

## Aufgabe 7: (SQL)

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)

```
1 CREATE TABLE Fahrer (  
2     Fahrer_ID integer PRIMARY KEY,  
3     Fahrername varchar(100) NOT NULL,  
4     Nation varchar(100) NOT NULL,  
5     Rennstall varchar(100) NOT NULL  
6 );  
7  
8 CREATE TABLE Strecke (  
9     Strecken_ID integer PRIMARY KEY,  
10    Streckenname varchar(100) NOT NULL,  
11    Land varchar(100) NOT NULL,  
12    Länge numeric(5,3) NOT NULL  
13 );  
14  
15 CREATE TABLE Rennen (  
16     Strecken_ID integer REFERENCES Strecke(Strecken_ID),  
17     Jahr integer NOT NULL,  
18     Wetter varchar(10) NOT NULL,  
19     PRIMARY KEY (Strecken_ID, Jahr)  
20 );  
21  
22 CREATE TABLE Rennteilnahme (  
23     Fahrer_ID integer REFERENCES Fahrer(Fahrer_ID),  
24     Strecken_ID integer REFERENCES Strecke(Strecken_ID),  
25     Jahr integer NOT NULL,  
26     Rundenbestzeit numeric(5,3) NOT NULL,  
27     Gesamtzeit numeric(5,3) NOT NULL,  
28     disqualifiziert boolean NOT NULL,  
29     PRIMARY KEY (Fahrer_ID, Strecken_ID, Jahr)  
30 );  
31  
32 INSERT INTO Fahrer VALUES  
33 (1, 'Kimi Räikkönen', 'Finnland', 'Alfa Romeo'),  
34 (2, 'Rubens Barrichello', 'Brasilien', 'Brawn'),  
35 (3, 'Fernando Alonso', 'Spanien', 'Ferrari'),  
36 (4, 'Michael Schumacher', 'Deutschland', 'Ferrari'),  
37 (5, 'Jenson Button', 'Vereinigtes Königreich Großbritannien', 'McLaren'),  
38 (6, 'Felipe Massa', 'Brasilien', 'Ferrari'),  
39 (7, 'Lewis Hamilton', 'Vereinigtes Königreich Großbritannien', 'Williams'),  
40 (8, 'Riccardo Patrese', 'Italien', 'Williams'),  
41 (9, 'Sebastian Vettel', 'Deutschland', 'Ferrari'),  
42 (10, 'Jarno Trulli', 'Italien', 'Toyota');  
43  
44 INSERT INTO Strecke VALUES  
45 (1, 'Autodromo Nazionale Monza', 'Italien', 5.793),  
46 (2, 'Circuit de Monaco', 'Monaco', 3.340),  
47 (3, 'Silverstone Circuit', 'Vereinigtes Königreich', 5.891),  
48 (4, 'Circuit de Spa-Francorchamps', 'Belgien', 7.004),
```

```

49 (5, 'Circuit Gilles-Villeneuve', 'Kanada', 4.361),
50 (6, 'Nürburgring', 'Deutschland', 5.148),
51 (7, 'Hockenheimring', 'Deutschland', 4.574),
52 (8, 'Interlagos', 'Brasilien', 4.309),
53 (9, 'Hungaroring', 'Ungarn', 4.381),
54 (10, 'Red Bull Ring', 'Österreich', 5.942),
55 (11, 'Abu Dhabi', 'Abu Dhabi', 5.554);
56
57 INSERT INTO Rennen VALUES
58 (11, 2011, 'sonnig'),
59 (10, 2006, 'sonnig'),
60 (9, 2007, 'regnerisch'),
61 (8, 2008, 'regnerisch'),
62 (7, 2009, 'sonnig'),
63 (6, 2010, 'regnerisch'),
64 (5, 2011, 'sonnig'),
65 (4, 2012, 'sonnig'),
66 (3, 2013, 'sonnig'),
67 (2, 2014, 'regnerisch'),
68 (1, 2015, 'regnerisch');
69
70 INSERT INTO Rennteilnahme VALUES
71 (1, 11, 2011, 2.001, 90.001, FALSE),
72 (2, 11, 2011, 2.002, 90.002, FALSE),
73 (3, 11, 2011, 2.003, 90.003, FALSE),
74 (4, 11, 2011, 2.004, 89.999, FALSE),
75 (5, 11, 2011, 2.005, 90.005, FALSE),
76 (6, 11, 2011, 2.005, 99.009, FALSE),
77 (4, 10, 2006, 2.782, 90.005, TRUE),
78 (3, 10, 2006, 2.298, 90.005, TRUE),
79 (3, 9, 2009, 2.253, 90.005, TRUE),
80 (2, 10, 2006, 2.005, 90.005, TRUE),
81 (2, 9, 2009, 3.298, 90.342, TRUE),
82 (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.

```

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

```

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

```

1 SELECT Nation FROM Fahrer GROUP BY Nation
2 EXCEPT
3 SELECT f.Nation FROM Fahrer f, Rennteilnahme t
4 WHERE f.Fahrer_ID = t.Fahrer_ID AND t.disqualifiziert = TRUE
5 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.

Mit LIMIT

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

Als Top-N-Query:

```
1 CREATE VIEW Rennen_Abu_Dhabi AS
2     SELECT f.Fahrer_ID, f.Fahrername, f.Nation, f.Rennstall,
3           ↪ t.Gesamtzeit
4     FROM Fahrer f, Rennteilnahme t, Strecke s
5     WHERE
6         f.Fahrer_ID = t.Fahrer_ID AND
7         s.Strecken_ID = t.Strecken_ID AND
8         s.Streckenname = 'Abu Dhabi' AND
9         t.Jahr = 2011
10    ORDER BY t.Gesamtzeit ASC;
11
12 SELECT a.Fahrer_ID, a.Fahrername, a.Nation, a.Rennstall
13 FROM Rennen_Abu_Dhabi a, Rennen_Abu_Dhabi b
14 WHERE
15     a.Gesamtzeit >= b.Gesamtzeit
16 GROUP BY a.Fahrer_ID, a.Fahrername, a.Nation, a.Rennstall,
17     ↪ a.Gesamtzeit
18 HAVING COUNT(*) <= 5
19 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.

Lösung für PostgreSQL:

```
1 ALTER TABLE Fahrer ADD Gehalt numeric(12,2);
2
3 UPDATE Fahrer SET Gehalt = 1000000 WHERE Fahrer_ID = 1;
4 UPDATE Fahrer SET Gehalt = 2000000 WHERE Fahrer_ID = 2;
5 UPDATE Fahrer SET Gehalt = 3000000 WHERE Fahrer_ID = 3;
6 UPDATE Fahrer SET Gehalt = 4000000 WHERE Fahrer_ID = 4;
7 UPDATE Fahrer SET Gehalt = 5000000 WHERE Fahrer_ID = 5;
8 UPDATE Fahrer SET Gehalt = 6000000 WHERE Fahrer_ID = 6;
9 UPDATE Fahrer SET Gehalt = 7000000 WHERE Fahrer_ID = 7;
10 UPDATE Fahrer SET Gehalt = 8000000 WHERE Fahrer_ID = 8;
11 UPDATE Fahrer SET Gehalt = 9000000 WHERE Fahrer_ID = 9;
12 UPDATE Fahrer SET Gehalt = 10000000 WHERE Fahrer_ID = 10;
13
14 CREATE FUNCTION trigger_function()
15     RETURNS TRIGGER
```

```

16  LANGUAGE PLPGSQL
17  AS $$
18  BEGIN
19      UPDATE Fahrer
20          SET gehalt = gehalt * 1.1
21          WHERE Fahrer_ID = NEW.Fahrer_ID;
22      RETURN NEW;
23  END;
24  $$;
25
26  CREATE TRIGGER mehr_gehalt
27      AFTER UPDATE OF Rennstall ON Fahrer
28      FOR EACH ROW EXECUTE PROCEDURE trigger_function();
29
30  -- Test:
31  SELECT * FROM FAHRER WHERE Fahrer_ID = 1;
32  UPDATE Fahrer SET Rennstall = 'Red Bull' WHERE Fahrer_ID = 1;
33  SELECT * FROM FAHRER WHERE Fahrer_ID = 1;

```