Praktikumsaufgabe 2

Lernziele

Wiederholung und weiteres Verständnis der in EPR/OOA behandelten C/C++-Programmierung. Erlernen des Konzepts des Datenbankzugriffs über ein natives Call Level Interfaces. Kennenlernen des Transaktionsbegriffs.

Vorbereitung

Folgende Informationen müssen Sie zum *Antestat* bereithalten:

- Wie können in einem C++-Programm die Funktionsdefinitionen auf mehrere Sourcedateien verteilt werden? Welche Schritte werden bis zur Erstellung des Executables durchlaufen? (Wiederholung EPR)
- Wie funktioniert Compilieren und Linken mit dem *gcc*? Wie werden Bibliotheken verwendet? ([1] Kap. 12)
- Wie kann die Projektverwaltung mit *make* und einem *Makefile* automatisiert werden? ([1] Kap. 12) Erläutern Sie das bereitgestellte Makefile.
- Wie kann man in einem C++-Programm die Kommandozeilenargumente auslesen? Legen Sie Beispielcode für das Auslesen der Argumente vor. ([2] Kap. 11.5)
- Wie wandelt man einen C-String (*char**) in einen STL-String um? Wie hängt man an einen STL-String einen C-String an? Wie kann man eine C Stringfunktion (z.B. *sprintf*) mit einem STL-String als Argument auf? Wieso vermeidet man dadurch den berüchtigten Buffer-Overflow Fehler? (Wiederholung OOA)
- Legen Sie ein schematisches Flussdiagramm für Ihr Hauptprogramm vor, aus dem der grobe Programmablauf mit den einzelnen Schritten deutlich wird. (Bemerkung: der Ablauf soll klar werden, deshalb bitte nicht auf die Ebene der Programmvariablen heruntergehen, sondern beschreibende Texte verwenden!)

Aufgabe

Es soll ein Programm "importflights" geschrieben werden, das die Daten aus der Tabelle *flightdata* der Datenbank *praktikum* in Ihre persönliche Datenbank *dbXY* mit den in Aufgabe 1 angelegten Tabellen *flugdaten* und *airlines* überträgt. Der Datenbankzugriff soll über die Bibliothek *libpq* erfolgen.

Die eigentliche Übertragung erfolgt in der Funktion *flightcopy*, deren Prototyp in der Datei *copydata.h* deklariert ist. Diese benutzt die Funktion *addairline*. Weitere Funktionen können Sie nach Bedarf hinzufügen. Der Code ist auf die zwei Sourcedateien *main.cpp* und *copydata.cpp* zu verteilen. Tragen Sie in die bereitgestellten Templates als estes Ihre Namen ein.

Kommandozeilenoptionen Das Programm soll folgendermaßen aufgerufen werden:

```
Usage:
        importflights [options]
Options:
        -del  delete table flugdaten before import
        -u    database user
        -p    password
        -h    database host
        -from database name, from which data is copied
        -to   database name, to which data is copied
```

Die Reihenfolge der Optionen ist egal. Bei fehlerhaftem Aufruf (z.B. unbekannte mit '-' beginnende Option) wird die obige Meldung ausgegeben und abgebrochen.

Funktionalität *importflights* soll sich wie folgt verhalten:

- Um auf beide Datenbanken zugreifen zu können, müssen Sie zu *beiden* Datenbanken eine Verbindung aufbauen.
- Der ganze Import erfolgt in einer Transaktion: bei Erfolg *commit* und bei einem Fehler Abbruch und *rollback*.
- Es werden nur die neuen Daten importiert (anhand eines geeigneten Kriteriums selektieren).
- Noch nicht in der Referenztabelle *airlines* hinterlegte Fluggesellschaften werden automatisch beim Import angelegt.
- Wenn über die Option *-del* gewünscht, wird vor dem Import der Tabelleninhalt gelöscht (innerhalb der Transaktion) und alle Daten komplett übernommen.

Am Ende gibt das Programm eine Importstatistik aus mit der Anzahl der vorher vorhanden Datensätze, der Anzahl der importierten Datensätze und der Anzahl der nach dem Import vorhandenen Datensätze.

Hinweise zur C++-Programmierung

- Makefile und *copydata.h* für diese Aufgabe finden Sie als *src-template.tgz* auf der Webseite zu dieser Veranstaltung.
- Wichtige Kommandozeilenoptionen für den *gcc* sind *-g* (enable debugging), *-c* (Compilieren ohne Linken), *-I* (weiteres Suchverzeichnis für Include-Dateien), *-o* (Outputfile beim Linken) und *-L* (weiteres Suchverzeichnis für Libraries beim Linken).
- Um gegen die PostgreSQL-Library *libpq* zu linken, müssen Sie beim Linken am Ende die Option `-lpq` angeben. Um gegen die STL zu Linken, müssen die Option `-lstdc++` verwenden.
- Das Verzeichnis mit den Postgres-Includedateien liefert der Befehl *pg_config −−includedir*, das Verzeichnis mit den Link-Libraries der Befehl *pg_config −−libdir*.

- Zum Debuggen können Sie den "Data Display Debugger" *ddd* verwenden. *ddd* ist ein grafisches Frontend zum *gdb*. Im unteren Fenster des *ddd* können Sie auch direkt Kommandos an den *gdb* absetzen, ohne sich durch unzählige Menüs hangeln zu müssen. Nützliche *gdb*-Kommandos sind *run arg1 arg2 ...* (Startet Programm mit angegebenen Argumenten), *print var* (druckt Inhalt der Variablen *var* aus) und *list file:line* (öffnet direkt Datei *file* und springt an Zeile *line*).

Referenzen

[1] Welsh, Kaufmann: *Linux - Wegweiser zur Installation&Anwendung.* Semesterapparat (`TWR Wels`)

[2] Karlheinz Zeiner: *Programmieren Lernen mit C.* Das Lehrbuch aus dem Fach EPR

[3] Hartwig: *PostgreSQL Professionell und Praxisnah.* Semesterapparat (`TWY Hart`)

[4] The PostgreSQL Global Development Group:
PostgreSQL 9.1.8 Dokumentation. `http://www.postgresql.org/docs/` (2013)
Kapitel "Client Interfaces, libpq"

[5] Die Testdaten unter `$HOME/../D999999/pub/aufg2dat.tar` können Sie mit dem Befehl *tar xf ...* entpacken. Dort liegt auch die Header-Datei *db.h* und das *Makefile.2a*.