TU München, Fakultät für Informatik
Lehrstuhl III: Datenbanksysteme
Prof. Alfons Kemper, Ph.D.

Übung zur Vorlesung *Einsatz und Realisierung von Datenbanken* im SoSe23

Alice Rey, Maximilian Bandle, Michael Jungmair (i3erdb@in.tum.de)
http://db.in.tum.de/teaching/ss23/impldb/

Blatt Nr. 03

Hinweise Die Datalogaufgaben können auf `https://souffle.db.in.tum.de/` getestet werden. Auf der Seite kann unter *examples* ein entsprechender Datensatz geladen werden. Die neuen IDB Regeln sollten am Ende der EDB definiert und dann im Query-Eingabefeld abgefragt werden.

Zusätzlich zu der in der Vorlesung vorgestellten Syntax hier noch eine Kurzübersicht der Vergleichsoperatoren: $X < Y, Y > X$ (kleiner, größer), $X <= Y, X >= Y$ (kleiner oder gleich, größer oder gleich), $X = Y, X! = Y$ (gleich, ungleich), $!pred(X, Y)$ (existiert nicht pred(X,Y)).

Hausaufgabe 1

Die AMU (Alexander-Maximilians-Universität) hat eine Datenbank mit den Durchschnittsnoten aller Studenten mit Name.

Schema: {[Name | Durchschnittsnote]}

Der einzige Schutzmechanismus dieser Datenbank ist, dass immer mindestens 3 Tupel aggregiert werden. Als Ausgabe sind nur `COUNT` und `AVG` zulässig.

1. Beschreibe eine Methode, um herauszufinden, was die Note des schlechtesten Studenten der AMU ist
2. Max will Alex' Durchschnittsnote herausfinden. Dazu stellt er folgende Anfragen mit Ergebnis:

   ```
   SELECT AVG(Durchschnittsnote), COUNT(*) FROM Noten => (2,5 ; 10.000)
   SELECT AVG(Durchschnittsnote), COUNT(*) FROM Noten WHERE Name != 'Alex'
   => (2,5001 ; 9.999)
   ```

   Kann er aus den Ergebnissen Alex' Note berechnen? Wenn ja, wie, wenn nein, wieso nicht?

Hausaufgabe 2

Die Prüfungs-Datenbank der AMU wurde leider von einem unachtsamen Programmierer geschrieben. Es gibt ein Formular, in dem man nach seinen Klausurnoten suchen kann, allerdings wird die Benutzereingabe nicht geprüft.

Schema: Prüfung: {[Vorlesung, Note, Matrikelnummer]}

Benutzte Anfrage:

```
SELECT *
FROM Prüfung
WHERE Matrikelnummer='{MatrikelNr}' AND Vorlesung='{Benutzereingabe}'
```

- Benutzereingabe: ist die Benutzereingabe.
- MatrikelNr: wird automatisch mit deiner Matrikelnummer befüllt.

Schreibe eine Benutzereingabe, mit der du alle deine Noten auf 1.0 setzen kannst.

Hausaufgabe 3

Sie wollen der AMU helfen, ihre Datenbank sicherer zu machen, und finden bei einer kurzen Suche im Internet *Prepared Statements* und *Input Sanitization*.

1. Erklären Sie kurz beide Methoden, besonders deren Unterschiede in der Behandlung von bösartigen Eingaben.
2. Beschreiben Sie Vorteile von *Prepared Statements*, die über Sicherheit hinaus reichen.
3. Nachfolgend sehen Sie die Funktion `exec_unsafe()`, die das Formular serverseitig aufruft, um die Klausurnote abzufragen. Ersetzen Sie diese durch eine Funktion `exec_prep()`, die mittels eines *Prepared Statements* auf die Datenbank zugreift.

```
#include <pqxx/pqxx>
#include <iostream>
#include <string>
pqxx::result exec_unsafe(pqxx::connection& conn, int matrnr, std::string vorl){
  std::string q = "SELECT␣*␣FROM␣pruefung␣WHERE␣matrikelnummer=",
    q2 = "AND␣vorlesung='", q3 ="'";
  pqxx::work tx{conn, ""}; // Begin of transaction
  pqxx::result r(tx.exec(q + std::to_string(matrnr) + q2 + vorl + q3));
  tx.commit(); // Commit transaction
  return r;
}
int main(int argc, char* argv[]){
  pqxx::connection conn;
  auto r = exec_unsafe(conn, 123, "Grundzuege");
  for(auto row: r)
     std::cout << row["note"] << std::endl;
  return 0;
}
```

Hausaufgabe 4

Bob hat ein Vorlesungsverzeichnis für die Universität programmiert und unter `http://db.in.tum.de/~schuele/sql_verzeichnis.html` online gestellt.

Um die Suche zu erleichtern, kann die Anzahl der SWS durch einen Parameter eingeschränkt werden. Finden Sie einen speziell präparierten Parameter, bei dessen Eingabe statt der Vorlesungen die Liste der Studenten ausgegeben wird. Die Datenbank folgt dem bekannten Universitätsschema.

Bob erfährt von der Sicherheitslücke und schlägt vor, die bekannten Tabellen einmalig mit zufälligen Namen umzubennen, so seien sie nicht zu finden. Würde diese *Sicherheitsmaßnahme* helfen?

Hausaufgabe 5

Sie haben die User-Tabelle zweier Pizzalieferanten ausgelesen, jedoch scheinen die Passwörter uncharakteristisch kompliziert zu sein. Das von Ihnen erhaltene Resultat ist das folgende:

```
id          name        password
----------  ----------  ----------------------------------------
```

```
1           luigi       4d75e8db6a4b6205d0a95854d634c27a
2           mario       fe78ea401158dd5847c4090b8bb22477e510febf
```

- Was könnte der Grund für diese hexadezimalen, 32 bzw. 40 Stellen langen Passwörter sein?
- Können Sie trotzdem den Klartext finden?
- Wie können Sie das Passwort sicherer speichern?
- Wie können Sie für diese Art von Passwortspeicherung Bruteforce-Attacken erschweren?

Gruppenaufgabe 6

Sie fangen die folgende, mit RSA verschlüsselte Nachricht ab: 13. Sie kennen den öffentlichen Schlüssel (3,15). Wie lautet die Nachricht im Klartext? Geben Sie die komplette Herleitung an.

Hausaufgabe 7

Ein bekannter Anwendungsbereich des RSA-Kryptosystems ist das Verschlüsseln und Signieren von E-Mails. Die beiden Standards sind S/MIME und OpenPGP. Für ersteres benötigen Sie ein Zertifikat, ausgestellt von einer Zertifizierungsstelle, wie sie die TUM für alle E-Mail-Adressen ausgibt: `https://wiki.rbg.tum.de/Informatik/Helpdesk/Mail#A_2.2_Zertifikat_f_195_188r_nicht_in.tum.de_Adresse`

Ein Schlüsselpaar für OpenPGP können Sie sich jederzeit selbst und für jede E-Mail-Adresse zulegen. Beschäftigen Sie sich näher mit OpenPGP, damit Sie Ihre E-Mails verschlüsseln können und schicken Sie Ihrem Tutor eine verschlüsselte und signierte E-Mail. Ihr Tutor erklärt Ihnen während der Übungsstunde, an welche Adresse Sie Ihre E-Mail schicken sollen und wo Sie den entsprechenden öffentlichen Schlüssel erhalten. Dafür erhalten Sie einen Bonuspunkt.

Hausaufgabe 8

Gegeben sei die folgende Segler-Boots-Reservierung-Datenbank:

```
.decl segler(sid: number, sname: symbol, einstufung: number, alter: number)
.decl boot(bid: number, bname: symbol, farbe: symbol)
.decl reservierung(sid: number, bid: number, datum: number)
```

Beantworten Sie die folgenden Anfragen in Datalog und testen Sie unter (`http://souffle.db.in.tum.de/`, Examples => Segler-Boots-Reservierung):

1. Geben Sie die Farben aller Boote, die von 'Lubber' reserviert wurden, aus.

   ```
   .decl lubber_farbe(farbe: symbol)
   ```

2. Geben Sie alle Segler aus, die eine Einstufung von mindestens 8 oder das Boot 103 reserviert haben.

   ```
   .decl a2(sid: number, name: symbol)
   ```

3. Geben Sie die Namen aller Segler aus, die mindestens zwei Boote reserviert haben.

   ```
   .decl doppelBoot(name: symbol)
   ```

4. Geben Sie alle Segler aus, die noch nie ein rotes Boot reserviert haben.
5. Geben Sie alle Segler aus, die mehr als 20 Jahre alt sind und kein rotes Boot reserviert haben.
6. Geben Sie die Ids der Segler aus, deren Einstufung besser als die eines Seglers mit Namen 'Horatio' ist.
7. Geben Sie die Ids der Segler aus, deren Einstufung besser als die aller Segler mit Namen 'Horatio' ist.
8. Geben Sie den Namen und Alter des ältesten Seglers aus.