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

20. Oktober 2011

Datenbanken II B: DBMS-Implementierung — Hausaufgabe 1 (Diskussonsvorschlag) —

Für unser kleines DBMS-Projekt legen Sie bitte eine Datei `ver.h` an, in der Sie eine Konstante `VER_BLOCKSIZE` als 8192 (8K) definieren. In dieser Datei sollen später noch weitere Konstanten (Parameter, Optionen) definiert werden. Eine elegantere Lösung wäre natürlich, solche Werte von einer Konfigurationsdatei bei Programmstart zu lesen. Dann müßte das Programm nicht neu compiliert werden, wenn man diese Werte ändert. Aber wir wollen den Aufwand begrenzen, und uns eher auf die Datenbank selbst konzentrieren. Es wäre aber günstig, im Hinterkopf zu behalten, dass die Konstanten eventuell später durch Variablen ersetzt werden müssen, was z.B. bei Verwendung in Arraygrößen zu Schwierigkeiten führen könnte.

Implementieren Sie bitte eine Klasse `file_c` (in Dateien `file.h` und `file.cpp`) mit folgenden Methoden:

- Einen Konstruktor mit dem Dateinamen und einer ID (`int`) als Argument.
- Eine Methode `filename`, die den Dateinamen liefert.
- Eine Methode `id`, die die ID liefert.
- Eine Methode `create` mit einer Anzahl Blöcken `int blocks` als Argument. Diese Methode soll die Datei mit `blocks * VER_BLOCKSIZE` Bytes beschreiben. Der erste Block soll dabei eine "magic number" enthalten, an der dieses Programm erkennen kann, dass die Datei von diesem Programm erzeugt wurde (Dateiformat). Außerdem soll die Anzahl Blöcke der Datei dort gespeichert werden. Es ist damit zu rechnen, dass später weitere Informationen in diesen speziellen "Header Block" gespeichert werden. Alle anderen Bytes der Datei sollen mit Null-Bytes initialisiert werden. Die Datei muß natürlich vorher geöffnet und hinterher geschlossen werden. Alle Methoden sollen `true` zurückliefern, falls der Aufruf erfolgreich war, und `false` im Fehlerfall. Diese Methode darf nur aufgerufen werden, wenn die Datei nicht bereits geöffnet ist. Bei solchen Programmierfehlern dürfen Sie das Programm abbrechen (z.B. durch Verwendung von "`assert.h`").
- Eine Methode `open`, die die Datei zum Lesen und Schreiben eröffnet. Sie muß vorher geschlossen sein.
- Eine Methode `close` zum Schließen. Die Datei muß vorher geöffnet gewesen sein.
- Eine Methode `read` mit den Parametern Blocknummer "`int block`" und Hauptspeicheradresse "`void *buf`": Der betreffende Block soll von der Datei in den Hauptspeicher ab Adresse `buf`gelesen werden. Die Datei muß natürlich geöffnet sein.

- Eine Methode `write` mit den Parametern Blocknummer "`int block`" und Hauptspeicheradresse "`void *buf`": Der betreffende Block soll vom Hauptspeicher (ab Adresse `buf`) in die Datei geschrieben werden. Die Datei muß natürlich geöffnet sein.
- Eine Methode `sync` ohne Parameter, die sicherstellt, dass alle mit `write` geschriebenen Blöcke auch wirklich auf die Platte geschrieben wurden, und nicht etwa noch in Puffern des Betriebssystems auf das eigentliche Schreiben warten. Die Datei muß geöffnet sein.
- Eine Methode `extend` mit einem Parameter `blocks`, der die Datei um diese Anzahl Blöcke verlängert. Die Datei muß dazu vorher geöffnet worden sein.