JUST DO IT
HÖHERE ABTEILUNG FÜR INFORMATIONSTECHNOLOGIE

tgm
Die Schule der Technik

Semesterprojekt
Fußballverein

Informationssysteme
4AHITM 2015/16

Georg Weinauer

Version 2.0

Note:

Betreuer: M. Borko

Begonnen am 13.04.2016

Beendet am 04.06.2016

Inhaltsverzeichnis

1. Einführung

Diese Übung dient als Semesterprojekt über die Entwicklung einer Datenbank.

1.1. Ziele

Ziel des Projekts ist, fehlende Kompetenzen nachzuholen beziehungsweise negativ bewertete Kompetenzen zu verbessern.

1.2. Voraussetzungen

- Grundwissen über den Datenbankentwurfsprozess
- Grundwissen über die Entwicklung von ER-Diagrammen
- Grundwissen über die Entwicklung von Relationen Modells
- Grundwissen über die Programmierung von DDL Scripts in SQL
  Grundwissen über die Programmierung von Select Statements

1.3. Aufgabenstellung

2. 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.

Aufgabenstellung

1.) Schreiben Sie die nötigen CREATE-Befehle, um die vorgestellten Relationen mittels SQL zu realisieren. Dabei sind folgende Punkte zu beachten:
* Die Datenbank soll keine NULL-Werte enthalten.
* Realisieren Sie folgende Attribute mit fortlaufenden Nummern mit Hilfe von Sequences: "persnr" von Personen und "sid" von Standorten. Für das "persnr"-Attribut sollen nur gerade, 5-stellige Zahlen vergeben werden (d.h.: 10000, 10002, ..., 99998). Für 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 Datensätze mittels INSERT in diese Tabellen einzufügen.

2.) Schreiben Sie INSERT-Befehle, um Testdaten für die kreierten Tabellen einzurichten. Jede Tabelle soll zumindest 100000 Zeilen enthalten. Sie dürfen die Wahl der Namen, Bezeichnungen, etc. so einfach wie möglich gestalten, d.h.: Sie müssen nicht "real existierende" Fußballer-Namen, Länder, Städte, etc. wählen. Stattdessen können Sie ruhig 'Spieler 1', 'Spieler 2', 'Land 1', 'Land 2', 'Stadt 1', 'Stadt 2', etc. verwenden. Sie können für die Erstellung der Testdaten auch entsprechende Generatoren verwenden!

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

Lösen Sie die folgenden Probleme mittels SQL:

S1.) (Fan-Club Betreuung) Wählen Sie "per Hand" die Personalnummer eines Angestellten aus Ihren Testdaten aus. 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.
Bemerkung: Ein Fan-Club wird von einem Angestellten im Moment nicht betreut, wenn

entweder der Angestellte diesen Fan-Club überhaupt nie betreut hat oder wenn das heutige Datum (= sysdate) außerhalb des Betreuungszeitraums liegt. Vergessen Sie nicht, jene Fan-Clubs zu berücksichtigen, die von überhaupt keinem Angestellten betreut werden (dieser Fall sollte zwar laut Datenmodell nicht vorkommen. Die Einhaltung dieser Bedingung wird aber vermutlich vom Datenbanksystem nicht überprüft)!

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

S3.) (Spielereinsätze) Geben Sie für 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.

S4.) (Spieler-Ranking) Geben Sie für jeden Spieler den Vornamen und Nachnamen sowie die Gesamtdauer ("gesamtdauer") der von ihm bei Spielen im Jahr 2015 geleisteten Einsätze aus. Vergessen Sie nicht, jene Spieler des Vereins zu berücksichtigen, 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.

S5.) (Der fleißigste Spieler) Geben Sie den Vornamen und Nachnamen jenes Spielers aus, von dem die unter b) berechnete Gesamtdauer am größten ist, d.h.: dieser Spieler ist bei Spielen im Jahr 2015 insgesamt am längsten 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 überhaupt kein Spiel stattfand, darf ignoriert werden.
Bemerkung: Berücksichtigen Sie bei Ihren Testdaten die Situation, dass sich zumindest 2 Spieler den ersten Platz teilen.

S6.) Schreiben Sie CREATE und DROP Befehle für eine View, die alle Informationen über Trainer aus der Personen- und Trainer-Tabelle zusammenfügt, 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).

Datenbankclient (Java/C++) und DB-Connector (JDBC/libpqxx):

Schreiben Sie einen Client, der eine Datenbank-Verbindung herstellt. Realisieren Sie eine

GUI (JavaFX/Qt), die das einfache Ändern (CRUD) der Spieler des Vereins erlaubt. Verwenden Sie dabei auf jeden Fall eine Tabelle (TableView, QTableView), die auch eine grafische Veränderung der Datensätze erlauben soll.

Ermöglichen Sie die gleichzeitige Verbindung von mehreren Clients auf die Datenbasis. Implementieren Sie dabei eine transaktionelle, gesicherte Erstellung und Änderung von Spielen. Beachten Sie dabei, dass der Spielstand und die Spielzeit der einzelnen Spieler laufend und von mehreren Clients gleichzeitig aktualisiert werden könnte. Stellen Sie für die Eingabe der Spielerzeit und Spielstand eine einfache grafische Möglichkeit zur Verfügung. Verwenden Sie dabei Transaktionen bzw. Locks und entsprechende programmtechnische Mittel um Inkonsistenzen zu vermeiden. Definieren Sie dabei für die einzelnen Informationen (Spielerzeit, Spielstand) eigene Threads.

3. Informationssysteme und Content Management:
4. Versuchen Sie in einer kurzen schriftlichen Ausführung anzuführen, welche Informationssysteme im betrieblichen Umfeld genutzt werden und wie diese für die vorliegende Aufgabenstellung eingesetzt werden könnten. Im Bereich der Informationssysteme wurde im Unterricht im Speziellen auf die Thematik der Content Management Systeme eingegangen. Versuchen Sie auch hier kurz zu erläutern welche der kennengelernten CMS für die vorliegende Aufgabenstellung besonders gut eingesetzt werden könnten und warum.
5. Ausgehend von der vorangegangenen Analyse wählt ein entsprechendes Content Management aus, Implementiert ein passendes Template zu unserem Fußballverein und legt entsprechende Benutzer an die in eurem System erstellen, bearbeiten oder auch nur lesen können sollen. Holen Sie hierbei mittels angelegten Benutzer aus dem Content Management System Datensätze aus der Aufgabe Fußballverein (z.b. Spieler usw.) und zeigt diese entsprechend in eurem CMS an um das Design verdeutlichen sollen.
6. Setzen Sie das entwickelte Rahmenwerk "Fußballverein", das im Zuge der vorliegenden Arbeit generiert wurde, in Form eines Plug-Ins für das von Ihnen gewählte CMS um.

2. Ergebnisse

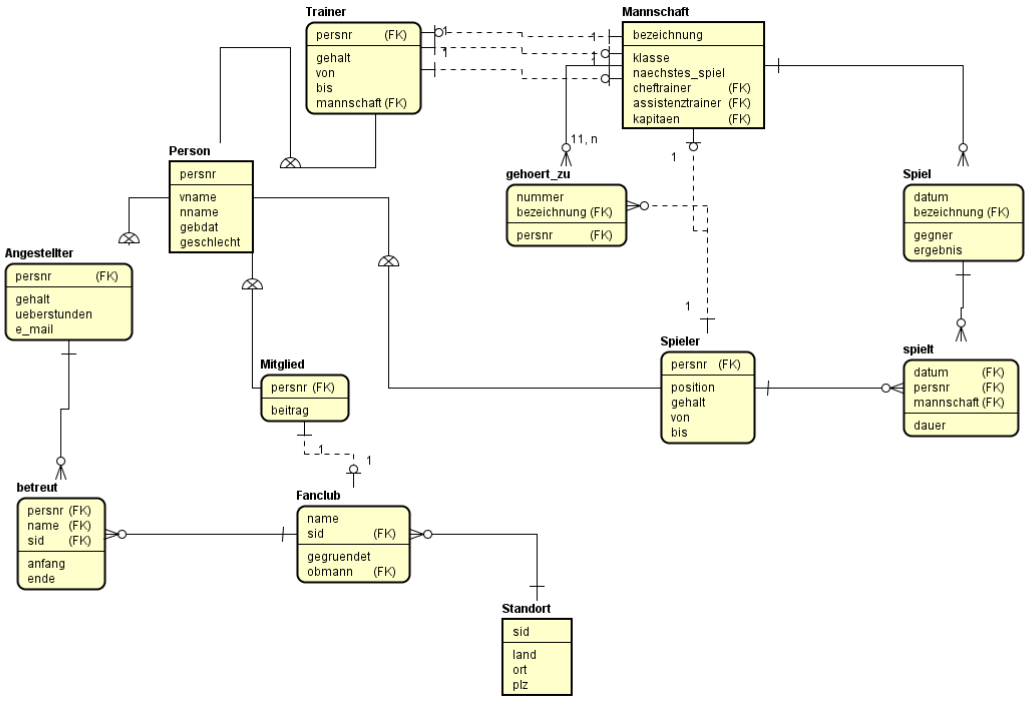
2.1 Datenbank Entwurfsprozess

Der Entwurfsprozess einer Datenbank wird in 4 Schritte unterteilt.

1. Anforderungsanalyse → schriftliche Beschreibung/Vereinbarung
2. Konzeptioneller Entwurf →ER-Diagramm (grafische Form)
3. Logischer Entwurf → Relationen Modell (RM) (Tabellenform)
4. Physikalischer Entwurf → DDL – Script

Diese 4 Schritte werden immer in chronologischer Reinfolge abgearbeitet. Der erste Schritt entspricht in diesem Fall der Angabe online auf E-learning[1].

2.2 Entity Relationship Diagramm



2.3 DDL Script

Im nächsten Schritt, auch physikalischer Entwurf genannt, wird das SQL Create Script erstellt wird. Für das Grundgerüst des Create Script reicht die automatische Generierung von Astah. Dazu einfach unter Tool → ER-Diagramm→ Export SQL

2.3.1 One to One Relationships

Als erstes müssen alle 1-1 Beziehungen modifiziert werden. Die einfachste Möglichkeit so etwas zu tun, ist den Fremdschlüssel UNIQUE zu setzen. Das bedeutet nämlich, dass jeder Fremdschlüssel genau einmal vorkommen kann und 1 to Many Beziehungen nicht mehr möglich sind.

2.3.2 Sequences

Die Datenbank soll zwei Spalten haben (persnr bei Person, sid bei Standort), welche Auto Inkrement, sprich von selbst hoch zählen, sind. Dazu wird der Datentyp SERIAL verwendet. Um nun die Besonderheit bei persnr (fünfstellig, nur gerade Zahlen) zu realisierten wird folgender Befehl benötigt:

```
ALTER SEQUENCE Person_persnr_seq RESTART WITH 10000 INCREMENT BY 2;
```

Erklärung: Eine Sequence wird automatisch angelegt, wenn der Datentyp SERIAL verwendet wird. Sie wird immer Tabellenname_Spaltenname_seq genannt. Die Phrase *RESTART WITH* gibt einen Startwert an und *INCREMENT* BY gibt die Schritgröße an in der gezählt wird.

2.3.3Check

Ein Check Constraint dient dazu Bedienungen für Spalten zu überprüfen. Die Angabe verlangt zum Beispiel, dass in der Tabelle Person nur begrenzte Werte eingefügt werden dürfen. Check Constraints schauen folgendermaßen aus:

```
ALTER TABLE Mannschaft ADD CONSTRAINT PK_Mannschaft PRIMARY KEY (bezeichnung);
```

2.3.4 Hinzufügen der Daten:

Beim Hinzufügen von Daten sollte man stets auf die Reihenfolge achten! Beispielsweise macht es keinen Sinn die Tabelle spielt vor der Tabelle Mannschaft hinzuzufügen.

2.3.5 Löschen der Daten

Für das Löschen der Daten gibt es ein eigenes Script mit Drop Befehlen für die gesamte Datenbank!

2.4 Datafiller

Der Datafiller basiert auf der Idee ein SQL Script einzulesen und über die Kommentare Zusatzinformationen zu addieren. Diese werden mit dem Tag „df:“ gesetzt. Der Datafiller ist ein Python Script das mit *./datafiller –size=wert CreateData.sql > data.sql*

Es wird ein data.sql File generiert in dem die Daten in einer Tabelle gespeichert werden. Dieses wird dann über \i data.sql in die Datenbank kopiert.

2.5 Selects

S1.) (Fan-Club Betreuung) Wählen Sie "per Hand" die Personalnummer eines Angestellten aus Ihren Testdaten aus. 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.
Bemerkung: Ein Fan-Club wird von einem Angestellten im Moment nicht betreut, wenn entweder der Angestellte diesen Fan-Club überhaupt nie betreut hat oder wenn das heutige Datum (= sysdate) außerhalb des Betreuungszeitraums liegt. Vergessen Sie nicht, jene Fan-Clubs zu berücksichtigen, die von überhaupt keinem Angestellten betreut werden (dieser Fall sollte zwar laut Datenmodell nicht vorkommen. Die Einhaltung dieser Bedingung wird aber vermutlich vom Datenbanksystem nicht überprüft)!

```
SELECT * FROM betreut
WHERE persnr!=10006 OR NOT CURRENT_DATE BETWEEN betreut.anfang AND
betreut.ende;
```

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

```
SELECT person.nname, person.persnr FROM betreut INNER JOIN person ON
betreut.persnr=person.persnr
WHERE CURRENT_DATE BETWEEN betreut.anfang AND betreut.ende
ORDER BY person.nname ASC;
```

S3.) (Spielereinsätze) Geben Sie für 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 spielt.mannschaft, spielt.datum, person.vname, person.nname, spielt.dauer
FROM spielt INNER JOIN person ON spielt.persnr=person.persnr
WHERE (SELECT EXTRACT(YEAR FROM spielt.datum))=2016
ORDER BY spielt.mannschaft ASC;
```

S4.) (Spieler-Ranking) Geben Sie für jeden Spieler den Vornamen und Nachnamen sowie die Gesamtdauer ("gesamtdauer") der von ihm bei Spielen im Jahr 2015 geleisteten Einsätze aus. Vergessen Sie nicht, jene Spieler des Vereins zu berücksichtigen, 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 person.vname, person.nname, SUM(spielt.dauer) AS dauer FROM spielt NATURAL
JOIN person
WHERE (SELECT EXTRACT(YEAR FROM spielt.datum))=2015
GROUP BY person.vname, person.nname
UNION
SELECT person.vname, person.nname, spielt.dauer AS dauer FROM spielt NATURAL JOIN
person
WHERE (SELECT EXTRACT(YEAR FROM spielt.datum)) = 2015 AND spielt.dauer = 0
GROUP BY person.vname, person.nname, spielt.dauer
ORDER BY dauer DESC;
```

S5.) (Der fleißigste Spieler) Geben Sie den Vornamen und Nachnamen jenes Spielers aus, von dem die unter b) berechnete Gesamtdauer am größten ist, d.h.: dieser Spieler ist bei Spielen im Jahr 2015 insgesamt am längsten 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 überhaupt kein Spiel stattfand, darf ignoriert werden.
Bemerkung: Berücksichtigen Sie bei Ihren Testdaten die Situation, dass sich zumindest 2 Spieler den ersten Platz teilen.

Leider nicht auf die richtige Lösung gekommen

S6.) Schreiben Sie CREATE und DROP Befehle für eine View, die alle Informationen über Trainer aus der Personen- und Trainer-Tabelle zusammenfügt, 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 trainer.persnr, person.vname, person.nname,
person.geschlecht, person.gebdat, trainer.gehalt, trainer.von, trainer.bis
FROM trainer INNER JOIN person ON trainer.persnr=person.persnr;

DROP VIEW trainer_view;
```