Händler-Datenbank (SQL-Beispiel)/Identität

Wechseln zu:Navigation, Suche

Dieser Artikel erfüllt die GlossarWiki-Qualitätsanforderungen:

Korrektheit: 4 (großteils überprüft)	**Umfang**: 4 (unwichtige Fakten fehlen)	**Quellenangaben**: 4 (fast vollständig vorhanden)	**Quellenarten**: 5 (ausgezeichnet)	**Konformität**: 5 (ausgezeichnet)

Die nachfolgenden Beispiele können beispielsweise mit SQLite oder PostgreSQL getestet werden. Installieren Sie dazu die zugehörige Händler-Datenbank.

Inhaltsverzeichnis

1 Die Identität

Die Identitätsfunktion verändert eine Tabelle nicht

Die Identitätsfunktion der Relationalen Algebra ist eine triviale Funktion: Sie bildet eine Relation (Tabelle) auf sich selbst ab:

[math]id: R \rightarrow R[/math]
[math]id(r) = r[/math]

Diese Funtion ist *idempotent*:

[math]id(id(r)) = id(r) = r[/math]

2 Beispiele bezüglich der Händler-

Datenbank

Den nachfolgenden Beispielen liegt die Händler-Datenbank zugrunde. Alle Beispiele wurden mit PostgreSQL und SQLite getestet. In jedem Fall wird Händlertabelle auf sich selbst abgebildet:

[math]\texttt{haendler}[/math] → [math]\textrm{id}(\texttt{haendler})[/math]

h_id	h_name	h_ortschaft
1	Maier	Königsbrunn
2	Müller	Königsbrunn
3	Maier	Augsburg
4	Huber	NULL
5	Schmidt	Hamburg

→

h_id	h_name	h_ortschaft
1	Maier	Königsbrunn
2	Müller	Königsbrunn
3	Maier	Augsburg
4	Huber	NULL
5	Schmidt	Hamburg

In SQL ist es nicht möglich, auf den Inhalt einer benannten Tabelle zuzugreifen, indem man einfach den Namen der Tabelle angibt. Folgendes ist also keine korrekte SQL-Anfrage:

```
haendler
```

Um den Inhalt einer Tabelle auszugeben, benötigt man daher in SQL auf jeden Fall eine Identitätsfunktion. In SQL-92 gab es noch eine dedizierte Identitätsfunktion: `TABLE`[1]

```
TABLE haendler
```

Dieser Befehl wird beispielsweise von PostgreSQL unterstützt . In SQL-99 und in den meisten SQL-Datenbank-Management-Systemen, wie beispielsweise SQLite, gibt es diese Funktion allerdings nicht mehr.[2] Als Identitätsfunktion kommt hier ersatzweise `SELECT * FROM` zum Einsatz:

```
SELECT * FROM haendler
```

Das zuvor formulierte Idempozenz-Gesetz gilt auch in SQL, wenn auch mit der genannten Einschränkung, dass es sich bei der direkten Angabe eines Tabellennamens um keinen korrekten SQL-Befehl handelt:

```
SELECT * FROM (SELECT * FROM haendler)
=
SELECT * FROM haendler
=
TABLE haendler /* SQL-92, aber nicht SQL-99 */
=
haendler /* kein SQL*/
```

Anmerkung 1
In PostgreSQL läuft – im Gegensatz zu SQLite – die erste der drei obigen Anfrage auf einen Fehler, da hier Unterabfragen stets benannt werden müssen. In PostgreSQL muss man die erste Select-Anweisung daher folgendermaßen schreiben:

```
SELECT * FROM (SELECT * FROM haendler) AS dummy
```

Anmerkung 2
Die Kurzschreibweise * sollte man i. Allg. nur für Ad-hoc-Anfragen in Tools wie phpPgAdmin oder pgAdmin verwenden. In einem Programmcode sollte man darauf verzichten und besser alle Attribute explizit aufführen. Das heißt, an Stelle von

```
SELECT * FROM haendler
```

sollte man

```
SELECT h_id, h_name, h_ortschaft FROM haendler
```

schreiben. Der Grund ist, dass die SELECT-*-Anfrage ihre Bedeutung ändert, wenn sich das Schema der Datenbank ändert, wenn also z.B. ein weiteres Attribut zur Tabelle `haendler` hinzugefügt wird (vgl. die Tabelle `haendler` in Händler-Datenbank). Das heißt, aufgrund einer Schema-Änderung könnte sich ungewollt das Verhalten eines Programms ändern. Zum Beispiel könnte eine Tabelle, die das Ergbnis einer Select-Anfrage ausgibt, plötzlich mehr Spalten als geplant enthalten. Darüber hinaus ist die Attribut-Reihenfolge bei SELECT-*-Anfragen nicht festgelegt. Auch diese kann sich plötzlich unerwartet ändern – zum Beispiel nach einer Modifikation der Domäne eines der Attribute, wie z. B. der Änderung der Anzahl der erlaubten Zeichen der Domäne des Attributs `h_ortschaft` (beispielsweise `VARCHAR(50)` → `VARCHAR(100)`).

3 Quellen

Date, Darwen (1993): Christopher J. Date und Hugh Darwen; A Guide to the SQL Standard – A user's guid to the standard relational language SQL; Auflage: 3; Verlag: Addison-Wesley; Adresse: Reading, Massachusetts, USA; 1993; Quellengüte: 5 (Buch), S. 126
Gulutzan, Pelzer (1999): Peter Gulutzan und Trudy Pelzer; SQL-99 complete, Really – An Example-Based Reference Manual of the New Standard; Verlag: R&D Books; ISBN: 0-87930-568-1; 1999;

Quellengüte: 5 (Buch)
Kowarschick (MMDB-Skript): Wolfgang Kowarschick; Vorlesung Multimedia-Datenbanksysteme – Sommersemester 2018; Hochschule: Hochschule Augsburg; Adresse: Augsburg; Web-Link; 2018; Quellengüte: 4 (Skript)
Kowarschick (MMDB): Wolfgang Kowarschick; Vorlesung „Multimedia-Datenbanksysteme“; Hochschule: Hochschule Augsburg; Adresse: Augsburg; Web-Link; 2016; Quellengüte: 3 (Vorlesung), http://mmdb.hs-augsburg.de/beispiel/haendler/

4 Siehe auch

Händler-Datenbank (SQL-Beispiel)
Händler-Datenbank (SQL-Beispiel)/Projektion
Händler-Datenbank (SQL-Beispiel)/Selektion

Kategorien:
PostgreSQL-Beispiel
Praktikum:MMDB