Prof. Dr. Stefan Brass
Institut für Informatik
MLU Halle-Wittenberg

12. Januar 2012

Datenbanken II B: DBMS-Implementierung — Hausaufgabe 9A —

In Aufgabe 8A wurde die Entwicklung eines einfachen Systemkatalogs mit Releationen-Objekten begonnen. Jetzt soll die Entwicklung mit Spalten-Objekten fortgesetzt werden. Entwickeln Sie dazu eine abstrakte Oberklasse `col_c` und Unterklassen `col_int_c` und `col_str_c`. Es soll leicht möglich sein, später weitere Unterklassen zu definieren, z.B. für TIDs/ROWIDs. Die Oberklasse kann nicht direkt instanziiert werden, d.h. man kann Objekte nur von den Typ-spezifischen Unterklassen erzeugen.

Die Spalten-Objekte haben drei Anwendungen:

- Vor dem Anlegen der Relation werden sie dem Relations-Objekt zugeordnet (das Relationsobjekt muss also eine verkettete Liste von Spalten-Objekten enthalten). Wenn man für die Relation dann `create` aufruft, werden die Spalten mit im Data Dictionary gespeichert.
- Auch vor dem Öffnen einer existierenden Relation ordnet man dem Relationsobjekt Objekte für die Spalten zu, für die man sich später interessiert (das müssen nicht immer alle Spalten sein). Beim Öffnen wird dann geprüft, dass es die Spalten wirklich gibt, und es werden Zugriffsdaten für die Spalten geladen (z.B. Offsets in Tupeln fester Länge).
- Später werden wir u.a. den Full Table Scan implementieren. Dann muß man natürlich auf jeden Spaltenwert des aktuellen Tupels zugreifen können. Dazu gibt man ein Spaltenobjekt an (dieses enthält ja die Zugriffsdaten für die Spalte in einem Tupel der Relation).

Im Moment müssen Sie für die Spalten-Klassen nur den Konstruktor implementieren:

- `col_int_c(rel_t rel, str_t name)`
- `col_str_c(rel_t rel, str_t name, int maxlen)`

Sie müssen aber auch die Klasse `rel_c` und hier besonders die Methoden `create` und `open` entsprechend erweitern. Wenn Sie Tupel fester Länge benutzen, können Sie für die Spalten-Objekte auch schon einen Offset verwalten. In jedem Fall sollten Sie sich gedanken über das Tupelformat machen.

Wenn Sie nicht Subklassen für die verschiedenen Datentypen wollen, sondern nur eine Klasse mit einem Typ-Attribut, dürfen Sie auch diese Variante wählen. Falls der String-Datentyp eine Länge bzw. Maximallänge hat, und der `int`-Datentyp keine Längenangabe braucht, wird dieser Punkt etwas schwieriger. Allgemein ist die Schnittstelle wieder nur als Vorschlag zu verstehen. Z.B. wäre es möglich, Konstruktoren ohne Parameter anzubieten, und die Daten dann später zu setzen. Sie dürfen die maximale Länge von Spaltennamen begrenzen, z.B. auf 18 (wie in SQL-86).

Es wäre später auch wünschenswert, die Spalten einer Tabelle abfragen zu können. Das ist mit der obigen Schnittstelle nicht möglich. Falls Sie das Data Dictionary über normale Relationen implementieren, hat man diese Funktionalität natürlich automatisch ohne zusätzlichen Aufwand. Ansonsten wäre zu überlegen, ob man später einen zusätzlichen Scan speziell für Spalten einführt, oder eine Methode, mit der man die `i`-te Spalte einer Tabelle (Name und Datentyp) abfragen kann. Die Relationenklasse müsste dann die Möglichkeit bieten, die Anzahl Spalten der Relation abzufragen.