückt werden (in dem Sinn, dass keine gültige Instanz ausgeschlossen wird)? Antworten Sie kurz.

b) Beschreiben Sie kurz die hauptsächlichen Unterschiede zwischen den Diagrammen 1 und 2 sowie zwischen 2 und 3.

c) Beschreiben Sie mit Worten für jedes ER-Modell eine Belegung, die nicht zulässig ist und die auch im Fußballszenario nicht vorkommt. Beschreiben Sie dann eine Belegung, die zulässig ist, die aber im Fußballszenario trotzdem nicht vorkommen darf.

d) Transformieren Sie die ER-Modelle in Relationen und erzeugen Sie die entsprechenden Tabellen. Verwenden Sie dafür auch Integritätsbedingungen, die Teil der Tabellendefinition sein können, aber keine Trigger.

Hinweis: Da der Bearbeitungszeitraum für Aufgabe 1.1 verlängert wurde, erhalten Sie die Diagramme erst nach dem Ablauf dieser Frist, oder wenn Sie Ihre Lösung voher abgeben.

Aufgabe 3.2 (Definition von referentiellen Integritätsbedingungen; 30 P.)
Das Datenbankschema MONDIAL enthält bisher keine referentiellen Integritätsbedingungen. Schreiben Sie nun ein Skript auf Basis von `create.sql`, das die MONDIAL-Datenbank mit referentiellen Integritätsbedingungen erstellt. Schreiben Sie dafür ein Skript `add-constraints.sql`, in dem Sie mit `ALTER TABLE` die Integritätsbedingungen hinzufügen. Sie können aber auch die jeweiligen Tabellendefinitionen ändern. Erstellen Sie außerdem ein Skript `drop-constraints.sql`, das diese referentiellen Integritätsbedingungen löscht.

Berücksichtigen Sie *alle* Constraints, die in der MONDIAL-Miniwelt sinnvoll sind. Es folgen einige Beispiele:

- Wenn ein Land oder eine Provinz gelöscht wird, soll alles, was in diesem Land(esteil) liegt, auch gelöscht werden. Das gleiche gilt für alle Daten, die nur in Zusammenhang mit einem Land(esteil), einem Kontinent oder einer Stadt relevant sind.
- Eine Stadt kann nicht gelöscht werden, wenn sie Hauptstadt von einem Land ist, das nicht gelöscht wird, oder wenn eine Organisation ihren Sitz dort hat.
- Eine Organisation kann nur gelöscht werden, wenn sie keine Mitglieder besitzt.
- Nachbarschaftsbeziehungen entfallen, wenn eines der beteiligten Gebiete gelöscht wird.
- Mit der Löschung eines Berges o.ä. wird auch seine geographische Lage überflüssig.
- Es darf keine kaskadierenden Löschungen von Informationen geben, die von der vom Benutzer geforderten Löschung unabhängig sind.

Hinweise: Berücksichtigen Sie die Tatsache, dass beim Einspielen der Tupel in der Datenbasis einige referentielle Integritätsbedingungen zeitweilig verletzt werden können. Zum Löschen der Datenbank dient das Skript `drop-mondial.sql`. Das MONDIAL-Schema *ohne* Integritätsbedingungen wird von `create-mondial.sql` erstellt.

Aufgabe 3.3 (Nested Tables; 20 P.)

a) Erstellen Sie eine Tabelle *Country_n*, die alle Spalten der Tabelle *Country* umfaßt und zusätzlich zu jedem Land die Informationen zu Sprachen und Religionen jeweils als verschachtelte Tabelle enthält. Füllen Sie die neue Tabelle mit den entsprechenden Daten. **Hinweis:** Dazu ist nur ein Insert-Befehl notwendig. Verwenden Sie `cast` und `multiset`.

b) Schreiben Sie nun eine Anfrage über dieser Tabelle, die alle Länder ermittelt, in denen deutsch gesprochen wird. Neben dem Landesnamen soll dabei der Bevölkerungsanteil in Prozent angegeben werden, der deutsch spricht.

c) Ermitteln Sie den Ausführungsplan für die Anfrage aus Teil b). Welche Rückschlüsse über die tatsächliche Organisation der Tabelle aus a) können Sie daraus ziehen? Halten Sie die Verwendung von verschachtelten Tabellen hier für sinnvoll? Gibt es Szenarien, bei denen dieses Vorgehen besser geeignet ist?

Hinweis: Die Angaben im Skript über unzulässige oder falsche Anfragen bzgl. der verschachtelten Tabellen beziehen sich auf Oracle 8 und treffen so nicht auf Oracle 11 zu.

Aufgabe 3.4 (Transitive Hülle, hierarchische Anfragen; 20 P.)

a) Suchen Sie alle Quellflüsse des Zaire, d. h. alle Flüsse, die direkt oder indirekt in den Zaire fließen. Verwenden Sie eine Hilfsrelation, in der Sie die Flüsse sammeln. `CONNECT BY PRIOR` soll hier noch nicht verwendet werden. Errechnen Sie die Länge des gesamten Flusssystems.

b) Ermitteln Sie das Flusssystem jetzt mit Hilfe von `CONNECT BY PRIOR` (siehe „Hierarchical Queries“ in ORACLEs „SQL Reference“).

c) Diskutieren Sie die Möglichkeiten, mittels `CONNECT BY PRIOR` alle Meere zu erhalten, die mit der Nordsee zusammenhängen (die sozusagen per Schiff erreicht werden können).

d) Verwenden Sie nun `WITH ... SELECT ...` (subquery factoring clause), um das Flusssystem zu berechnen.

Aufgabe 3.5 (Komplexe Attributtypen; 5 P.)

Geben Sie zu jedem Berg in Russland die nächstgelegene Stadt an. Benutzen Sie zur Bestimmung der Entfernung die angegebenen Koordinaten (Vereinfachung: Verwenden Sie den Satz von Pythagoras zur Ermittlung der Distanz).

Hinweis: Die gesuchte Stadt kann auch in einem Nachbarstaat liegen.

Aufgabe 3.6 (Optimierung, 10 P.))

Zählen Sie die Städte, Großstädte und Millionenstädte. Dabei gelten Ortschaften mit weniger als 100 000 Einwohnern als Städte, mit 100 000 oder mehr aber weninger als 1 000 000 Einwohnern als Großstädte. Die verbleibenden größeren Orte fallen in die dritte Kategorie. Zugriffe auf den Sekundärspeicher sind teuer. Schreiben Sie deshalb eine Anfrage, die die *City*-Tabelle für die Berechnung nur einmal liest. **Hinweis:** Im Ausführungsplan können Sie erkennen, wie oft eine Tabelle gelesen wird. Verwenden Sie kein PL/SQL.

Abgabe: 9.6.2010, 11h