Datenbankpraktikum

Miniwelt des LWB-Adventures

Dozent: Sebastian Herker
Abgebende: Annalena Cyriacus, Benjamin Schneider, Philipp Liehm, Martin Seiß

1 Modellierung

1.1 Beschreibung der Miniwelt

Die Miniwelt beschreibt die Spielwelt des im Rahmen des Software-Praktikums selbst programmierten Spiels LWBadventure. Dieses Spiel ist an die Lehrerweiterbildung Informatik Berlin angelehnt. Die SpielerInnen absolvieren die vier Semester des Studiums, indem sie kleine Mini-Games spielen, welche die Lehrinhalte der jeweiligen Semester thematisieren und abfragen.

Die Welt beinhaltet Non-Playing-Characters (NPCs), die eine Namen (Alias) haben und sich in DozentInnen und sonstige NPCs unterteilen. Die DozentInnen haben ein Lieblingsgetränk, die NPCs erfüllen eine bestimmte Aufgabe im Spiel.

Das Spiel hat verschiedene Räume an verschieden Standorten (FU Berlin, STEPS, Home Office). In den Räumen 1 bis 4 finden die Veranstaltungen statt (Kursräume), die anderen Räume haben eine andere Funktion (Start- und Schluss-Raum).

Sonstige NPCs können sich (im Gegensatz zur realen Welt) in allen Räumen aufhalten – sogar gleichzeitig in mehreren.

Die Veranstaltungen werden von einem Dozenten in einem Kursraum gehalten. Dabei können sie von einem anderen Dozenten unterstützt werden (Assistenz). Veranstaltungen haben einen Titel, ein Abkürzung (Kürzel), eine Semesterwochenstundenzahl, ein Semester und können einem Themengebiet zugeordnet werden.

Zu den Veranstaltungen kann es Minispiele (oder Minigames) geben, welche Prüfungen zu den Veranstaltungen repräsentieren, es muss aber nicht zu allen Veranstaltungen ein Minigame geben. Die SpielerInnen können verschiedene Räume betreten und dabei die Minispiele spielen. Jede/r SpielerIn erhält nach Beenden des Spiels eine Punktzahl und eine Note.

Die SpielerInnen haben am Anfang des Spiels einen Schlüssel für den Kursraum 1 und können diesen betreten. Im Laufe des Spiels können sie sich weitere Schlüssel erspielen, um die anderen Räume zu betreten. Dazu müssen Sie in den jeweiligen vorherigen Semestern die Spiele mindestens mit der Note 4.0 beenden.

1.2 Integritätsbedingungen

1. Die Schlüsselanzahl wird von den erspielten Noten abgeleitet. Die Schlüsselanzahl entspricht dem Semester, in dem sie sich befinden. Es können nur Spiele gespielt werden, deren Semesternummer der Schlüsselanzahl entspricht bzw. unterbietet. Um die Schlüsselanzahl zu erhöhen müssen die SpielerInnen in den jeweiligen vorherigen Semestern die Spiele mit mindestens Note 4.0 beenden.

2. Die Kursraumnummern und Semester stimmen überein.

3. In den Räumen 1 bis 4 finden die Veranstaltungen statt (Kursräume), und die anderen Räume haben eine andere Funktion.

4. Es gibt maximal 4 Semester.

5. Die Kürzel der Veranstaltungen dürfen nicht doppelt/mehrfach vergeben werden.

1.3 Relationenmodell

1. NPCs(NPCNr, NPCName)
2. Dozent_Innen (NPCNr ↑, Lieblingsgetränk)

1.4 Beschreibung des Transformationsweges

1.5 Funktionale Abhängigkeiten

1.6 Beispieldaten

1.6.1 NPCs

NPCNr	*NPCName*
1	Darth Schmidter
2	Winnie the K
3	Fab Web
4	Amoebi
5	J.EthI
6	Herk
7	Heidi
8	Palim Palim
9	Hubi-Horde

1.6.2 s

onstige NPCs

NPCs

npcnr	*npcname*
1	Darth Schmidter
2	Winnie the K
3	Fab Web
4	Amoebi
5	J.EthI
6	Herk
7	Heidi
8	Palim Palim
9	Hubi-Horde

(9 Zeilen)

sonstige NPCs

npcnr	*aufgabe*
7	StEPS-Chefin
8	Helferlein
9	Kontrolletis

(3 Zeilen)

DozentInnen

npcnr	*lieblingsgetraenk*
1	Extraschwarzer Kaffee
2	Bier
3	Cappuccino
4	Grüner Tee
5	Kaffee mit Milch und 2x Zucker
6	Hefeweizen

(6 Zeilen)

Unterricht

vnr	*npcnr*	*raumnr*
1	2	1
2	3	1
3	2	1
4	1	2
5	2	2
6	5	2
7	1	3
8	6	3
9	2	3
10	1	4
11	2	4
12	4	4
13	6	4
14	3	4

(14 Zeilen)

Assistenz

vnr	*npcnr*
1	1
2	5
3	1
4	4
10	2
12	3
14	4

(7 Zeilen)

Veranstaltungen

vnr	*vname*	*kuerzel*	*sws*	*semester*	*gebietnr*
1	Betriebssystemwerkzeuge	BSW	2	1	1
2	Funktionale Programmierung	FP	8	1	2
3	Grundlagen der Technischen Informatik	RS	6	1	3
4	Imperative und objektorientierte Programmierung	ALP2	7	2	2
5	Rechnerarchitektur	RO	4	2	1
6	Einführung in die Theoretischen Informatik	EthI	5	2	3
7	Datenstrukturen und Datenabstraktion	ALP3	6	3	2
8	Datenbanksysteme	DBSA	6	3	4
9	Fachdidaktik Informatik	DDI	4	3	5
10	Nichtsequentielle und verteilte Programmierung	NSP	6	4	2
11	Rechnernetze	NET	6	4	1
12	Unterrichtsbezogenes Softwarepraktikum	SWP	3	4	2
13	Unterrichtsbezogenes Datenbankpraktikum	DBP	3	4	4
14	Analyse fachlichen Lernens	AfL	3	4	5

(14 Zeilen)

Themengebiete

gebietnr	*gebietname*
1	Rechnerarchitektur, Betriebs- und Kommunikationssysteme
2	Programmierung
3	Theoretische und technische Informatik
4	Datenbanken
5	Didaktik

(5 Zeilen)

Minigames

gamenr	*gamename*	*vnr*
1	Muster-Spiel	2
2	Bauelemente-Spiel	3
3	Vaderobi-Game	4
4	Getränkeautomaten-Spiel	6
5	SQL-Quest	8
6	FachJargon	9
7	Food-Moorhuhn	10
8	theNETgame	11
9	BugAttack	12

(9 Zeilen)

SpielerInnen

spnr	*spname*	*schuesselanzahl*	*raumnr*
1	Cyra	1	0
2	Maddi	1	0
3	Ben	1	0
4	Phil	1	0
5	Klocki	1	0
6	Bob	1	0
7	LWB-Master	1	0
8	Nerd42	1	0

(8 Zeilen)

Räume

raumnr	*raumname*	*ort*	*funktion*
0	Main Floor	LWB-World	Start-Raum
1	1. Semester	FU Berlin	Kursraum 1
2	2. Semester	Home Office	Kursraum 2
3	3. Semester	FU Berlin	Kursraum 3
4	4. Semester	StEPS	Kursraum 4
5	Nichtzeugnis-Verleihung	schöner Ort	Schluss-Raum

(6 Zeilen)

Spielstände

gamenr	*spnr*	*note*	*punkte*
1	1	1.7	1333
1	2	1.3	1456
1	3	1.0	1800
1	4	2.3	999
1	5	2.0	1210
1	6	4.0	400
1	7	1.0	2000
1	8	3.0	789
2	1	1.3	29
2	2	1.0	33
2	3	1.3	29
2	4	1.7	27
2	5	2.0	25
2	6	3.7	17
2	7	1.0	33
2	8	2.7	22
3	1	1.0	630
3	2	2.3	432
3	3	1.3	555
3	4	1.7	512
3	5	2.0	487
3	6	3.3	333
3	7	1.0	650
3	8	1.0	600
4	1	1.3	5
4	2	1.0	6
4	3	1.7	4
4	4	1.3	5
4	5	2.0	3
4	6	3.0	2
4	7	1.0	6
5	1	1.0	100
5	2	1.7	88
5	3	1.3	96
5	4	2.0	79
5	6	6.0	44
5	7	1.0	100
5	8	2.3	73
6	1	2.3	71
6	2	2.0	76
6	3	1.7	83
6	4	1.0	99
6	7	1.0	100
7	1	1.7	456
7	2	2.3	369
7	3	1.0	555
7	4	1.3	512
7	7	1.0	600
8	1	1.3	140
8	2	1.0	150
8	3	1.7	128
8	4	1.3	128

(58 Zeilen)

Aufenthaltsorte

npcnr	*raumnr*
7	0
7	4
7	5
8	0
9	0
9	4

(6 Zeilen)

2 Relationenalgebra mit DES

Aufgaben-Sequenz zur LWBadventure-Datenbank

RELATIONEN

NPCs (**$ NPCNr**, NPCName) *-- NPC: Non Playing Character*

Dozent_innen (**! NPCNr**, Lieblingsgetränk)

sonstigeNPCs (**! NPCNr**, Aufgabe)

Veranstaltungen (**$ VNr**, VName, Kürzel, SWS, Semester, **! GebietNr**)

Themengebiete (**$ GebietNr**, GebietName)

Minigames (**$ GameNr**, GameName, **! VNr**)

Spieler_innen (**$ SpNr**, SpName, Schlüsselanzahl, **! RaumNr**)

Räume (**$ RaumNr**, Raumname, Ort, Funktion)

Unterricht (**!$ VNr**, **! NPCNr**, **! RaumNr**)

Spielstände (**!$ GameNr**, **!$ SpNr**, Note, Punkte)

Aufenthaltsorte (**!$ NPCNr**, **!$ RaumNr**)

Assistenz (**!$ VNr**, **! NPCNr**)

ANFRAGEN

1a. Welche Räume gibt es in der LWB-Adventure-World?

Raeume

```
/listing Raeume
```

```
SELECT * FROM raeume;
```

1b. Welche Aufgaben haben die sonstigen NPCs im LWB-Adventure?

$\pi_{\text{Aufgabe}}(\text{sonstigeNPCs})$

```
project Aufgabe (sonstigeNPCs);
```

```
SELECT aufgabe FROM sonstigenpcs;
```

2a. Welche Lehrveranstaltungen haben 6 SWS?

$\sigma_{\text{SWS} = 6}(\text{Veranstaltungen})$

```
select SWS = 6 (Veranstaltungen);
```

```
SELECT * FROM veranstaltungen WHERE sws = 6;
```

2b. Welche Lehrveranstaltungen gibt es im 4. Semester?

$\sigma_{\text{Semester} = 4}(\text{Veranstaltungen})$

```
select Semester = 4 (Veranstaltungen);
```

```
SELECT * FROM veranstaltungen WHERE semester = 4;
```

Cyriacus

2c. Welche Minigames gibt es im 4. Semester?

$\sigma_{\text{Semester = 4}}(\text{Minigames} \bowtie \text{Veranstaltungen})$

```
select Semester = 4 (Minigames njoin Veranstaltungen);
```

```
SELECT * FROM minigames NATURAL JOIN veranstaltungen WHERE semester = 4;
```

3a. Wie heißen die Spieler_innen, die bisher das LWB-Adventure gespielt haben?

$\pi_{\text{SpName}}(\text{Spieler\_innen})$

```
project SpName (Spieler_innen);
```

```
SELECT spname FROM spieler_innen;
```

3b. Wie heißen die Dozenten im LWB-Adventure?

$\pi_{\text{NPCName}}(\text{Dozent\_innen} \bowtie \text{NPCs})$

```
project NPCName (Dozent_innen njoin NPCs);
```

```
SELECT npcname FROM dozent_innen NATURAL JOIN npcs;
```

3c. Welche Aufgabe hat NPC 'Heidi'?

$\pi_{\text{Aufgabe}}(\sigma_{\text{NPCName = 'Heidi'}}(\text{sonstigeNPCs} \bowtie \text{NPCs}))$

```
project Aufgabe (select NPCName = 'Heidi' (sonstigeNPCs njoin NPCs));
```

```
SELECT aufgabe FROM sonstigenpcs NATURAL JOIN npcs WHERE npcname = 'Heidi';
```

4a. Welche Lehrveranstaltungen gehören zum Themengebiet 'Programmierung'?

$\sigma_{\text{GebietName = 'Programmierung'}}(\text{Veranstaltungen} \bowtie \text{Themengebiete})$

```
select GebietName = 'Programmierung' (Veranstaltungen njoin Themengebiete);
```

```
SELECT * FROM veranstaltungen NATURAL JOIN themengebiete WHERE gebietname =
'Programmierung';
```

4b. Welche Lehrveranstaltungen haben etwas mit 'Daten' oder 'Programmierung' zu tun?

```
% Wildcards in RA nicht möglich!
```

```
SELECT * FROM veranstaltungen WHERE vname LIKE '%Daten%' OR vname LIKE
'%Programmierung%';
```

oder

```
SELECT * FROM veranstaltungen NATRUAL JOIN themengebiete WHERE gebietname LIKE '%Daten%'
OR gebietname LIKE '%Programmierung%';
```

5. Was ist das Lieblingsgetränk von Darth Schmidter?

$\pi_{\text{Lieblingsgetränk}}(\sigma_{\text{NPCName = 'Darth Schmidter'}}(\text{Dozent\_innen} \bowtie \text{NPCs}))$

Cyriacus

```
project Lieblingsgetraenk (select NPCName = 'Darth Schmidter' (Dozent_innen njoin
NPCs));
```

```
SELECT lieblingsgetraenk FROM dozent_innen NATURAL JOIN npcs WHERE npcname = 'Darth
Schmidter';
```

Cyriacus

6. Welche Lehrveranstaltungen finden nicht in der 'FU Berlin' statt?

$\pi_{VName,Ort}((\sigma_{Ort\ !=\ 'FU\ Berlin'}(Räume)) \bowtie Unterricht \bowtie Veranstaltungen)$

```
project VName, Ort ((select Ort != 'FU Berlin' (Raeume)) njoin Unterricht njoin
Veranstaltungen);
```

oder mit Differenz:

$\pi_{VName,Ort}((Räume \setminus \sigma_{Ort\ =\ 'FU\ Berlin'}(Räume)) \bowtie Unterricht \bowtie Veranstaltungen)$

```
project VName, Ort ((Raeume difference (select Ort = 'FU Berlin' (Raeume))) njoin
Unterricht njoin Veranstaltungen);
```

```
SELECT vname, semester, ort FROM raeume NATURAL JOIN unterricht NATURAL JOIN
veranstaltungen WHERE ort != 'FU Berlin';
```

7. Welche Dozenten sind in der LWB nur leitend tätig und machen keine Assistenz?

$\pi_{NPCName}(NPCs \bowtie (\pi_{NPCNr}(Dozent\_innen) \setminus \pi_{NPCNr}(Assistenz)))$

```
project NPCName (NPCs njoin (project NPCNr (Dozent_innen) difference project NPCNr
(Assistenz)));
```

```
SELECT npcname FROM npcs NATURAL JOIN (SELECT npcnr FROM dozent_innen EXCEPT SELECT
npcnr FROM assistenz) AS xyz;
```

Kommentar: Hier braucht es einen Alias, damit der NATURAL JOIN mit der Unterabfrage funktioniert. Die Bezeichnung ist jedoch egal, da nur auf den NPCNamen projiziert wird.

Anfragen, die nur mit erweiterter relationaler Algebra beschrieben werden können:

8. Wieviele Mini-Games gibt es in der LWB-Adventure-World? (Ausgaben-Titel: Anzahl.Minigames)

$Anzahl.Minigames \leftarrow \gamma_{COUNT(*)}(Minigames)$

```
rename Anzahl (Minigames) (group_by [] count(*) true (Minigames));
```

```
SELECT COUNT(*) AS Anzahl.Minigames FROM minigames;
```

9. Wieviele SWS müssen in der LWB ingesamt absolviert werden?
(Ausgaben-Titel: Gesamtanzahl.SWS)

$Gesamtanzahl.SWS \leftarrow \gamma_{SUM(SWS)}(Veranstaltungen)$

```
rename Gesamtanzahl (SWS) (group_by [] sum(SWS) true (Veranstaltungen));
```

```
SELECT SUM(sws) AS Gesamtanzahl.SWS FROM veranstaltungen;
```

10. Wie heißt die Veranstaltung mit den meisten SWS?

$\pi_{Vname}(select\ SWS = MaxAnzahl.SWS\ ((MaxAnzahl.SWS \leftarrow \gamma_{MAX(sws)}(Veranstaltungen)) \bowtie Veranstaltungen))$

```
project VName (select SWS = MaxAnzahl.SWS (rename MaxAnzahl (SWS) (group_by []
max(SWS) true (Veranstaltungen)) njoin Veranstaltungen));
```

```
SELECT vname FROM veranstaltungen WHERE sws = (SELECT MAX(sws) FROM veranstaltungen);
```

Cyriacus

11. Gesucht sind die Namen, Semester und SWS aller Veranstaltungen von Winnie the K absteigend sortiert nach SWS-Anzahl!

$T_{SWS\downarrow}(\pi_{VName,SWS,Semester}(\sigma_{NPCName = 'Winnie\ the\ K'}(Veranstaltungen \bowtie Unterricht \bowtie NPCs)))$

$\tau_{SWS\downarrow}(\pi_{VName,SWS,Semester}(\rho_{Semester \leftarrow Raumnr}(\sigma_{NPCName = 'Winnie\ the\ K'}(Veranstaltungen \bowtie Unterricht \bowtie NPCs))))$

```
sort SWS desc (project VName, SWS, Semester (select NPCName = 'Winnie the K'
(Veranstaltungen njoin Unterricht njoin NPCs)));
```

```
SELECT vname, sws, semester FROM veranstaltungen NATURAL JOIN unterricht NATURAL JOIN npcs
WHERE npcname = 'Winnie the K' ORDER BY sws DESC;
```

12. Wieviele Veranstaltungen gibt es pro Standort?

$\tau_{AnzahlVeranstaltungen}(\gamma_{Ort,\ AnzahlVeranstaltungen \leftarrow COUNT(*)}(Räume \bowtie Unterricht))$

```
sort AnzahlVeranstaltungen (rename Standorte (OrtName, AnzahlVeranstaltungen)
(group_by Ort Ort,count(VNr) true (Raeume njoin Unterricht)));
```

```
SELECT ort, COUNT(*) AS AnzahlVeranstaltungen FROM raeume NATRUAL JOIN unterricht GROUP BY
ort ORDER BY COUNT(*);
```

13. Welche Spieler_innen haben einen Gesamt-Notendurchschnitt, der nicht zwischen 2.0 und 4.0 liegt? (Sortierung nach Gesamt-Notendurchschnitt aufsteigend, also bester Schnitt zuerst)

$Q1 \leftarrow \gamma_{SpName,\ AVG(Note)}(Spielstände \bowtie Spieler\_innen)$

$Q2 \leftarrow \rho_{Schnitt \leftarrow AVG(Note)}(Q1)$

$Q3 \leftarrow \sigma_{Schnitt<2.0\ \wedge\ Schnitt>4.0}(Q2)$

$Q4 \leftarrow \tau_{Schnitt,SpName}(Q3)$

$Q5 \leftarrow \pi_{SpName}(Q4)$

oder

$\pi_{SpName}(\tau_{Schnitt,SpName}(\sigma_{Schnitt<2.0\ \wedge\ Schnitt>4.0}(\rho_{Schnitt \leftarrow AVG(Note)}(\gamma_{SpName,\ AVG(Note)}(Spielstände \bowtie Spieler\_innen)))))$

```
project SpName (sort Schnitt,SpName (select Schnitt < 2.0 or Schnitt > 4.0 (rename
Noten (SpName,Schnitt) (group_by SpName SpName,avg(Note) true (Spielstaende njoin
Spieler_innen)))));
```

```
SELECT SpName FROM spieler_innen NATURAL JOIN spielstaende GROUP BY spname HAVING
AVG(note) NOT BETWEEN 2.0 AND 4.0 ORDER BY AVG(note),spname;
```

Kommentar: Hier könnte die Projektion auch weggelassen werden, wenn die Ausgabe außer den SpNamen auch den Notendurchschnitt enthalten soll.

$\tau_{Schnitt,SpName}(\sigma_{Schnitt<2.0\ \wedge\ Schnitt>4.0}(\rho_{Schnitt \leftarrow AVG(Note)}(\gamma_{SpName,\ AVG(Note)}(Spielstände \bowtie Spieler\_innen))))$

```
sort Schnitt,SpName (select Schnitt < 2.0 or Schnitt > 4.0 (rename Noten
(SpName,Schnitt) (group_by SpName SpName,avg(Note) true (Spielstaende njoin
Spieler_innen))));
```

```
SELECT SpName, ROUND(AVG(note),2) AS Notendurchschnitt FROM spieler_innen NATURAL JOIN
spielstaende GROUP BY spname HAVING AVG(note) NOT BETWEEN 2.0 AND 4.0 ORDER BY
AVG(note),spname;
```

Cyriacus

3 Implementierung im Datenbankystem (PostgreSQL)

3.1 Create-Datei

Listing 1: Create-LWBadventure.sql

```
--  Miniwelt: LWBadventure
--  Umsetzung des Datenbankpraktikums der LWB Informatik
--
--  Annalena Cyriacus, Benjamin Schneider, Philipp Liehm, Martin Seiss
--
--  Start: 13.06.2023


-- psql-Befehle:
--              \i Create-LWBadventure.sql       -- zum Erzeugen der Tabellen
--              \i Drop-LWBadventure.sql         -- zum Loeschen der Tabellen
--              \i Insert-LWBadventure-data.sql -- zum Einfuegen der Daten
--              \dt             -       alle Tabelle anzeigen
--              drop table dozentin;    - Loescht Tabelle dozentIn


-- Folgende Tabellen werden implementiert:
-- $ = Schluessel; ! = Fremdschluessel

-- npcs ($npcNr, npcName)                                   NPCs - Non-Playing-Character
-- dozent_innen (!npcNr, lieblingsgetraenk)
-- sonstigeNPCs (!npcNr, aufgabe)
-- raeume ($raumNr, raumName, ort, funktion)
-- themengebiete ($gebietNr, gebietName)
-- veranstaltungen ($vNr, vName, kuerzel, sws, !gebietNr)
-- spieler_innen ($spNr, spName, schluesselanzahl, !raumNr)
-- minigames ($gameNr, gameName, !vNr)
-- spielstaende (!gameNr, !spNr, Note, Punktzahl)
-- aufenthaltsorte (!npcNr, !raumNr)
-- unterricht (!vNr, !npcNr, !raumNr)
-- assistenz (!vNr,!npcNr)



-- npcs ($npcNr, npcName)                                   NPCs - Non-Playing-Character
CREATE TABLE npcs (
  npcNr                 INTEGER                         NOT NULL,
  npcName               VARCHAR (50)            NOT NULL,
  CONSTRAINT npcKEY PRIMARY KEY (npcNr)
);
COMMENT ON Table npcs IS 'Miniwelt LWBadventure';


-- dozent_innen (!npcNr, lieblingsgetraenk)
CREATE TABLE dozent_innen (
  npcNr                                 INTEGER                                 REFERENCES npcs (npcNr),        -- N
  lieblingsgetraenk             VARCHAR (100)           NOT NULL,                                               --
  CONSTRAINT dozent_innenKEY PRIMARY KEY (npcNr)
);
COMMENT ON Table dozent_innen IS 'Miniwelt LWBadventure';


-- sonstigeNPCs (!npcNr, aufgabe)
CREATE TABLE sonstigeNPCs (
  npcNr                 INTEGER                         REFERENCES npcs (npcNr),                        -- N
  aufgabe               VARCHAR (100)           NOT NULL,
  CONSTRAINT sonstigeNPCsKEY PRIMARY KEY (npcNr)
);
COMMENT ON Table sonstigeNPCs IS 'Miniwelt LWBadventure';
```

```
-- raeume ($raumNr, raumName, ort, funktion)
CREATE TABLE raeume (
  raumNr                INTEGER                         NOT NULL,
  raumName              VARCHAR (50)            NOT NULL,
  ort                   VARCHAR (50)            NOT NULL,
  funktion              VARCHAR (50)            NOT NULL,
  CONSTRAINT raumKEY PRIMARY KEY (raumNr)
);
COMMENT ON Table raeume IS 'Miniwelt LWBadventure';


-- themengebiete ($gebietNr, gebietName)
CREATE TABLE themengebiete (
  gebietNr              INTEGER                         NOT NULL,
  gebietName    VARCHAR (100)           NOT NULL,
  CONSTRAINT themengebieteKEY PRIMARY KEY (gebietNr)
);
COMMENT ON Table themengebiete IS 'Miniwelt LWBadventure';


-- veranstaltungen ($vNr, vName, kuerzel, sws, !gebietNr)
CREATE TABLE veranstaltungen (
  vNr                   INTEGER                         NOT NULL,       -- Veranstaltungsnummer
  vName                 VARCHAR (50)            NOT NULL,
  kuerzel               VARCHAR (5)                     NOT NULL,
  sws                   INTEGER                         CHECK (sws > 0),
  semester              INTEGER                         CHECK (semester > 0),
  gebietNr              INTEGER                         REFERENCES themengebiete (gebietNr),
  CONSTRAINT veranstaltungenKEY PRIMARY KEY (vNr)
);
COMMENT ON Table veranstaltungen IS 'Miniwelt LWBadventure';


-- spieler_innen ($spNr, spName, schluesselanzahl, !raumNr)
CREATE TABLE spieler_innen (
  spNr                          INTEGER                         NOT NULL,       --CHECK (MatrNr BETWEEN 1000
  spName                        VARCHAR (50)            NOT NULL,
  schuesselanzahl       INTEGER                         NOT NULL,
  raumNr                        INTEGER                         REFERENCES raeume (raumNr) DEFAULT 1,
  CONSTRAINT spieler_innenKEY PRIMARY KEY (spNr)
);
COMMENT ON Table spieler_innen IS 'Miniwelt LWBadventure';


-- minigames ($gameNr, gameName, !vNr)
CREATE TABLE minigames(
  gameNr                INTEGER                         NOT NULL,       -- oder CHECK (vNr > 0),
  gameName              VARCHAR (50)            NOT NULL,
  vNr                   INTEGER                         REFERENCES veranstaltungen (vNr),       -- vNr = Ve
  CONSTRAINT minigameKEY PRIMARY KEY (gameNr)
);
COMMENT ON Table minigames IS 'Miniwelt LWBadventure';


-- Nur spezielle Notenformate
CREATE DOMAIN NOTEN
        AS NUMERIC (2,1)
        DEFAULT 6.0
        CHECK (VALUE IN (1.0,1.3,1.7,
                         2.0,2.3,2.7,
                         3.0,3.3,3.7,
                         4.0,4.3,4.7,
                         5.0,5.3,5.7,
                         6.0));


-- spielstaende (!gameNr, !spNr, Note, Punktzahl)
CREATE TABLE spielstaende (
```

```
  gameNr                INTEGER                 REFERENCES minigames (gameNr),
  spNr                  INTEGER                 REFERENCES spieler_innen (spNr),
  note                  NOTEN                   ,-- nur spezielle Noten
  punkte                INTEGER                 NOT NULL
);
COMMENT ON Table spielstaende IS 'Miniwelt LWBadventure';


-- aufenthaltsorte (!npcNr, !raumNr)
CREATE TABLE aufenthaltsorte (
  npcnr                 INTEGER                 REFERENCES sonstigeNPCs (npcnr),
  raumNr                INTEGER                 REFERENCES raeume (raumNr)
);
COMMENT ON Table aufenthaltsorte IS 'Miniwelt LWBadventure';


-- unterricht (!vNr, !npcNr, !raumNr)
CREATE TABLE unterricht (
  vNr                   INTEGER                 REFERENCES veranstaltungen (vNr),
  npcNr                 INTEGER                 REFERENCES dozent_innen (npcNr),
  raumNr                INTEGER                 REFERENCES raeume (raumNr),
  CONSTRAINT unterrichtRaumCHECK CHECK (raumNr BETWEEN 1 AND 4)
);
COMMENT ON Table unterricht IS 'Miniwelt LWBadventure';


-- assistenz (!vNr,!npcNr)
CREATE TABLE assistenz (
  vNr                   INTEGER                 REFERENCES veranstaltungen (vNr),
  npcNr                 INTEGER                 REFERENCES dozent_innen (npcNr)
);
COMMENT ON Table assistenz IS 'Miniwelt LWBadventure';
```