**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

RA: /listing Raeume

DRC: { RaumNr, RaumName, Ort, Funktion | Räume (RaumNr, RaumName, Ort, Funktion) }

SQL: SELECT \* FROM raeume;

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

*π*Aufgabe(sonstigeNPCs)

RA: project Aufgabe (sonstigeNPCs);

DRC: { Aufgabe | (∃ NPCNr) (sonstigeNPCs (NPCNr, Aufgabe)) }

SQL: SELECT aufgabe FROM sonstigenpcs;

**2a. Welche Lehrveranstaltungen haben 6 SWS?**

*σ*SWS = 6(Veranstaltungen)

RA: select SWS = 6 (Veranstaltungen);

DRC: { VNr, VName, Kürzel, SWS, GebietNr | Veranstaltungen (VNr, VName, Kürzel, SWS, GebietNr) ∧ SWS = 6 }

SQL: SELECT \* FROM veranstaltungen WHERE sws = 6;

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

*σ*Semester = 4(Veranstaltungen)

RA: select Semester = 4 (Veranstaltungen);

DRC: { VNr, VName, Kürzel, SWS, Semester, GebietNr | Veranstaltungen (VNr, VName, Kürzel, SWS, Semester, GebietNr) ∧ Semester = 4 }

SQL: SELECT \* FROM veranstaltungen WHERE semester = 4;

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

*σ*Semester = 4(Minigames ⋈ Veranstaltungen)

RA: select Semester = 4 (Minigames njoin Veranstaltungen);

DRC: { GameNr, GameName, VNr, VName, Kürzel, SWS, Semester, GebietNr | Minigames (GameNr, GameName) ∧ Veranstaltungen (VNr, VName, Kürzel, SWS, Semester, GebietNr) ∧ Semester = 4 }

SQL: SELECT \* FROM minigames NATURAL JOIN veranstaltungen WHERE semester = 4;

**3a. Wie heißen die Spieler\_innen, die bisher das LWB-Adventure gespielt haben?**

*π*SpName(Spieler\_innen)

RA: project SpName (Spieler\_innen);

DRC: { SpName | (∃ SpNr)(∃ Schlüsselanzahl)(∃RaumNr) (Spieler\_innen (SpNr, SpName, Schlüsselanzahl, RaumNr)) }

SQL: SELECT spname FROM spieler\_innen;

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

*π*NPCName(Dozent\_innen ⋈ NPCs)

RA: project NPCName (Dozent\_innen njoin NPCs);

DRC: { NPCName | (∃ NPCNr)(∃ Lieblingsgetränk) (Dozent\_innen (NPCNr, Lieblingsgetränk) ∧ NPCs (NPCNr, NPCName)) }

SQL: SELECT npcname FROM dozent\_innen NATURAL JOIN npcs;

**3c. Welche Aufgabe hat NPC 'Heidi'?**

*π*Aufgabe(*σ*NPCName = 'Heidi'(sonstigeNPCs ⋈ NPCs))

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

DRC: { Aufgabe | (∃ NPCNr)(∃ NPCName) (sonstigeNPCs (NPCNr, Aufgabe) ∧ NPCs (NPCNr, NPCName) ∧ NPCName = 'Heidi') }

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

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

*σ*GebietName = 'Programmierung' (Veranstaltungen ⋈ Themengebiete)

RA: select GebietName = 'Programmierung' (Veranstaltungen njoin Themengebiete);

DRC: { VNr, VName, Kürzel, SWS, Semester, GebietNr, GebietName | Veranstaltungen (VNr, VName, Kürzel, SWS, Semester, GebietNr) ∧ Themengebiete (GebietNr, GebietName) ∧ GebietName = 'Programmierung' }

SQL: SELECT \* FROM veranstaltungen NATURAL JOIN themengebiete WHERE gebietname = 'Programmierung';

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

RA: Wildcards in RA nicht möglich!

DRC: Wildcards in DRC nicht möglich!

SQL: SELECT \* FROM veranstaltungen WHERE vname LIKE '%Daten%' OR vname LIKE '%Programmierung%'; oder

SQL: SELECT \* FROM veranstaltungen NATURAL JOIN themengebiete WHERE gebietname LIKE '%Daten%' OR gebietname LIKE '%Programmierung%';

**5. Was ist das Lieblingsgetränk von Darth Schmidter?**

*π*Lieblingsgetränk(*σ*NPCName = 'Darth Schmidter' (Dozent\_innen ⋈ NPCs))

RA: project Lieblingsgetraenk (select NPCName = 'Darth Schmidter' (Dozent\_innen njoin NPCs));

DRC: { Lieblingsgetränk | (∃ NPCNr)(∃ NPCName) (Dozent\_innen (NPCNr, Lieblingsgetränk) ∧ NPCs (NPCNr, NPCName) ∧ NPCName = 'Darth Schmidter') }

SQL: SELECT lieblingsgetraenk FROM dozent\_innen NATURAL JOIN npcs WHERE npcname = 'Darth Schmidter';

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

*π*VName,Ort((*σ*Ort != 'FU Berlin'(Räume)) ⋈ Unterricht ⋈ Veranstaltungen)

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

oder mit Differenz:

*π*VName,Ort((Räume \ *σ*Ort = 'FU Berlin'(Räume)) ⋈ Unterricht ⋈ Veranstaltungen)

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

DRC: { VName, Semester, Ort | (∃ RaumNr)(∃ RaumName)(∃ Funktion)(∃ VNr)(∃ NPCNr)(∃ VName)(∃ Kürzel)  
(∃ SWS)(∃ GebietNr) (Räume (RaumNr, RaumName, Ort, Funktion) ∧ Unterricht (VNr, NPCNr, RaumNr) ∧ Veranstaltungen (VNr, VName, Kürzel, SWS, Semester, GebietNr) ∧ Ort ≠ 'FU Berlin') }

SQL: 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?**

*π*NPCName(NPCs ⋈ (*π*NPCNr(Dozent\_innen) \ *π*NPCNr(Assistenz)))

RA: project NPCName (NPCs njoin (project NPCNr (Dozent\_innen) difference project NPCNr (Assistenz)));

DRC: { NPCName | (¬∃ NPCNr ) (Assistenz (VNr, NPCNr)) ∧ (∃ NPCNr)(∃ Lieblingsgetränk) (Dozent\_innen (NPCNr, Lieblingsgetränk) ∧ NPCs (NPCNr, NPCName)) }

SQL: 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: AnzahlMinigames)**

AnzahlMinigames ← γCOUNT(\*)(Minigames)

RA: rename Anzahl (Minigames) (group\_by [] count(\*) true (Minigames));

SQL: SELECT COUNT(\*) AS AnzahlMinigames FROM minigames;

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

GesamtanzahlSWS ← γ SUM(SWS)(Veranstaltungen)

RA: rename Gesamtanzahl (SWS) (group\_by [] sum(SWS) true (Veranstaltungen));

SQL: SELECT SUM(sws) AS GesamtanzahlSWS FROM veranstaltungen;

**10. Wie heißt die Veranstaltung mit den meisten SWS?**

*π*Vname(select SWS = MaxAnzahl.SWS ((MaxAnzahl.SWS ← γMAX(sws)(Veranstaltungen)) ⋈ Veranstaltungen))

RA: project VName (select SWS = MaxAnzahl.SWS (rename MaxAnzahl (SWS) (group\_by [] max(SWS) true (Veranstaltungen)) njoin Veranstaltungen));

SQL: SELECT vname FROM veranstaltungen WHERE sws = (SELECT MAX(sws) FROM veranstaltungen);

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

ΤSWS↓(*π*VName,SWS,Semester(*σ*NPCName = 'Winnie the K' (Veranstaltungen ⋈ Unterricht ⋈ NPCs)))

τSWS↓(*π*VName,SWS,Semester(ρSemester ← Raumnr(*σ*NPCName = 'Winnie the K' (Veranstaltungen ⋈ Unterricht ⋈ NPCs))))

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

SQL: 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?**

τAnzahlVeranstaltungen(γOrt, AnzahlVeranstaltungen ← COUNT(\*)(Räume ⋈ Unterricht))

RA: sort AnzahlVeranstaltungen (rename Standorte (OrtName, AnzahlVeranstaltungen) (group\_by Ort Ort,count(VNr) true (Raeume njoin Unterricht)));

SQL: 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 🡐 γSpName, AVG(Note)(Spielstände ⋈ Spieler\_innen)

Q2 🡐 ρSchnitt ← AVG(Note)(Q1)

Q3 🡐 *σ*Schnitt<2.0 ∧ Schnitt>4.0(Q2)

Q4 🡐 τSchnitt,SpName(Q3)

Q5 🡐 *π*SpName(Q4)

oder

*π*SpName(τSchnitt,SpName(*σ*Schnitt<2.0 ∧ Schnitt>4.0(ρSchnitt ← AVG(Note)(γSpName, AVG(Note)(Spielstände ⋈ Spieler\_innen)))))

RA: 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)))));

SQL: 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.

τSchnitt,SpName(*σ*Schnitt<2.0 ∧ Schnitt>4.0(ρSchnitt ← AVG(Note)(γSpName, AVG(Note)(Spielstände ⋈ Spieler\_innen))))

RA: 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))));

SQL: 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;