Sehr geehrter Herr Professor Bott,

wie Sie von Herrn Amelung bereits erfahren haben, arbeite ich im Rahmen meiner

Bachelorarbeit ebenfalls gerade an der Umsetzung eines Backends zur Überprüfung von SQL-

Anfragen. Und wie Sie in einer ihrer Mails bereits vorgeschlagen hatten, möchte ich im

Folgenden einige Gedanken über die Anforderungen eines solchen Systems austauschen.

Ich habe mir ebenfalls bereits einige Gedanken über generelle Anforderungen,

Möglichkeiten und eventuelle Probleme gemacht und in Anlehnung an eine Ausarbeitung

von Herrn Amelung, Katrin Krieger und Professor Dietmar Rösner zum Thema

Backendentwicklung für E-Assessment zu Papier gebracht.

Im Vordergrund stehen im Allgemeinen die Fragen "Was soll getestet werden?" und "Wie

soll getestet werden?". Da erstere mit den drei Buchstaben SQL praktisch schon beantwortet

ist, geht es also hauptsächlich um die Frage, wie getestet werden soll.

Es gibt also noch einige Punkte, die im Voraus geklärt werden sollten. Anbei stehen meine

bisherigen Überlegungen. Ich würde mich über Ihre Meinung zu den einzelnen Punkten

freuen.:

**Gliederung:**

1. Testverfahren
2. Aufbau
   1. Nur eine Datenbank
   2. Nur ein Nutzer
3. Eingabedaten
   1. Eingabedaten der Lehrkraft
   2. Eingabedaten der Lernenden
4. Auswertung
   1. Syntaktische Korrektheit
   2. Semantische Korrektheit
   3. Sicherheit
5. Ausgabe

**1. Testverfahren**

Sinnvoll wäre zunächst ein dynamisches Testverfahren. Dazu gibt

es auch bereits Lösungen -> dbUnit (www.dbunit.org) ermöglicht das

Testen ganzer Datenbanken. Ich habe es zwar noch nicht getestet,

aber schonmal mit in die erste Wahl aufgenommen.

**2. Aufbau**

2.1. Nur eine Datenbank

Vor jedem einzelenen Test werden die original Tabellen aus der von

der Lehrkraft hinterlegten Aufgabenstellung in eine Datenbank geschrieben.

Nach dem Test wird diese Datenbank wieder geleert. Das hat zum einen den

Vorteil, dass nur eine einzige Datenbank benötigt wird, und zum anderen,

dass die Anfragen des Lernenden die Tabellen nicht nachhaltig zerstören

können. Es kann also auch von Anfragen wie UPDATE, DELETE oder REPLACE

gebrauch gemacht werden, die die Datenbanktabellen verändern würden.

2.2. Nur ein Benutzer für alle Lernenden

Vor jedem Test, also nachdem der Lernende auf "einreichen" geklickt hat,

schlüpft dieser in die Rolle des Benutzers (mit entsprechenden

erforderlichen Berechtigungen).

**3. Eingabedaten**

Unterschieden werden sollte generell nur zwischen den Eingabedaten

der Lehrkraft, und denen des Lernenden.

3.1. Eingabedaten der Lehrkraft

3.1.1. Die Ausgangstabellen sollten, zusammen mit der eigentlichen

Aufgabenstellung, direkt auf dem Formular, das der Lernende beim Anklicken

der Aufgabe zu sehen bekommt, angezeigt werden beziehungsweise eingeblendet

werden können (da das Wissen über alle Tabellen natürlich essentiell für

die Lösung der Aufgabe ist).

3.1.2. Außerdem soll die Lehrkraft das erwartete Ergebnis für die Datenbankanfrage

hinterlegen, damit es mit dem Ergebnis der Anfrage des Lernenden verglichen werden kann.

Gegebenenfalls kann das erwartete Ergebnis (also die Zieltabelle) bereits in der

Aufgabenstellung genannt

sein.

3.2.3. Des Weiteren sollte die Möglichkeit bestehen, bei Bedarf mehrere alternative

"erwartete Ergebnisse" zu hinterlegen. Zum Beispiel für den Fall, dass die Aufgabenstellung

leicht missverstanden werden kann oder ein Ergebnis noch als korrekt anzusehen ist, in dem

einzelne Zeilen fehlen oder zu viele vorhanden sind.

3.2. Eingabedaten des Lernenden

3.2.1. Der Lernende gibt für seine Einreichung ausschließlich SQL-Anfragen ein (mit denen er

das erwartete Ergebnis zu erhalten glaubt).

3.2.2. Kommentare sollten erlaubt sein. Zum einen ermöglicht es dem Lernenden, im Falle

eines Vortrages seiner Lösung ein schnelleres "Hineindenken", und zum anderen kann er

damit seine Ideen für Lösungen schildern, die das Backend nicht akzeptiert hat.

**4. Auswertung**

4.1. Als erstes sollte die komplette Anfrage in der Einreichung vom SQL-Interpreter auf

**syntaktische Korrektheit** überprüft werden. Schlägt dieser Test bereits fehl, wird die weitere

nächste Überprüfung (auf Korrektheit der Semantik) nicht durchgeführt.

4.2. Dann wird das Ergebnis der Anfrage des Lernenden wird mit dem von der Lehrkraft

hinterlegten erwarteten Ergebnis verglichen (**semantische Korrektheit**). Permutationen der

Tabellenzeilen müssen hierbei im Normalfall erlaubt sein, da durch ORDER BY die

Reihenfolge nach individuell anders sein kann.

4.3. Sicherheit

Da die Anfragen des Lernenden immer auf einer temporären Kopie der originalen Datenbank

durchgeführt werden, gibt es meiner Ansicht nach keine nennenswerten

Sicherheitsbedenken (Zum Beispiel durch SQL Injections). Dennoch sollte es einen Timeout

für die Auswertung geben.

**5. Ausgabe**

5.1. Schlägt die Syntaxprüfung fehl, soll nicht nur der erste Syntaxfehler, sondern auch alle

weiteren angezeigt werden. Das ist aus meiner Sicht als Student, der solche

Einreichungssysteme bereits genutzt hat, vor allem dann wichtig, wenn man für die

Einreichungen nur eine begrenzte Anzahl an Versuchen hat. In dem Fall kann man schon

aufgrund dreier kurz aufeinanderfolgender Tippfehler schnell alle Einreichungsversuche

verbrauchen.

5.2. Schlägt die Semantikprüfung fehl, so reicht es aus, wenn bei der Ausgabe

das Ergebnis der Anfrage dem erwarteten Ergebnis gegenübergestellt wird.

Eine schönere Lösung wäre es, wenn in der aus der Anfrage generierten Tabelle

falsche Zeilen rot markiert und fehlende Zeilen in gelber Farbe hinzugefügt

werden (sinnvollerweise nur für teilweise korrekte Tabellen).

Auch interessiere ich mich für Ihre Überlegungen dazu und würde mich über

konstruktive Kritik freuen.

Mit freundlichen Grüßen,

Hannes Biermann