JUST DO IT
HÖHERE ABTEILUNG FÜR
INFORMATIONSTECHNOLOGIE

tgm
Die Schule der Technik

Protokoll

Semesteraufgabe — Fussball

SEW & INSY
4AHITM 2015/16

Lorenz Breier

Version 0.1
Begonnen am 2016-03-16
Beendet am 7. April 2016

Note:
Betreuer: Prof. Dolezal, Prof. Borko

Inhaltsverzeichnis

1 Mein Github-Repository

Hier geht's zu meinem Github - Repository.
Na los, hopp, hopp, worauf wartest du?

2 Negative Kompetenzen

- können komplexe Abfragen för konkrete Problemstellungen entwickeln und optimieren
- können Anwendungen mit Datenanbindung entwickeln
- können Anwendungssysteme unter Verwendung von Nebenlöufigkeit entwickeln
- können einfache Schnittstellen zur Kommunikation zwischen Anwendungen entwerfen und implementieren
- können umfangreiche Client-Server Anwendungen entwickeln

3 Angabe

Ein österreichischer Fußballverein braucht eine neue Datenbank, um zumindest die Personenverwaltung auf eine professionelle Basis zu stellen. In dieser Datenbank werden folgende Daten verwaltet:

Jede Person ist identifiziert durch eine eindeutige Personalnummer (kurz "persnr"). Außerdem werden zu jeder Person folgende Informationen verwaltet: Vorname ("vname"), Nachname ("nname"), Geschlecht ("geschlecht") und Geburtsdatum ("gebdat"). Für "geschlecht" sind einzig die Eingaben "W", "M" und "N" gültig.

Jede Person, die am Vereinsleben beteiligt ist, gehört zu genau einer der folgenden 4 Kategorien: Angestellter, Vereinsmitglied (kurz "Mitglied"), Spieler oder Trainer. Für all diese Kategorien von Personen müssen zusätzliche Informationen verwaltet werden:
Von jedem Angestellten werden das Gehalt ("gehalt"), die Überstunden ("ueberstunden") und die E-Mail-Adresse ("e_mail") vermerkt.
Für jedes Vereinsmitglied werden ausständige Beitrage ("beitrag") abgespeichert.
Jeder Spieler hat eine Position ("position") (z.B. Tormann, Stürmer, etc.). Außerdem sind Gehalt ("gehalt") sowie Beginn ("von") und Ende ("bis") der Vertragsdauer abzuspeichern. Jeder Spieler ist zumindest einer Mannschaft zugeordnet. Die Spieler innerhalb einer Mannschaft müssen jeweils eine eindeutige Trikot-Nummer ("nummer") zwischen 1 und 99 haben.
Jeder Trainer hat ein Gehalt ("gehalt") sowie Beginn ("von") und Ende ("bis") der Vertragsdauer. Jeder Trainer ist genau einer Mannschaft zugeordnet.

Der Verein hat mehrere Mannschaften. Für jede Mannschaft sind folgende Informationen zu verwalten: eine eindeutige Bezeichnung ("bezeichnung") (wie 1. Mannschaft, Junioren, A-Jugend, ...), die Klasse ("klasse") sowie das Datum des nächsten Spiels ("naechstes_spiel"). Jede Mannschaft hat mindestens 11 Spieler, genau einen Chef-Trainer und genau einen Trainer-Assistenten. Beide sind als Trainer des Vereins registriert. Außerdem hat jede Mannschaft einen Kapitän (der ein Spieler des Vereins ist).

Der Verein hat mehrere Standorte. Jeder Standort ist identifiziert durch eine eindeutige ID ("sid"). Außerdem sind zu jedem Standort folgende Informationen verfügbar: Land ("land"), Postleitzahl ("plz") und Ort ("ort").

An jedem Standort gibt es 1 oder mehrere Fan-Clubs. Jeder Fan-Club hat einen für den jeweiligen Standort eindeutigen Namen ("name"), ein Gründungsdatum ("gegruendet") und einen Obmann ("obmann"), der Vereinsmitglied sein muss. Ein Vereinsmitglied kann von höchstens einem Fan-Club der Obmann sein. Die Fan-Clubs werden von Angestellten des Vereins betreut: Und zwar kann jeder Angestellte einen Fan-Club jeweils für einen gewissen Zeitraum betreuen. Dieser Zeitraum ist durch Anfangsdatum ("anfang") und Endedatum ("ende") bestimmt. Jeder Angestellte kann 0 oder mehere Fan-Clubs betreuen, und jeder Fanclub kann von einem oder mehreren Angestellten betreut werden.

Der Verein führt Aufzeichnungen über die absolvierten Spiele. Jedes Spiel ist gekennzeichnet durch die Bezeichnung der Mannschaft ("mannschaft") und das Datum ("datum"). Für jedes Spiel werden der Gegner ("gegner") und das Ergebnis ("ergebnis") eingetragen, das einen der Werte "Sieg", "Unentschieden" oder "Niederlage" haben kann. Außerdem werden zu jedem Spiel die beteiligten Spieler abgespeichert, wobei für jeden Spieler die Dauer Einsatzes ("dauer") als Wert zwischen 1 und 90 vermerkt wird.

4 Vorraussetzungen

4.1 Server starten

4.2 SSH - Verbindung

4.3 Postgres

Datenbank, User erzeugen, Rechte verteilen & Einloggen

```
CREATE DATABASE verein;
CREATE USER manager WITH PASSWORD 'iderfrein';
GRANT ALL ON DATABASE verein TO manager;
```

Connection :

```
psql -d verein -h localhost -U manager
```

5 Aufgaben

5.1 CREATE-Befehle

Schreiben Sie die nötigen CREATE-Befehle, um die vorgestellten Relationen mittels SQL zu realisieren. Dabei sind folgende Punkte zu beachten:

5.1.1 Die Datenbank soll keine NULL-Werte enthalten.

Dies lösst sich umsetzen, indem man am Ende des Attributes ein 'NOT NULL' anfögt.
Beispiel:

```
CREATE TABLE angestellter(
    persnr SERIAL PRIMARY KEY REFERENCES person,
    ueberstunden NUMERIC(3,0) NOT NULL,
    gehalt NUMERIC(8,2) NOT NULL,
    e_mail VARCHAR(50) NOT NULL
);
```

5.1.2 Realisieren Sie folgende Attribute mit fortlaufenden Nummern...

...mit Hilfe von Sequences und Serial: "persnr"von Personen und "sid" von Standorten.

- För das "persnr" - Attribut sollen nur gerade, 5-stellige Zahlen vergeben werden (d.h.: 10000, 10002, ..., 99998).

  Sequences kann man sich wie eine Zöhlervariable vorstellen, welche im Hintergrund mitlöuft. Erstellt wird unsere Sequence so :

  ```
  CREATE SEQUENCE pid START WITH 10000 INCREMENT BY 2;
  ```

Die 5 Stellen erhalten wir, indem wir uns an der Option **START WITH** bedienen.
För die geraden Zahlen helfen wir uns mit einem Trick, in dem wir mit Hilfe von **INCREMET BY** immer um 2 erhöhen.
=> da er nur immer um 2 erhöht, können nur gerade Zahlen kommen !

- För das "sid" - Attribut sollen beliebige positive Zahlen (d.h. 1,2, ...) vergeben werden.
  Falls zwischen 2 Tabellen zyklische FOREIGN KEY Beziehungen herrschen, dann sind diese FOREIGN KEYs auf eine Weise zu definieren, dass es möglich ist, immer weitere Datensötze mittels INSERT in diese Tabellen einzufögen.
  Die positiven Zahlen erhalten wir mittels einer **CHECK** - Constraint.

```
sid INTEGER PRIMARY KEY CHECK (sid > 0),
```

5.2 INSERT-Befehle

Schreiben Sie INSERT-Befehle, um Testdaten för die kreierten Tabellen einzurichten.
Jede Tabelle soll zumindest 10.0000 Zeilen enthalten.
Sie dörfen die Wahl der Namen, Bezeichnungen, etc. so einfach wie möglich gestalten, d.h.: Sie mössen nicht "real existierende"Fuöballer-Namen, Lönder, Stödte, etc. wöhlen.
Stattdessen können Sie ruhig 'Spieler 1', 'Spieler 2', 'Land 1', 'Land 2', 'Stadt 1', 'Stadt 2', etc. verwenden.
Sie können för die Erstellung der Testdaten auch entsprechende Generatoren verwenden!
Nicht mein Kompetenzbereich, deshalb erstmal aufgeschoben.

5.3 DROP-Befehle

Schreiben Sie die nötigen DROP-Befehle, um alle kreierten Datenbankobjekte wieder zu löschen.
z.B

```
DROP TABLE person;
DROP TABLE angestellter;
DROP TABLE mitglied;
DROP TABLE trainer;
```

6 Problemlösungen mittels SQL

```
# ========================================
#              SQL - AUFGABEN
# ========================================

# S1.) (Fan-Club Betreuung) Waehlen Sie "per Hand" die Personalnummer eines Angestellten aus Ihren Testdaten aus.

# Auswahl : 10000

# Schreiben Sie eine SQL-Anfrage, die jene Fan-Clubs ermittelt, die dieser Angestellte im Moment nicht betreut.
# Geben Sie zu jedem derartigen Fan-Club die Standort-ID und den Namen des Fan-Clubs aus. DONE
# Bemerkung: Ein Fan-Club wird von einem Angestellten im Moment nicht betreut,
# wenn entweder der Angestellte diesen Fan-Club ueberhaupt nie betreut hat
# oder wenn das heutige Datum (= sysdate) ausserhalb des Betreuungszeitraums liegt.
# Vergessen Sie nicht, jene Fan-Clubs zu beruecksichtigen,
# die von ueberhaupt keinem Angestellten betreut werden (dieser Fall sollte zwar laut Datenmodell nicht vorkommen.
# Die Einhaltung dieser Bedingung wird aber vermutlich vom Datenbanksystem nicht ueberprueft)!

SELECT name, sid
FROM betreuung
WHERE persnr != 10000
AND persnr IS NOT NULL
OR anfang > sysdate
OR ende < sysdate;

# S2.) (Die eifrigsten Angestellten) Schreiben Sie eine SQL-Anfrage, die den Nachnamen und
# die Personalnummer jener Angestellten ausgibt, die im Moment saemtliche Fan-Clubs betreuen. Ordnen Sie die Nachnamen alphabetisch.
# Bemerkung: Passen Sie die Testdaten so an, dass diese Anfrage zumindest zwei Angestellte liefert.

(SELECT nname, persnr
ORDER BY nname DESC
FROM person
INNER JOIN angestellter ON angestellter.persnr = person.persnr)
LEFT OUTER JOIN betreuung ON person.persnr = fanclub.persnr;


# S3.) (Spielereinsaetze) Geben Sie fuer alle Spiele des Jahres 2015 jeweils alle Spieler und die Dauer ihres Einsatzes aus, d.h.: Gesucht sind alle Tupel (mannschaft, datum, vorname, nachname, dauer), mit folgender Eigenschaft:
#         "mannschaft" ist die Bezeichnung der Mannschaft, die gespielt hat.
#         "datum" ist das Datum, an dem das Spiel stattfand.
#         "vorname" und "nachname" beziehen sich auf einen Spieler, der bei diesem Spiel zum Einsatz kam.
#         "dauer" gibt die Dauer des Einsatzes (in Minuten) dieses Spielers bei diesem Spiel an.
SELECT mannschaft, datum, person.vname, person.nname, dauer
FROM einsatz
NATURAL JOIN person
WHERE datum
BETWEEN '2015-01-01'
AND '2015-12-31';


# S4.) (Spieler-Ranking) Geben Sie fuer jeden Spieler den Vornamen und Nachnamen
# sowie die Gesamtdauer ("gesamtdauer") der von ihm bei Spielen im Jahr 2015 geleisteten Einsaetze aus.
# Vergessen Sie nicht, jene Spieler des Vereins zu beruecksichtigen, die im Jahr 2015 bei keinem einzigen Spiel mitgespielt haben (d.h. gesamtdauer = 0).
# Ordnen Sie die Ausgabe in absteigender Gesamtdauer. Bei Gleichheit der Gesamtdauer sollen die Spieler in alphabetischer Reihenfolge (zuerst des Nachnamen, dann des Vornamen) sortiert werden.
SELECT vname, nname, (SUM(einsatz.dauer)) AS gesamtdauer
FROM (SELECT vname, nname
FROM person
NATURAL JOIN spieler) AS spieler
ORDER BY gesamtdauer DESC;

# S5.) (Der fleissigste Spieler) Geben Sie den Vornamen und Nachnamen jenes Spielers aus, von dem die unter
# b) berechnete Gesamtdauer am groessten ist, d.h.: dieser Spieler ist bei Spielen im Jahr 2015 insgesamt am laengsten im Einsatz gewesen. Falls sich mehrere Spieler den ersten Platz teilen (d.h.
```

```
    sie kommen auf die gleiche Gesamtdauer), dann sollen diese in alphabetischer Reihenfolge (zuerst des
     Nachnamen, dann des Vornamen) geordnet werden. Der Fall, dass im Jahr 2015 ueberhaupt kein Spiel
    stattfand, darf ignoriert werden.
# Bemerkung: Beruecksichtigen Sie bei Ihren Testdaten die Situation, dass sich zumindest 2 Spieler den
    ersten Platz teilen.
SELECT vname, nname, (SUM(einsatz.dauer)) AS gesamtdauer
FROM (SELECT vname, nname FROM person NATURAL JOIN spieler) AS spieler,
ORDER BY gesamtdauer DESC
LIMIT 1;


# S6.) Schreiben Sie CREATE und DROP Befehle fuer eine View, die alle Informationen ueber Trainer aus
    der Personen- und Trainer-Tabelle zusammenfuegt, d.h.: sowohl die allgemeinen Personendaten (
    Personalnummer, Vorname, Nachname, Geschlecht und Geburtsdatum) als auch die Trainer-spezifischen
    Informationen (Gehalt sowie Beginn und Ende der Vertragsdauer). In Summe ist also folgende View
    erforderlich:
#    Trainer_view (persnr, vname, nname, geschlecht, gebdat, gehalt, von, bis).
CREATE VIEW trainer_view AS
SELECT persnr, vname, nname, geschlecht, gebdat, gehalt, von, bis
FROM person
NATURAL JOIN trainer;

DROP VIEW Trainer_view;
```

Tabellenverzeichnis

Listings

Abbildungsverzeichnis