



Übung zur Vorlesung *Einsatz und Realisierung von Datenbanken im SoSe25*

Alice Rey, Maximilian Reif, Tobias Goetz (i3erdb@in.tum.de)

<http://db.in.tum.de/teaching/ss25/impldb/>

Blatt Nr. 04

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), $X \leq Y$, $X \geq Y$ (kleiner oder gleich, größer oder gleich), $X = Y$, $X \neq Y$ (gleich, ungleich), $!pred(X, Y)$ (existiert nicht $pred(X, Y)$).

Hausaufgabe 1

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 2

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 matnr, 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(matnr) + 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 3

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 umzubenennen, so seien sie nicht zu finden. Würde diese *Sicherheitsmaßnahme* helfen?

Hausaufgabe 4

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 Brute-force-Attacken erschweren?

Hausaufgabe 5

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.

Hausaufgabe 6

Gegeben sei die nachfolgende *KindEltern*-Ausprägung für den Stammbaum-Ausschnitt der griechischen Götter und Helden:

KindEltern		
Vater	Mutter	Kind
Zeus	Leto	Apollon
Zeus	Leto	Artemis
Kronos	Rheia	Hades
Zeus	Maia	Hermes
Koios	Phoebe	Leto
Atlas	Pleione	Maia
Kronos	Rheia	Poseidon
Kronos	Rheia	Zeus
Poseidon	Amphitrite	Triton

Formulieren Sie folgende Anfragen in Datalog und testen Sie unter (<http://souffle.db.in.tum.de/>):

- a) Bestimmen Sie alle Geschwisterpaare.

- b) Ermitteln Sie Paare von Cousins und Cousinen beliebigen Grades. Die Definition finden Sie auf Wikipedia.
- c) Geben Sie alle Verwandtschaftspaare an. Überlegen Sie sich eine geeignete Definition von Verwandtschaft und setzen Sie diese in Datalog um.
- d) Bestimmen Sie alle Nachfahren von Kronos. Formulieren Sie die Anfrage auch in SQL, so dass sie unter HyPer ausführbar ist (online testen unter: <http://hyper-db.de/interface.html>). Sie können die Daten als Common Table Expression definieren und dann nutzen:

```
WITH RECURSIVE kindEltern(vater,mutter,kind) as (
  VALUES ('Zeus', 'Leto', 'Apollon'), ('Zeus', 'Leto', 'Artemis'),
  ('Kronos', 'Rheia', 'Hades'), ('Zeus', 'Maia', 'Hermes'),
  ('Koiros', 'Phoebe', 'Leto'), ('Atlas', 'Pleione', 'Maia'),
  ('Kronos', 'Rheia', 'Poseidon'), ('Kronos', 'Rheia', 'Zeus'),
  ('Poseidon', 'Amphitrite', 'Triton')
), parent(eltern,kind) as (
  select vater, kind from kindEltern UNION select mutter, kind from kindEltern
) select * from parent where eltern='Zeus'
```

Hausaufgabe (wird nicht in der Übung besprochen)

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 (wird nicht in der Übung besprochen)

Gegeben das folgende Schema der EDB¹:

```
.decl product(maker: symbol, model: symbol, type: symbol)
.decl pc(model: symbol, speed: float, ram: number, hd: number, price: number)
.decl laptop(model: symbol, speed: float, ram: number,
             hd: number, screen: symbol, price: number)
.decl printer(model: symbol, color: symbol, type: symbol, price: number)
```

Beantworten Sie in Datalog und testen Sie unter (<http://souffle.db.in.tum.de/>):

- a) What PC models have a speed of at least 3.00 GHz?


```
.decl fast_pc(model: symbol, speed: float, price: number)
```
- b) Which manufacturers make laptops with a hard disk (hd) of at least 100 GB?


```
.decl manufacturers_100GB(maker: symbol)
```
- c) Find the model number and price of products (of any type) made by manufacturer B.


```
.decl b_prod(model: symbol, price: number)
```
- d) Find the model numbers of all color laser printers.


```
.decl color_laser_printers(model: symbol)
```
- e) Find those manufacturers that sell Laptops, but not PC's.

¹Inspiziert von http://people.inf.elte.hu/sila/DB1English/exercise06_products.pdf.

```
.decl laptop_manuf(maker: symbol)
```

f) Find those hard-disk sizes that occur in two or more PC's.

```
.decl pop_sizes(hd: number)
```

g) Find those pairs of PC models that have both the same cpu speed and RAM. A pair should be listed only once, e.g., list (i,j) but not (j,i).

```
.decl sim_pc(m1: symbol, m2: symbol)
```