Einzelprüfung „Datenbanksysteme / Softwaretechnologie (vertieft)“

Einzelprüfungsnummer 66116 / 2019 / Herbst

Thema 2 / Teilaufgabe 2 / Aufgabe 7 (*Formel-1-Rennen*)

Stichwörter: SQL, SQL mit Übungsdatenbank, EXCEPT, Top-N-Query, ALTER TABLE, TRIGGER

Gegeben sind folgende Relationen aus einem Verwaltungssystem für die jährlichen Formel-1-Rennen:

```
Strecke(Strecken_ID, Streckenname, Land, Länge)

Fahrer(Fahrer_ID, Fahrername, Nation, Rennstall)

Rennen(Strecken_ID[Strecke], Jahr, Wetter)

Rennteilnahme(Fahrer_ID[Fahrer], Strecken_ID[Rennen], Jahr[Rennen], Rundenbestzeit, Gesamtzeit,

disqualifiziert)
FK (Strecken_ID, Jahr) referenziert Rennen (Strecken_ID, Jahr)
```

```sql
CREATE TABLE Fahrer (
  Fahrer_ID INTEGER PRIMARY KEY,
  Fahrername VARCHAR(100) NOT NULL,
  Nation VARCHAR(100) NOT NULL,
  Rennstall VARCHAR(100) NOT NULL
);

CREATE TABLE Strecke (
  Strecken_ID INTEGER PRIMARY KEY,
  Streckenname VARCHAR(100) NOT NULL,
  Land VARCHAR(100) NOT NULL,
  Länge NUMERIC(5,3) NOT NULL
);

CREATE TABLE Rennen (
  Strecken_ID INTEGER REFERENCES Strecke(Strecken_ID),
  Jahr INTEGER NOT NULL,
  Wetter VARCHAR(10) NOT NULL,
  PRIMARY KEY (Strecken_ID, Jahr)
);

CREATE TABLE Rennteilnahme (
  Fahrer_ID INTEGER REFERENCES Fahrer(Fahrer_ID),
  Strecken_ID INTEGER REFERENCES Strecke(Strecken_ID),
  Jahr INTEGER NOT NULL,
  Rundenbestzeit NUMERIC(5,3) NOT NULL,
  Gesamtzeit NUMERIC(5,3) NOT NULL,
  disqualifiziert BOOLEAN NOT NULL,
  PRIMARY KEY (Fahrer_ID, Strecken_ID, Jahr)
);
```

```
INSERT INTO Fahrer VALUES
  (1, 'Kimi Räikkönen', 'Finnland', 'Alfa Romeo'),
  (2, 'Rubens Barrichello', 'Brasilien', 'Brawn'),
  (3, 'Fernando Alonso', 'Spanien', 'Ferrari'),
  (4, 'Michael Schumacher', 'Deutschland', 'Ferrari'),
  (5, 'Jenson Button', 'Vereinigtes Königreich Großbritannien', 'McLaren'),
  (6, 'Felipe Massa', 'Brasilien', 'Ferrari'),
  (7, 'Lewis Hamilton', 'Vereinigtes Königreich Großbritannien', 'Williams'),
  (8, 'Riccardo Patrese', 'Italien', 'Williams'),
  (9, 'Sebastian Vettel', 'Deutschland', 'Ferrari'),
  (10, 'Jarno Trulli', 'Italien', 'Toyota');

INSERT INTO Strecke VALUES
  (1, 'Autodromo Nazionale Monza', 'Italien', 5.793),
  (2, 'Circuit de Monaco', 'Monaco', 3.340),
  (3, 'Silverstone Circuit', 'Vereinigtes Königreich', 5.891),
  (4, 'Circuit de Spa-Francorchamps', 'Belgien', 7.004),
  (5, 'Circuit Gilles-Villeneuve', 'Kanada', 4.361),
  (6, 'Nürburgring', 'Deutschland', 5.148),
  (7, 'Hockenheimring', 'Deutschland', 4.574),
  (8, 'Interlagos', 'Brasilien', 4.309),
  (9, 'Hungaroring', 'Ungarn', 4.381),
  (10, 'Red Bull Ring', 'Österreich', 5.942),
  (11, 'Abu Dhabi', 'Abu Dhabi', 5.554);

INSERT INTO Rennen VALUES
  (11, 2011, 'sonnig'),
  (10, 2006, 'sonnig'),
  (9, 2007, 'regnerisch'),
  (8, 2008, 'regnerisch'),
  (7, 2009, 'sonnig'),
  (6, 2010, 'regnerisch'),
  (5, 2011, 'sonnig'),
  (4, 2012, 'sonnig'),
  (3, 2013, 'sonnig'),
  (2, 2014, 'regnerisch'),
  (1, 2015, 'regnerisch');

INSERT INTO Rennteilnahme VALUES
  (1, 11, 2011, 2.001, 90.001, FALSE),
  (2, 11, 2011, 2.002, 90.002, FALSE),
  (3, 11, 2011, 2.003, 90.003, FALSE),
  (4, 11, 2011, 2.004, 89.999, FALSE),
  (5, 11, 2011, 2.005, 90.005, FALSE),
  (6, 11, 2011, 2.005, 99.009, FALSE),
  (4, 10, 2006, 2.782, 90.005, TRUE),
  (3, 10, 2006, 2.298, 90.005, TRUE),
  (3, 9, 2009, 2.253, 90.005, TRUE),
  (2, 10, 2006, 2.005, 90.005, TRUE),
  (2, 9, 2009, 3.298, 90.342, TRUE),
  (2, 8, 2008, 4.782, 78.005, TRUE);
```

Der Einfachheit halber wird angenommen, dass Fahrer den Rennstall nicht wechseln können. Das Attribut „disqualifiziert“ kann die Ausprägungen „ja“ und „nein“ haben. Formulieren

Sie folgende Abfragen in SQL. Vermeiden Sie nach Möglichkeit übermäßige Nutzung von Joins und Views.

(a) Geben Sie für jeden Fahrer seine ID sowie die Anzahl seiner Disqualifikationen in den Jahren 2005 bis 2017 aus. Ordnen Sie die Ausgabe absteigend nach der Anzahl der Disqualifikationen.

Lösungsvorschlag

```
SELECT Fahrer_ID, COUNT(disqualifiziert) as anzahl_disqualifikationen FROM
↪  Rennteilnahme
WHERE disqualifiziert = TRUE
GROUP BY Fahrer_ID, disqualifiziert
ORDER BY anzahl_disqualifikationen DESC;
```

(b) Gesucht sind alle Länder, aus denen noch nie ein Fahrer disqualifiziert wurde.

Lösungsvorschlag

```
SELECT Nation FROM Fahrer GROUP BY Nation
EXCEPT
SELECT f.Nation FROM Fahrer f, Rennteilnahme t
WHERE f.Fahrer_ID = t.Fahrer_ID AND t.disqualifiziert = TRUE
GROUP BY f.Nation;
```

(c) Gesucht sind die ersten fünf Plätze des Rennens von 2011 in „Abu Dhabi“ (Streckenname). Die Ausgabe soll nach der Platzierung absteigend erfolgen. Geben Sie Fahrer_ID, Fahrername, Nation und Rennstall mit aus.

Lösungsvorschlag

Mit LIMIT

```
SELECT f.Fahrer_ID, f.Fahrername, f.Nation, f.Rennstall
FROM Fahrer f, Rennteilnahme t, Strecke s
WHERE
  f.Fahrer_ID = t.Fahrer_ID AND
  s.Strecken_ID = t.Strecken_ID AND
  s.Streckenname = 'Abu Dhabi' AND
  t.Jahr = 2011
ORDER BY t.Gesamtzeit ASC LIMIT 5;
```

Als Top-N-Query:

```
CREATE VIEW Rennen_Abu_Dhabi AS
  SELECT f.Fahrer_ID, f.Fahrername, f.Nation, f.Rennstall, t.Gesamtzeit
  FROM Fahrer f, Rennteilnahme t, Strecke s
  WHERE
    f.Fahrer_ID = t.Fahrer_ID AND
    s.Strecken_ID = t.Strecken_ID AND
    s.Streckenname = 'Abu Dhabi' AND
    t.Jahr = 2011
  ORDER BY t.Gesamtzeit ASC;

SELECT a.Fahrer_ID, a.Fahrername, a.Nation, a.Rennstall
FROM Rennen_Abu_Dhabi a, Rennen_Abu_Dhabi b
```

```
WHERE
  a.Gesamtzeit >= b.Gesamtzeit
GROUP BY a.Fahrer_ID, a.Fahrername, a.Nation, a.Rennstall, a.Gesamtzeit
HAVING COUNT(*) <= 5
ORDER BY a.Gesamtzeit;
```

(d) Führen Sie eine neue Spalte `Gehalt` in die Tabelle `Fahrer` ein. Da sich die Prämien für die Fahrer nach einem Rennstallwechsel ändern, soll ein Trigger geschrieben werden, mit dem das Gehalt des betreffenden Fahrers um 10% angehoben wird.

Lösungsvorschlag

Lösung für PostgreSQL:

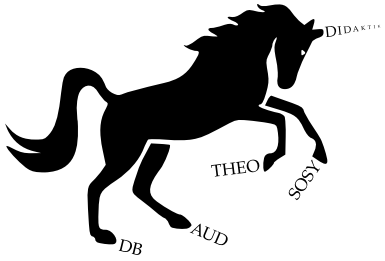
```
ALTER TABLE Fahrer ADD Gehalt numeric(12,2);

UPDATE Fahrer SET Gehalt = 1000000 WHERE Fahrer_ID = 1;
UPDATE Fahrer SET Gehalt = 2000000 WHERE Fahrer_ID = 2;
UPDATE Fahrer SET Gehalt = 3000000 WHERE Fahrer_ID = 3;
UPDATE Fahrer SET Gehalt = 4000000 WHERE Fahrer_ID = 4;
UPDATE Fahrer SET Gehalt = 5000000 WHERE Fahrer_ID = 5;
UPDATE Fahrer SET Gehalt = 6000000 WHERE Fahrer_ID = 6;
UPDATE Fahrer SET Gehalt = 7000000 WHERE Fahrer_ID = 7;
UPDATE Fahrer SET Gehalt = 8000000 WHERE Fahrer_ID = 8;
UPDATE Fahrer SET Gehalt = 9000000 WHERE Fahrer_ID = 9;
UPDATE Fahrer SET Gehalt = 10000000 WHERE Fahrer_ID = 10;

CREATE FUNCTION trigger_function()
  RETURNS TRIGGER
  LANGUAGE PLPGSQL
AS $$
  BEGIN
    UPDATE Fahrer
      SET gehalt = gehalt * 1.1
      WHERE Fahrer_ID = NEW.Fahrer_ID;
    RETURN NEW;
  END;
$$;

CREATE TRIGGER mehr_gehalt
  AFTER UPDATE OF Rennstall ON Fahrer
  FOR EACH ROW EXECUTE PROCEDURE trigger_function();

-- Test:
SELECT * FROM FAHRER WHERE Fahrer_ID = 1;
UPDATE Fahrer SET Rennstall = 'Red Bull' WHERE Fahrer_ID = 1;
SELECT * FROM FAHRER WHERE Fahrer_ID = 1;
```



Die Bschlangaul-Sammlung
Hermine Bschlangaul and Friends

Eine freie Aufgabensammlung mit Lösungen von Studierenden für Studierende zur Vorbereitung auf die 1. Staatsexamensprüfungen des Lehramts Informatik in Bayern.

Hilf mit! Die Hermine schafft das nicht allein! Das ist ein Community-Projekt! Verbesserungsvorschläge, Fehlerkorrekturen, weitere Lösungen sind herzlich willkommen - egal wie - per Pull-Request oder per E-Mail an hermine.bschlangaul@gmx.net.Der TEX-Quelltext dieses Dokuments kann unter folgender URL aufgerufen werden: `https://github.com/bschlangaul-sammlung/examens-aufgaben/blob/main/Staatsexamen/66116/2019/09/Thema-2/Teilaufgabe-2/Aufgabe-7.tex`