# DBS – Projekt Protokoll

|  |  |  |
| --- | --- | --- |
| **Titel:** | Fußballverein | * FBV |
|  |  |  |
| **Leitung:** | **Carab Lucian-Valentin** | |
| **Projektteilnehmer:** | Pchelnikava Aliaksandra |  |
|  | Seidel Hans |  |
|  | Höller Nicholas |  |
|  |  |  |

## Dokumentbeschreibung:

Dieses Dokument beinhaltet sowohl die Planung als auch die Dokumentation der einzelnen Arbeitsschritte. Außerdem wurde es für das Projektteam als *Mini-Kanban* verwendet.

Das Projekt ist grundlegend in drei Teile aufgegliedert (Planung, Datenbank, Interface), wobei der Datenbanken-Bereich den größten Arbeitsaufwand beinhaltet.

Es wurden mehrere Teammeetings abgehalten und die notwendigen Planungsdokumente erstellt. Da der Fokus allerdings auf Datenbanken-Entwicklung gesetzt war und nicht Projektmanagement wurde auf einen großen Teil der Feinplanung verzichtet.

Inhalt

[DBS – Projekt Protokoll 1](#_Toc57901317)

[Dokumentbeschreibung: 1](#_Toc57901318)

[Planung 3](#_Toc57901319)

[ER-Modell 4](#_Toc57901320)

[Relationen-Modell 5](#_Toc57901321)

[Select-Statements 9](#_Toc57901322)

# Planung

Grundsätzlich haben wir für dieses Projekt in einzelnen Etappen die Planung gestaltet. D.h. begonnen wurde mit der Planung der Datenbank -> dann folgte die Umsetzung der Datenbank. In einem weiteren Schritt wurde die Planung der SELECTS, TRIGGERS, PACKAGES usw. vorgenommen und in Folge wieder die Umsetzung durchgeführt.

Für die Planung wurde ein ER-Modell und ein Relationen Modell erstellt. Das ER-Modell wurde gemeinsam entworfen. Nach dem Entwurf wurden die Create- und Insert-Statements im Team aufgeteilt und jeder hat einen Teil der Arbeit übernommen. Da alle nach dem entworfenen ER-Modell gearbeitet haben, war das Zusammenfügen der einzelnen Teilbereiche mit ziemlich wenig Arbeitsaufwand verbunden.

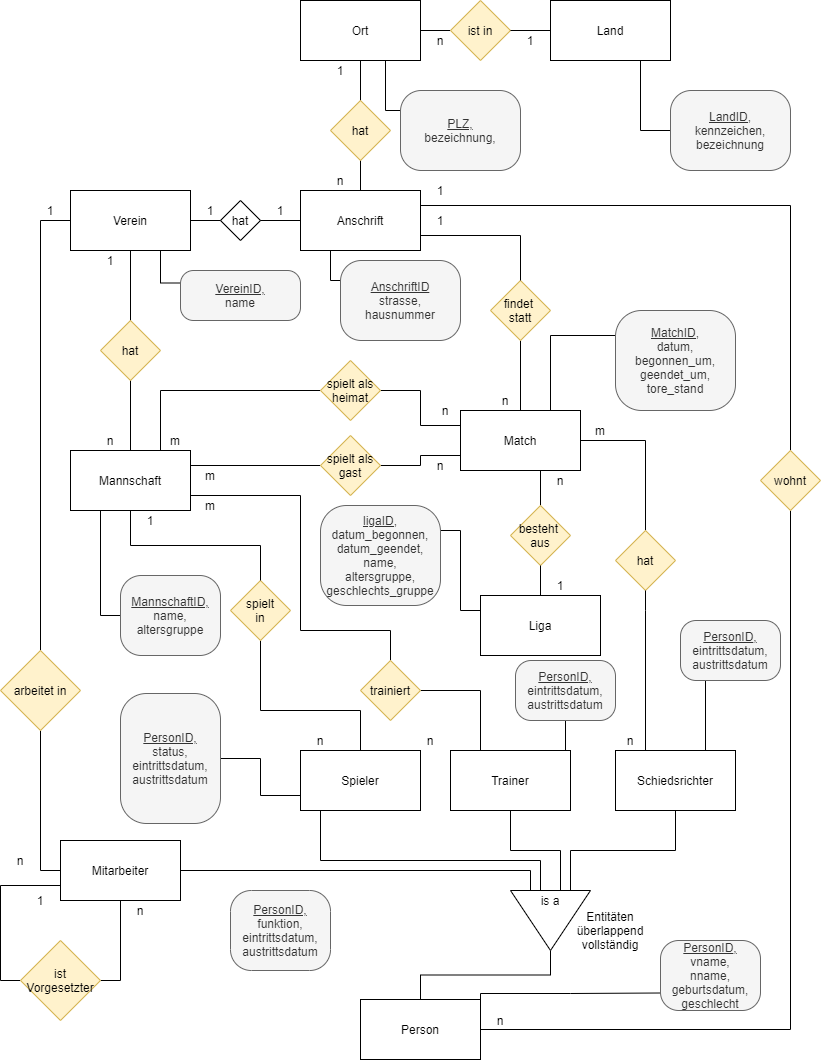
Wir haben für dieses Projekt ein GitHub – Repository erstellt, um distribuiert arbeiten zu können und die Software in verschiedene Version gesichert zu haben.

Um die Testdaten zu generieren haben wir [www.generatedata.com](http://www.generatedata.com) verwendet.

[xx] – Interface Planung

## ER-Modell

Das ER-Modell wurde einmalig verändert und die Veränderungen wurden in der Datenbank aktualisiert. Dies ist das aktuelle ER-Modell.



## Relationen-Modell

Das Relationen-Modell wurde entgegengesetzt der eigentlichen Vorgehensweiße nach dem Erstellen der Datenbank modelliert und nicht aus dem ER-Modell heraus modelliert.

|  |  |  |
| --- | --- | --- |
| **Tabellen-Name:** | Land |  |
|  |  | ***Datentyp*** |
| **Primär-Schlüssel:** | **LandID** | Number |
| **Fremd-Schlüssel:** | ***-*** | - |
| **Attribute:** | Kennzeichen | Varchar (20) |
|  | Bezeichnung | Varchar (255) |
|  |  |  |
| **Tabellen-Name:** | ORT |  |
|  |  | ***Datentyp*** |
| **Primär-Schlüssel:** | **PLZ** | Number |
| **Fremd-Schlüssel:** | ***LandID*** | Number |
| **Attribute:** | Bezeichnung | Varchar(255) |
|  |  |  |
| **Tabellen-Name:** | Anschrift |  |
|  |  | ***Datentyp*** |
| **Primär-Schlüssel:** | **AnschriftID** | Number |
| **Fremd-Schlüssel:** | ***PLZ*** | Number |
| **Attribute:** | Strasse | Varchar(255) |
|  | Hausnummer | Varchar(20) |
|  |  |  |
| **Tabellen-Name:** | Verein |  |
|  |  | ***Datentyp*** |
| **Primär-Schlüssel:** | **VereinID** | Number |
| **Fremd-Schlüssel:** | ***AnschriftID*** | Number |
| **Attribute:** | Name | Varchar(255) |
|  |  |  |
| **Tabellen-Name:** | Liga |  |
|  |  | ***Datentyp*** |
| **Primär-Schlüssel:** | **LigaID** | Number |
| **Fremd-Schlüssel:** | ***-*** | - |
| **Attribute:** | Datum\_Begonnen | Date |
|  | Datum\_Geendet | Date |
|  | Name | Varchar(255) |
|  | Altersgruppe | Number |
|  | Geschlechts\_Gruppe | Char (1) |

|  |  |  |
| --- | --- | --- |
| **Tabellen-Name:** | Mannschaft\_Match |  |
|  |  | ***Datentyp*** |
| **Primär-Schlüssel:** | **HeimmannschaftID**  **GastmannschaftID**  **MatchID** | Number  Number  Number |
| **Fremd-Schlüssel:** | ***-*** | - |
| **Attribute:** | - | - |
|  |  |  |
| **Tabellen-Name:** | Mannschaft\_Trainer |  |
|  |  | ***Datentyp*** |
| **Primär-Schlüssel:** | **MannschaftID**  **TrainerID** | Number  Number |
| **Fremd-Schlüssel:** | ***-*** | - |
| **Attribute:** | - | - |
|  |  |  |
| **Tabellen-Name:** | Match\_Schiedsrichter |  |
|  |  | ***Datentyp*** |
| **Primär-Schlüssel:** | **MatchID**  **SchiedsrichterID** | Number  Number |
| **Fremd-Schlüssel:** | ***-*** | - |
| **Attribute:** | - | - |
|  |  |  |
| **Tabellen-Name:** | Match |  |
|  |  | ***Datentyp*** |
| **Primär-Schlüssel:** | **MatchID** | Number |
| **Fremd-Schlüssel:** | ***AnschriftID***  ***LigaID*** | Number  Number |
| **Attribute:** | Datum | Date |
|  | Begonnen\_um | Timestamp |
|  | Geendet\_um | Timestamp |
|  | Tore\_Stand | Varchar (7) |
|  |  |  |
| **Tabellen-Name:** | Mannschaft |  |
|  |  | ***Datentyp*** |
| **Primär-Schlüssel:** | **MannschaftID** | Number |
| **Fremd-Schlüssel:** | ***VereinID*** | Number |
| **Attribute:** | Name | Varchar (255) |
|  | Altersgruppe | Number |

|  |  |  |
| --- | --- | --- |
| **Tabellen-Name:** | Spieler |  |
|  |  | ***Datentyp*** |
| **Primär-Schlüssel:** | **PersonID** | Number |
| **Fremd-Schlüssel:** | ***MannschaftID*** | Number |
| **Attribute:** | Status | Char(5) |
|  | Eintrittsdatum | Date |
|  | Austrittsdatum | Date |
|  |  |  |
| **Tabellen-Name:** | Trainer |  |
|  |  | ***Datentyp*** |
| **Primär-Schlüssel:** | **PersonID** | Number |
| **Fremd-Schlüssel:** | ***-*** | - |
| **Attribute:** | Eintrittsdatum | Date |
|  | Austrittsdatum | Date |
|  |  |  |
| **Tabellen-Name:** | Schiedsrichter |  |
|  |  | ***Datentyp*** |
| **Primär-Schlüssel:** | **PersonID** | Number |
| **Fremd-Schlüssel:** | ***-*** | - |
| **Attribute:** | Eintrittsdatum | Date |
|  | Austrittsdatum | Date |
|  |  |  |
| **Tabellen-Name:** | Mitarbeiter\_Vorgesetzter |  |
|  |  | ***Datentyp*** |
| **Primär-Schlüssel:** | **PersonID** | Number |
| **Fremd-Schlüssel:** | ***VorgesetzterID*** | Number |
| **Attribute:** | - | - |
|  |  |  |
| **Tabellen-Name:** | Mitarbeiter |  |
|  |  | ***Datentyp*** |
| **Primär-Schlüssel:** | **PersonID** | Number |
| **Fremd-Schlüssel:** | ***VereinID*** | Number |
| **Attribute:** | Funktion | Varchar (50) |
|  | Eintrittsdatum | Date |
|  | Austrittsdatum | Date |

|  |  |  |
| --- | --- | --- |
| **Tabellen-Name:** | Person |  |
|  |  | ***Datentyp*** |
| **Primär-Schlüssel:** | **PersonID** | Number |
| **Fremd-Schlüssel:** | ***AnschriftID*** | Number |
| **Attribute:** | Vorname | Varchar (50) |
|  | Nachname | Varchar (50) |
|  | Geburtsdatum | Date |
|  | Geschlecht | Char (1) |

# Select-Statements

Die Folgenden Select-Statements wurden von den bestehenden Teammitgliedern entwickelt. Dieses Dokument wurde auch als *Mini-Kanban* verwendet, weshalb die untenstehenden Tabellen einen Status Wert mit ABGESCHLOSSEN haben. Die Werte, die verwendet wurden, sind:

|  |  |  |  |
| --- | --- | --- | --- |
| BEREIT | IN ARBEIT | ZUM TESTEN | ABGESCHLOSSEN |
|  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | HÖL | Optional: | N |
| Art: | SELECT | *Name:* | Liga-Matches | | |
| Aufgabe: | Es ist ein Select-Statement geschrieben, welches angibt, wie viele Matches einer Liga zugewiesen sind. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | PCH | Optional: | N |
| Art: | SELECT | *Name:* | Trainer-Spieler | | |
| Aufgabe: | Es ist ein Select-Statement geschrieben, dass alle Trainer auflistet und die Spieler, die er trainiert. Es ist nach den Trainern gruppiert. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | SEI | Optional: | N |
| Art: | SELECT | *Name:* | Tore-Durchschnitt | | |
| Aufgabe: | Es ist ein Select-Statement geschrieben, welches den durchschnittlichen Tore-Stand aller Mannschaften ausgibt. Es soll nach den besten Mannschaften absteigend geordnet werden. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | CAR | Optional: | N |
| Art: | SELECT | *Name:* | Verein-Daten | | |
| Aufgabe: | Es werden alle Vereine mit ihrer Anschrift, der Anzahl der Mannschaften, der Anzahl der Mitarbeiter, der Anzahl der Spieler und der Anzahl der Trainer ausgegeben werden. Es sollen auch berücksichtigt werden, Personen, die ausgetreten sind, nicht angezeigt werden. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

# DB-Users

Password für alle erstellten User soll sein: ganz\_geheim!

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | PCH | Optional: | N |
| Art: | User | *Name:* | User\_op | | |
| Aufgabe: | Es ist ein User erstellt, der alle Rechte in der FBV-DB hat, allerdings keine Rechte in anderen Schemata. Er darf keine User / Rollen erstellen oder vergeben. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | HÖL | Optional: | J |
| Art: | User | *Name:* | User\_r | | |
| Aufgabe: | Es ist ein User erstellt, der keine Create Rechte hat und keine Daten hinzufügen darf (Unabhängig der Methode). Er darf lediglich Daten über vordefinierte Views oder SELECTS auslesen. In beiden Fällen ist der User auf das FBV-Schema limitiert. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

# VIEWS

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | PCH | Optional: | N |
| Art: | VIEW | *Name:* | list\_matches | | |
| Aufgabe: | Es werden alle Matches ausgegeben, die in einer Liga geplant oder gespielt sind / wurden. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | HÖL | Optional: | N |
| Art: | VIEW | *Name:* | List\_person | | |
| Aufgabe: | Es werden alle Personen ausgegeben, mit der aktiven Rolle, als auch mit den früheren Rollen. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | CAB | Optional: | N |
| Art: | VIEW | *Name:* | Show\_ranking | | |
| Aufgabe: | Es werden alle Mannschaften eines Vereins angezeigt. Die Mannschaften werden alphabethisch nach dem Namen des Vereins und der Mannschaft geordnet. (Die Mannschaften sind nicht nach dem Verein gruppiert, der Verein wird einfach mehrfach ausgegeben). | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | SEI | Optional: | N |
| Art: | VIEW | *Name:* | Ausgabe einer Person | | |
| Aufgabe: | Es wird ein Ranking eines Matches ausgegeben. Es zeigt an, welche Mannschaft im Moment am besten ist die meisten Punkte hat. Die Punkte werden in dieser View als zusätzliche Spalte ausgegeben und nach den gewonnen Spielen und dem Tore-Stand berechnet.  Diese View soll so gestaltet sein, dass eine neue Mannschaft einem Verein hinzugefügt werden kann. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

Procedures & Functions

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | HÖL | Optional: | N |
| Art: | Function | *Name:* | getPersonID | | |
| Aufgabe: | Es wird die PersonID einer Person zurückgegeben.  Input Parameter sind Vorname und Nachname.  Folgende Return Codes sind definiert:  ret = -3 … Falsche Eingabe (Z.B. falsche Anzahl an Parameter)  ret = -2 … Mehrere Personen gefunden  ret = -1 ... Person nicht gefunden  ret >= 0 … PersonID | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | PCH | Optional: | N |
| Art: | Function | *Name:* | getVereinID | | |
| Aufgabe: | Es wird die VereinID einer Person zurückgegeben.  Input Parameter ist die PersonenID oder Vorname und Nachname (TIPP: Function-Overloading).  Wenn Vorname und Nachname eingegeben sind wird die PersonenID mit getPersonID() abgefragt.  Folgende Return Codes sind definiert:  ret = -4X … Mehrere Vereine gefunden. Person ist aktuell bei VereinID = X.  ret = -3 … Falsche Eingabe (Z.B. falsche Anzahl an Parameter)  ret = -2 … Mehrere Personen gefunden  ret = -1 ... Person nicht gefunden  ret >= 0 … VereinID. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | CAB | Optional: | N |
| Art: | Function | *Name:* | personIsA | | |
| Aufgabe: | Es wird zurückgegeben, als was eine Person gespeichert ist.  Input Parameter ist die PersonenID oder Vorname und Nachname.  Optionaler Input Parameter ist StepsBack. Dieser InputParameter hat als Default Wert 0. 0 gibt die aktuelle Art der Person aus. 1 gibt die Art einer Person aus, der sie zuvor zugwiesen war. 2 geht zwei Schritte zurück. 3 geht ….  (TIPP: Function-Overloading und defaultParameter).  Wenn Vorname und Nachname eingegeben sind wird die PersonenID mit getPersonID() abgefragt.  Folgende Return Codes sind definiert:  ret = -3 … Falsche Eingabe (Z.B. falsche Anzahl an Parameter)  ret = -2 … Mehrere Personen gefunden  ret = -1 ... Person nicht gefunden  ret = 0 … Die Person wird in keiner Untergruppe gefunden.  Ret = X … Die Person ist ein (X=1; Mitarbeiter / X=2; Spieler /  X=3; Trainer / X=4; Schiedsrichter) | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | SEI | Optional: | N |
| Art: | Function | *Name:* | personHistory | | |
| Aufgabe: | Es wird zurückgegeben, ob die Person die Art gewechselt hat. Also ob die Person als Spieler in einem Verein war und jetzt als Trainer arbeitet.  Input Parameter ist die PersonenID / Vorname und Nachname / VereinID und PersonID / VereinID und Vorname und Nachname (TIPP: Function-Overloading).  Wenn Vorname und Nachname eingegeben sind wird die PersonenID mit getPersonID() abgefragt. Die VereinID kann mit der PersonenID angegeben werden  Folgende Return Codes sind definiert:  ret = -5 … Die Person war bei mehr als 2 Vereinen.  Ret = -4X00Z00Y …  X = Aktueller Verein  Z = Vorheriger Verein  Y = Code gleicht dem Ergebniss ret >= 0  ret = -3 … Falsche Eingabe (Z.B. falsche Anzahl an Parameter)  ret = -2 … Mehrere Personen gefunden  ret = -1 ... Die Person wird in keiner Untergruppe gefunden.  ret = 0 … Die Person ist keiner Untergruppe zugeordnet.  ret = A[B][C][..] … Die Person ist A, war zuvor B, war vor B C, … ABC usw. sind nach dem gleichen Prinzip aufgebaut wie (ret => 1) aus der Funktion personIsA. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | CAB | Optional: | N |
| Art: | Procedure | *Name:* | addPerson | | |
| Aufgabe: | In dieser Procedure wird eine Person hinzugefügt. Es wird mit der Procedure sichergestellt, dass die Person auch einer Untergruppe zugewiesen ist (Mitarbeiter, Spieler, Trainer oder Schiedsrichter).  Ist die Person ein Mitarbeiter, wird sie gleich einem Verein zugewiesen.  Ist die Person ein Spieler oder ein Trainer wird sie gleich einer Mannschaft zugewiesen.  Die ID der Person wird automatisch erstellt.  Input Parameter sind: vname, nname, gebDat, geschlecht, eintrittsdatum  Zusätzlich Inputparameter: isA, funktionStatus, VereinMannschaftID  Optionaler Parameter ist (Tipp: Procedure Overloading): anschriftID.  Vname,nname,gebDat und Geschlecht dürfen nicht null sein.  Ist das Eintritts Datum null, wird es automatisch das aktuelle Datum genommen.  isA darf nicht null sein und ist nach dem gleichen Prinzip aufgebaut wie (ret => 1) aus der Funktion personIsA.  funktionStatus darf null sein, wenn isA = 3/4.  VereinMannschaftID darf null sein, wenn isA = 4.  anschriftID darf null sein oder weggelassen werden. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | RCH | Optional: | N |
| Art: | Procedure | *Name:* | addLiga | | |
| Aufgabe: | In dieser Procedure wird eine Liga hinzugefügt. Es wird mit der Procedure sichergestellt, dass die Liga auch mindestens aus 5 Matches besteht.  Die ID der Liga wird automatisch erstellt.  Input Parameter sind: name, beginn, ende, altersgruppe, geschlecht, t\_matches  Kein Input Parameter darf null sein.  Der input Parameter t\_matches ist vom Type ownArr.  Der Type ownArr kann mehrere Number Datentypen beinhalten und ist loopable. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | SEI | Optional: | N |
| Art: | Procedure | *Name:* | addMannschaft | | |
| Aufgabe: | In dieser Procedure wird eine Mannschaft hinzugefügt. Es wird mit der Procedure sichergestellt, dass die Mannschaft auch einem Verein zugewiesen ist und mind. Einen Trainer hat.  Ist die Person ein Mitarbeiter, wird sie gleich einem Verein zugewiesen.  Die ID der Mannschaft wird automatisch erstellt.  Input Parameter sind: name, altersgruppe, t\_trainer  Kein Input Parameter darf null sein.  Der input Parameter t\_matches ist vom Type ownArr.  Der Type ownArr kann mehrere Number Datentypen beinhalten und ist loopable. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | HÖL | Optional: | N |
| Art: | Procedure | *Name:* | addMatch | | |
| Aufgabe: | In dieser Procedure wird ein Match hinzugefügt. Es wird mit der Procedure sichergestellt, dass das Match auch mind. 1 Schiedrichter zugewiesen hat und das genau zwei Mannschaften bestimmt sind.  Die ID des Matches wird automatisch erstellt.  Input Parameter sind: datum, begonnen, geendet, heim, gast, t\_schiedsrichter  Kein Input Parameter darf null sein.  Der input Parameter t\_schiedsrichter ist vom Type ownArr.  Der Type ownArr kann mehrere Number Datentypen beinhalten und ist loopable. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | SEI | Optional: | N |
| Art: | TYPE | *Name:* | ownArr | | |
| Aufgabe: | Eine Instanz dieser Type kann mehrere Number-Datensätze speichern und ausgeben. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

# Package

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | CAB | Optional: | N |
| Art: | PACKAGE | *Name:* | fbv | | |
| Aufgabe: | Dieses Package hat folgende public Members:  AddMatch - Procedure  addMannschaft - Procedure  addLiga - Procedure  addPerson - Procedure  personHistory - Function  personIsA - Function  getVereinID - Function  getPersonID - Function  Und folgenden private Members:  ownArr - Type  getNextId – optional Function | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

# Triggers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | PCH | Optional: | N |
| Art: | TRIGGER | *Name:* | - | | |
| Aufgabe: | Nach dem Löschen einer Liga sollen auch alle Matches, die in dieser Liga stattgefunden haben, gelöscht werden. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | SEI | Optional: | N |
| Art: | TRIGGER | *Name:* | - | | |
| Aufgabe: | Wird eine Person hinzugefügt soll kontrolliert werden, ob diese Person bereits in einer anderen Rolle oder in einem anderen Verein tätig ist. Wenn dem so ist soll diese Aktion wieder Rückgängig gemacht werden. (Es soll gecheckt werden, ob das Austrittsdatum bereits eingetragen ist) | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | HÖL | Optional: | N |
| Art: | TRIGGER | *Name:* | - | | |
| Aufgabe: | Wird eine Person gelöscht soll diese Aktion rückgängig gemacht werden. Es wird nur das Austritts Datum der Person gesetzt.  (Dieser Trigger soll nicht für den User: User\_r gefeuert werden. Dieser Unterpunkt ist optional, da ich nicht weiß, ob er möglich ist) | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Status: | BEREIT | *Zugewiesen an:* | CAB | Optional: | N |
| Art: | TRIGGER | *Name:* | - | | |
| Aufgabe: | Wird irgendeine Aktion getätigt sollen folgende Punkte in einer log-Tabelle gespeichert werden:   * User * Aktion * Datum * Uhrzeit   Die Log-Tabelle muss dafür erstellt werden. Fügen sie auch das Drop-Statement hinzu. | | | | |
|  |
| Code: |  | | | | |
|  |
| Ausführung: |  | | | | |
|  |
| Resultat (Z.B. Screenshot): |  | | | | |