Noname manuscript No.
(will be inserted by the editor)

Das E-Assessment-Tool DMT

Ein Werkzeug zur automatischen Bewertung und Feedback-Generierung für typische Übungsaufgaben im Fachbereich Datenbanken

Andreas Thor · Toralf Kirsten

Received: date / Accepted: date

Zusammenfassung Die Bearbeitung von Übungsaufgaben ist ein wichtiges Element in der Datenbank-Lehre. Lösungen der Studierenden lassen sich dabei häufig in strukturierten Ergebnisformaten festhalten, wie z.B. SQL-Anfragen oder die Spezifikation von Schemata und Relationen. Dieser Beitrag stellt das E-Assessment-Tool DMT (Data Management Tester) vor, das sowohl eine automatische Bewertung als auch eine automatische Feedback-Generierung solcher strukturierter Lösungen ermöglicht. Es soll dabei insbesondere Studierende unterstützen, nicht ganz korrekte Lösungen zielgerichtet überarbeiten zu können. Dieser Betrag skizziert Konzept und Architektur von DMT und erläutert den Einsatz an der HTWK Leipzig und der Hochschule Mittweida.

Schlüsselwörter E-Assessment · SQL · Datenbanklehre

1 Einführung

Die regelmäßige Bearbeitung von Übungsaufgaben durch Studierende ist ein zentrales Element für den Lernerfolg (nicht nur) in MINT-Fächern. Nach der Abgabe der Lösungen sollte eine zeitnahe Bewertung inklusive Feedback erfolgen, damit Studierende eine regelmäßige Einschätzung ihres Leistungsstandes erhalten und auf Grund eventuell gemachter Fehler ihre Lernarbeit anpassen können. Die manuelle Bewertung von Lösungen sowie die Erstellung eines Feedbacks, das u.a. angibt, welche Fehler bzw. Arten von Fehlern gemacht worden sind, resultieren in einem hohen Arbeitsaufwand für die Lehrenden insbesondere bei großen Studiengruppen.

Zur Reduktion dieser Zeitaufwände hat sich in den letzten Jahren E-Assessment etabliert, welches standardisierte Aufgabenformate anbietet, die größtenteils automatisch auswertbar sind. Gängige Lern-Management-Systeme wie Ilias[1], Moodle[2] oder OLAT[3] bieten dazu Lehrenden u.a. die Möglichkeit, Single- und Multiple-Choice-Aufgaben sowie Aufgaben zur Bestimmung von Zuordnungen oder Reihenfolgen zu definieren. Dabei wird neben Aufgabenstellung und Antwortoptionen zusätzlich die korrekte Lösung spezifiziert, welche dann für die automatische Bewertung der studentischen Lösungen genutzt wird. Zusätzlich kann ein zuvor hinterlegtes, standardisiertes Feedback angezeigt werden, welches z.B. bei der Auswahl einer falschen Antwortoption einer Multiple-Choice-Frage erläutert, warum die gewählte Option falsch ist.

E-Assessment mit den erwähnten standardisierten Aufgabenformaten lässt sich natürlich auch in der Datenbank-Lehre einsetzen. Neben dem Abfragen von Faktenwissen (Kompetenzstufe *Erinnern* in der Taxonomie nach Anderson und Krathwohl [5]) erlauben z.B. Multiple-Choice-Fragen auch die Prüfung von Kompetenzen der Taxonomiestufen *Verstehen* und *Anwenden*. Ein Beispiel wäre die Frage „Welche der folgenden Tupel sind im Ergebnis der Query `SELECT a,b FROM R WHERE c>5` enthalten?“, bei der zusätzlich eine Relation `R` angegeben wird und die Antwortoptionen aus

Andreas Thor
Hochschule für Technik, Wirtschaft und Kultur Leipzig
E-Mail: andreas.thor@htwk-leipzig.de

Toralf Kirsten
Hochschule Mittweida
E-Mail: toralf.kirsten@hs-mittweida.de

[1] https://www.ilias.de/
[2] https://moodle.de
[3] https://olat.org/

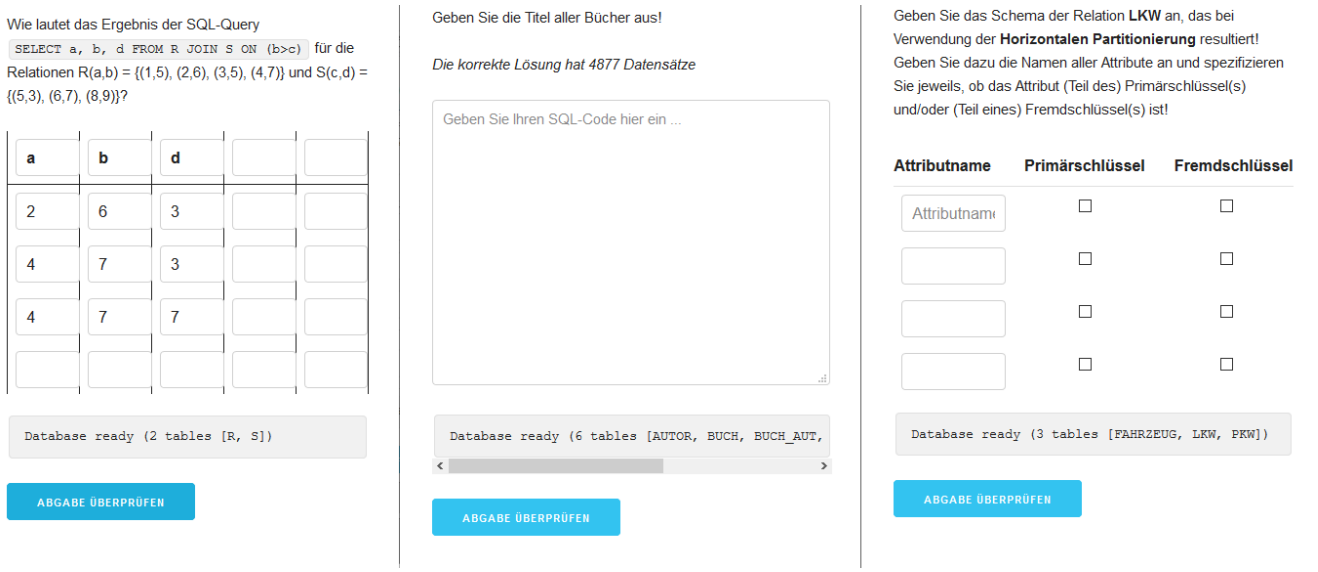

Abb. 1 Screenshot für drei verschiedene Aufgaben vom Typ TABLE (links, mit eingetragener korrekter Lösung), SELECT (mittig) und SCHEMA (rechts).

Tupeln bestehen. Damit lässt sich z.B. das Verständnis von SQL-Queries überprüfen.

Die Nutzung standardisierter Aufgabenformate für die Datenbank-Lehre hat jedoch drei gravierende Nachteile. Erstens schränkt die Spezifikation von Antwortoptionen den Lösungsraum stark ein und ermöglicht ein Raten bzw. Ausschlussverfahren seitens der Studierenden. Im Beispiel der Multiple-Choice-Frage können Studierende aus den Tupeln zufällig wählen und offensichtlich unsinnige Antwortoptionen, die z.B. Attributwerte enthalten, die in `R` nicht vorkommen, aussschließen. Zweitens ist die Bewertung gegebener Artefakte wie z.B. einer SQL-Query oder eines Relationenschemas im Allgemeinen einfacher als deren selbständige Erstellung. Im oben genannten Beispiel muss der Studierende z.B. die korrekte Reihenfolge der Schlüsselworte `SELECT`, `FROM` und `WHERE` nicht wissen, da ja bereits eine syntaktisch korrekte Query Teil der Aufgabenstellung ist. Bei der selbständigen Erstellung einer Query, z.B. „Alle Tupel (a,b) aus R, bei denen c größer als 5 ist!“, ist dieses Wissen aber notwendig. Drittens sind die Möglichkeiten für ein Feedback bei falschen Lösungen begrenzt. Bei der Beispiel-Aufgabe kann für falsche Tupel erklärt werden, warum sie nicht Teil des Anfrageergebnisses sind. Ein Feedback, was zur selbständigen Verbesserung bzw. Korrektur der studentischen Lösung führt, kann aber nicht gegeben werden, da in diesem Moment die korrekte Lösung offenbart wird.

In den folgenden Abschnitten stellen wir das E-Assessment-Tool DMT (Data Management Tester) vor. Es erlaubt die automatische Bewertung strukturierter Ergebnisformate, die häufig als Lösung von Aufgaben in der Datenbank-Lehre stehen. Die Ergebnisformate umfassen u.a. SQL-Anfragen, die Spezifikation von Schemata und Relationen sowie View- und Constraint-Definitionen. Wir zeigen an Hand eines exemplarischen Bachelor-Moduls „Datenbanksysteme“, in welchen Themengebieten Aufgaben mit den Ergebnisformaten möglich sind und wie diese durch die Lehrenden definiert werden. Anschließend erläutern wir den Bewertungs- und Feedback-Mechanismus und skizzieren die Architektur der Implementation, ehe wir einen kurzen Abriss zu zwei Einsatzszenarien von DMT in den letzten Semestern an der HTWK Leipzig und der Hochschule Mittweida geben.

2 Aufgaben und Ergebnisformate

Das E-Assessment-Tool DMT erlaubt die Spezifikation von Aufgaben durch Lehrende und ist in der Lage, Lösungen von Studierenden zu bewerten und Feedback zu generieren. Jede Aufgabe hat eine eindeutige Id und wird zunächst durch einen Titel und eine Aufgabenstellung beschrieben. In Abhängigkeit vom Ergebnisformat spezifiert der Lehrende weitere Informationen, die zur automatischen Bewertung notwendig sind. Im Folgenden werden zunächst die fünf bisher unterstützten Ergebnisformate inklusive der benötigten Informationen kurz charakterisiert. Darüber hinaus nennt Tabelle 1 für typische Themengebiete eines Datenbank-Moduls in der Hochschullehre jeweils zwei Beispielaufgaben inkl. Ergebnisformat.

– **SELECT:** Der Studierende gibt als Ergebnis eine SQL-Query an. Zur automatischen Bewertung

Tabelle 1 Beispiele für typische Aufgaben innerhalb eines Datenbank-Moduls inkl. Angabe des Ergebnisformats

Ergebnisformat	Beispiel einer Aufgabe (z.T. verkürzt)
Kapitel: Das Relationale Modell	
TABLE	Führen Sie folgendes `UPDATE`-Statement aus und geben Sie die resultierende Tabelle `R` an!
CHECK	Ergänzen Sie mittels `ALTER TABLE` eine `NOT-NULL`-Bedingung für das Attribut `a` in `R`!
Kapitel: Datenbankanfragen mit SQL	
SELECT	Erstellen Sie eine SQL-Query, die die Titel aller Bücher ausgibt!
TABLE	Wie lautet das Ergebnis der SQL-Query `SELECT a FROM R WHERE b>7` für die gegebene Relation `R`?
Kapitel: Umwandlung eines ER-Modells in das Relationale Modell	
SCHEMA	Geben Sie das Schema der Relation LKW an, welches sich bei Umwandlung des gegebenen ER-Modells unter Verwendung der horizontalen Partitionierung ergibt!
CHECK	Ergänzen Sie einen Constraint zur Relation Station, der den 1:1-Abbildungstyp der Beziehung „Arzt leitet Station“ sicherstellt.
Kapitel: Normalisierung	
SCHEMA	Geben Sie das Schema der Relation Mitarbeiter nach Umwandlung in die dritte Normalform an!
VIEW	Definieren Sie die Relation Mitarbeiter der 2NF als Sicht über die Relationen in 3NF!
Kapitel: Datenkontrolle	
CHECK	Schreiben Sie einen Trigger, so dass ein Nutzer nur maximal drei Bücher gleichzeitig ausleihen kann.
VIEW	Formulieren Sie eine Sicht, die zu jedem Leser die Anzahl der von ihm ausgeliehenen Bücher angibt.

spezifiziert der Lehrende eine SQL-Query als Musterlösung und gibt zusätzlich an, ob die Reihenfolge der Datensätze relevant ist. Abbildung 1 (mittig) zeigt das User-Interface einer Beispielaufgabe diesen Typs.

- **SCHEMA:** Der Studierende spezifiziert das vereinfachte Schema einer Relation (siehe Abbildung 1 (rechts)). Dazu gibt er die Attributnamen (ohne Datentypen) an und markiert für jedes Attribut, ob es ein Primärschlüssel- und/oder Fremdschlüssel-Attribut ist (ohne Angabe der referenzierten Relation). Zum Vergleich mit der korrekten Lösung gibt der Lehrende den Namen einer Relation an, die das gesuchte Schema hat.
- **TABLE:** Der Studierende gibt die Attributnamen und Tupel einer Relation in Tabellenform an (siehe Abbildung 1 (links)). Zum Vergleich mit der korrekten Lösung gibt der Lehrende eine SQL-Query an.
- **VIEW:** Der Studierende gibt eine View-Definition als SQL-Code an. Der Lehrende spezifiziert den View-Namen sowie als Musterlösung die SQL-Query, die zur View-Definition genutzt wird.
- **CHECK:** Der Studierende gibt SQL-Code bestehend aus DDL-Statements, wie z.B. `CREATE TRIGGER` oder `ALTER TABLE`, an. Zur Überprüfung definiert der Lehrende ein oder mehrere SQL-Statements, für die jeweils ein definiert Datenbank-spezifischer Status- bzw. Fehlercode oder eine definierte Tupel-Menge erwartet wird.

Die Spezifikation der Aufgaben erfolgt im JSON-Format. Jede Aufgabe wird durch ein JSON-Objekt mit den oben genannten Attribute spezifiziert. Aufgaben, die auf der selben Datenbank operieren, werden in einer als Repository bezeichneten JSON-Datei zusammengefasst. Diese Datei enthält neben den Aufgaben den Namen des standardmäßig zu verwendenden Datenbankschema sowie in einem extra ausgezeichneten JSON-Objekt *init* DDL-Statements zum Anlegen weiterer Tabellen als Datengrundlage aller Aufgaben des Repositories. Zusätzlich kann jede Aufgabe ein eigenes *init*-Unterobjekt definieren, der dann nur für die jeweilige Aufgabe ausgeführt wird. Damit ist es möglich, sowohl mehrfach verwendete Daten in einem gemeinsamen Schema zu verwenden (wie z.B. die Mondial-Datenbank[4]) als auch Aufgaben-spezifische Daten zu nutzen, wie z.B. bei der ersten Aufgabe in Abbildung 1 mit den Relationen R und S. Zur Unterstützung für die Studierenden zeigt das User Interface – neben der eigentlichen Aufgabenstellung – die verfügbaren Tabellennamen zusätzlich an.

3 Automatische Bewertung und Generierung von Feedback

Wesentlicher Bestandteil von DMT ist die automatische Bewertung sowie die automatische Generierung von Feedback bzgl. der eingereichten Lösung der Studierenden. Aufgaben der Formate **SELECT**, **TABLE** und **VIEW** werden dabei in den folgenden drei Schritten verarbeitet.

[4] https://www.dbis.informatik.uni-goettingen.de/Mondial/

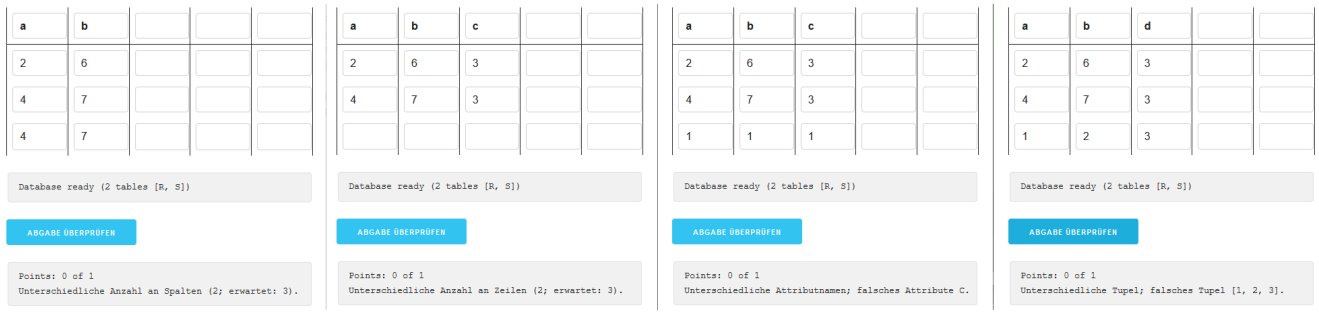

Abb. 2 Screenshot für vier verschiedene Arten von Feedback für die Aufgaben vom Typ **TABLE** aus Abbildung 1(links).

1. Umwandlung der Studierendenlösung in eine temporäre Relation: Dazu wird beim Aufgabentyp **SELECT** die eingegebene Query ausgeführt und bei **TABLE** die eingebenen Relation geparst. Für den Aufgabentyp **VIEW** wird zunächst die vom Studierenden angegebene View-Definition ausgeführt und anschließend das Anfrageergebnis `SELECT * FROM ViewName` verwendet.
2. Umwandlung der Musterlösung in eine temporäre Relation durch Ausführen der als Musterlösung hinterlegten SQL-Query.
3. Vergleich der Relationen ohne Berücksichtigung der Reihenfolge, es sei denn, beim Aufgabentyp **SELECT** wurde die Relevanz der Reihenfolge markiert, da z.B. die Anfrage ein `ORDER BY` benötigt.

An Hand der Prüfung kann zunächst eine Bewertung vorgenommen werden, ob die Lösung *korrekt* oder *nicht korrekt* ist. Das vorgestelle Bewertungsverfahren ist natürlich insbesondere für die Aufgabenformate **SELECT** und **VIEW** formal nicht korrekt, da die Prüfung nicht an Hand der Query bzw. View-Definition erfolgt, sondern am Ergebnis bzgl. eines Datenbankzustandes. Erfahrungsgemäß spielt das aber in der Praxis kaum eine Rolle, da bei hinreichend großen Datenmengen fehlerhafte Queries meist auch zu unterschiedlichen Anfrageergebnissen führen. Ausnahmen bilden Anfragen, die eine leere Menge oder skalare Werte liefern, wie z.B. `SELECT COUNT(*) FROM T`. Hier würde z.B. eine Query `SELECT 42` als korrekt erkannt, falls `T` 42 Tupel enthält.

Der Fokus von DMT liegt aber nicht auf der Bewertung sondern primär auf der Generierung individellen Feedbacks, damit Studierende animiert werden, auch bei einer falschen Lösung an der Aufgabe weiterzuarbeiten. Das Feedback resultiert aus einer strukturierten Fehleranalyse, bei der die nachfolgenden Schritte in der angegebenen Reihenfolge bearbeitet werden, wobei das Verfahren beim ersten erkannten Fehler abbricht und das generierte Feedback zurückliefert.

1. Syntax-Fehler: Die eingegebene Lösung konnte nicht ausgeführt werden und lieferte einen Syntaxfehler. Fehlermeldung und Fehlercode werden als Feedback zurückgemeldet.
2. Grad und Kardinalität: Die zu vergleichenden Relationen unterscheiden sich in Grad und/oder Kardinalität. Als Feedback werden die erwarteten Werte für Grad und Kardinalität zurückgegeben.
3. Schema: Die Relationen unterscheiden sich im Schema, d.h. die Menge der Attributnamen ist nicht äquivalent. Ein falscher Attributname wird als Feedback ausgegeben.
4. Tupel: Die Relationen unterscheiden sich in ihrer Tupel-Menge. Analog zum Schema-Feedback wird hier als Feedback ein falsches Tupel zurückgegeben.
5. Reihenfolge: Die Relationen sind bzgl. ihrer Tupel-Menge identisch, unterscheiden sich aber in ihrer Reihenfolge. Dem Studierenden wird dieser Hinweis als Feedback zurückgegeben. Dieser Schritt wird nur ausgeführt, wenn die Relevanz der Tupel-Reihenfolge in der Aufgabe spezifiziert wurde.

Insbesondere aus den Fehlermeldungen der Schritte 2 bis 4 lassen sich für den Studierenden Hinweise ziehen, um das Ergebnis zu korrigieren. Einfache Fehler, wie z.B. ein vergessenes oder falsch geschriebenes Attribut im `SELECT`, lassen sich so schnell korrigieren. Durch den konkreten Hinweis auf falsche Attribute bzw. falsche Tupel erhalten die Studierenden insbesondere ein *personalisiertes Feedback* bzgl. ihrer Lösung und sind tendenziell eher motiviert, im Selbststudium ihre Lösung zu überarbeiten. Natürlich besteht dabei auch die Möglichkeit, die richtige Lösung durch schrittweises Ausprobieren „abzufragen“ und sich der korrekten Lösung „anzunähern“. Diese Möglichkeit wird aber bewusst in Kauf genommen und wird erfahrungsgemäß von den Studierenden kaum ausgenutzt.

Bewertung und Feedbackgenerierung für das Aufgabenformat **SCHEMA** verläuft ähnlich zu den oben skizzierten Verfahren. Die Musterlösung wird durch die Angabe eines Tabellennamens definiert, mit dessen Hilfe dann das Schema (Attributname, Primär und Fremdschlüssel) aus der Datenbank ausgelesen werden. Eine Prüfung auf weitere Constraints wie z.B. Datentypen,

`NOT NULL` oder `UNIQUE` erfolgt nicht, wäre aber prinzipiell denkbar. Die Feedbackgenerierung erfolgt auch hier wieder abgestuft: zunächst wird die Anzahl der Attribute geprüft, danach die Attributnamen und dann pro Attribut die Angaben zum Primär- bzw. Fremdschlüssel. An jedem Schritt kann das entsprechende Feedback generiert werden, wie z.B. „Falsches Attribut X“ oder „Falsche Primärschlüsselangabe bei Attribut Y“.

Bei Aufgaben des Ergebnisformats **CHECK** werden innerhalb der Aufgabe entsprechende Testfälle angegeben. Diese bestehen aus SQL-Statements, die entweder ein definiertes Query-Ergebnis zurückliefern müssen (Feedback analog zu **SELECT**) oder einen entsprechenden Status- bzw. Fehlercode liefern müssen. Bei letzterem ist das Feedback dann der erwartete Code.

4 Implementation und Einsatzszenarien

Die aktuelle DMT-Implementation[5] ist eine Webservice-basierte Webanwendung unter Verwendung von Java-Servlets, HTML und Javascript. Die Funktionalität wird durch zwei REST-basierte Webservices realisiert:

- `/gettaskinfo?taskid=...` liefert alle zum Anzeigen der Aufgabe notwendigen Informationen (u.a. Aufgabenstellung und Ergebnisformat) sowie den Status (u.a. verfügbare Tabellen) als JSON-Objekt
- `/gettaskresult?taskid=...&answer=....` liefert Bewertung und Feedback bzgl. der übergebenen Antwort ebenfalls als JSON-Objekt

Die Architektur auf Basis der REST-Schnittstellen ermöglicht eine einfache Integration in verschiedene User Interfaces. In der aktuellen Implementation ist das eine HTML-Seite, die per Javascript mit den Webservices kommuniziert und mittels JQuery dynamisch die Darstellung manipuliert.

DMT wurde in den letzten Semestern von den Autoren an der Hochschule für Technik, Wirtschaft und Kultur (HTWK) Leipzig und der Hochschule Mittweida (HSMW) jeweils in der Datenbank-Lehre in den Bachelor-Studiengängen eingesetzt. Zielgruppe des Einsatzszenarios an der HTWK Leipzig waren Studierende der Telekommunikationsinformatik, die DMT als Werkzeug zur Überprüfung von Übungsaufgaben einsetzen. Dabei wurden Lösungen wie z.B. SQL-Queries in eigenen Werkzeugen oder auf Papier entwickelt und dann für die Bewertung und Feedbackgenerierung in das DMT-System übertragen. Sowohl die manuelle Analyse der anonymisierten Logfiles als auch die Gespräche mit den Studierenden zeigten, dass sie das unmittelbar verfügbare Feedback wertschätzten und zur Überarbeitung ihrer Lösungen verwendeten. Teilweise zeigten die Studierenden sportlichen Ehrgeiz, um unbedingt ein „Lösung ist korrekt“ von DMT zu erhalten. In wenigen Ausnahmefällen zeigte sich auch studentischer Ehrgeiz, das System mit den bereits oben beschriebenen Ansätzen auszutricksen. Dieses Beobachtung deckt sich mit den Erfahrungen anderer E-Assessment=SSysteme zu automatischen Auswertung von Programm-Code [9].

Demgegenüber wurde DMT an der HS Mittweida vorrangig von Studierenden in Nicht-Informatik-Studiengängen eingesetzt, die SQL als erste „Informatik-Sprache“ erlernen. Die Verwendung mächtiger SQL-Clients bzw. anderweitiger Werkzeuge ist oftmals zu komplex für diese Zielgruppe. Daher entwickelten diese Studierende unter Anleitung SQL-Anfragen direkt mit DMT und profitierten bei syntaktisch korrekten Anfragen von dem Feedback, was zum besseren Verständnis sowie zum schrittweisen Aufbau der Anfrage genutzt wurde.

[5] https://github.com/andreas-thor/dasp-dmt

5 Verwandte Arbeiten

In den letzten Jahren wurden einige Systeme zur automatischen Bewertung studentischer Lösungen insbesondere in der Mathematik- und Programmierausbildung entwickelt, die u.a. in dem seit 2013 durchegführten Workshop „Automatische Bewertung von Programmieraufgaben (ABP)“ [10] publiziert werden. Dazu zählen beispielsweise die Kontrolle mathematischer Beweise und die Verifikation von Quellcode in verschiedenen Programmiersprachen (z.B. Java). Freie Systeme, die solche Funktionalitäten anbieten umfassen u.a. EASy [8], MathX[3] [6] oder den an der Universität Passau entwickelten Praktomaten.

Lerning-Management-Systeme wie z.B. Ilias, Moodle oder OLAT ermöglichen häufig über einen Plugin-Schnittstelle Möglichkeit zu Erweiterungen. Exemplarisch sei für Moodle auf [2] für eine große Auswahl an Plugins im Bereich E-Assessment verwiesen. Weitere Linked-in Lösungen für diese Plattformen wie aSQLg [12, 7] bieten die Möglichkeit, Aufgaben mit thematischen Bezug zu Datenbanksystemen und insbesondere zur Anfragesprache SQL zu erstellen und die selbstständig von Studierenden eingestellten Lösungen automatisch nach einer vorgebenen Konfiguration bewerten zu lassen.

Oftmals liegt der Schwerpunkt dieser E-Assessment-Systeme in der Bewertung der von Studierenden spezifizierten Ergebnisse einer Aufgabe, um z.B. in Zusammenhang mit einem Aufgabenblatt oder einer Prüfung eine übergreifende Bewertung der Leistung und ggf. eine Note automatisiert abzuleiten. Darüber hinaus erlauben einige Systeme die Erstellung parametrisierter

Aufgaben, so dass Studierende personalisierte Aufgaben bekommen. Im Gegensatz zu DMT kommt die Rückmeldung von personalisiertem Feedback an die Studierenden meist zu kurz.

Im Bereich der Datenbank-Lehre gibt es noch eine Vielzahl weiterer Werkzeuge, die das Lernen der Studierenden unterstützen. Einen Learning-by-Doing Ansatz verfolgt SQL Islands [11]. Ähnlich wie DMT geht es in dem Tool um das individuelle Feedback an die Studierenden. Dazu werden die Aufgaben jedoch in einer Geschichte „gamifiziert". Analog zu den SELECT-Aufgaben bei DMT agiert das Tool SQLcoach [4]. Studierende sind aufgefordert, SQL-Anfragen für ein gegebenes Schema zu erstellen. Bei falschen Antworten zeigt SQLcoach neben dem Ergebnis der (falschen) SQL-Anfrage eine Preview (z.B. einen Datensatz) des korrekten Ergebnisses. Das eLearning Portal der TH Köln [1] ist ein Toolset und bietet neben Multiple-Choice Fragen dem Studierenden die Möglichkeit mit speziellen Trainern zu arbeiten, die jeweils einen konkreten Bereich abdecken. Ein SQL Trainer dient dem Erlernen der Anfragesprache SQL, der ER-Trainer bedient die Modellierung mit ER-Diagrammen und der 3NF-Trainer den Bereich der Normalisierung (insbesondere 3NF). Ein SQL Puzzle vermittelt spielerisch Kenntnisse in der Anfragesprache SQL. Speziell für die relationale Algebra wurde an der Universität Innsbruck mit RelaX [3] eine Umgebung zum Entwickeln und Ausführen von Ausdrücken der relationalen Alegbra entwickelt.

6 Zusammenfassung und Ausblick

Dieser Artiekl stellt das E-Assessment-Tool DMT (Data Management Tester) vor, das in der Datenbank-Lehre sowohl in Praktika als auch während des Selbststudiums eingesetzt werden kann. DMT unterstützt dazu fünf verschiedene Ergebnisformate mit deren Hilfe eine große Bandbreite typischer Aufgaben im Datenbankbereich abgedeckt werden kann. Studierende erhalten für ihre Lösungen eine automatische Bewertung sowie ein Feedback, welches konkrete Hinweise bzgl. ihrer fehlerhaften Lösung gibt. Damit verfolgt DMT das Ziel, Studierende in ihrem individuellen Lernprozess durch zielgerichtetes Feedback zu animieren, ihre Lösungen kontinuierlich zu verbessern.

Zukünftig sollen weitere Datenmodelle (JSON, XML, Graph) sowie korrespondierende Anfragesprachen wie (u.a. XPath und XQuery) in entsprechenden Ergebnisformaten adressiert werden. Ebenso ist die Unterstützung der LTI-Schnittstelle (Learning Tools Interoperability) geplant, um DMT in bestehende Lernplattformen integrieren zu können. Zusätzlich wollen die Autoren in den kommenden Semestern Evaluierungen gemeinsam mit den Studierenden vornehmen.

Literatur

1. eLearning Datenbank Portal der TH-Köln. URL https://edb2.gm.th-koeln.de
2. Liste der Moodle-Plugins im Bereich E-Assessment. URL https://moodle.org/plugins/index.php?q=assessment
3. RelaX -Rrelational Algebra Calculator. URL https://dbis-uibk.github.io/relax/landing
4. SQLcoach der Hochschule Kaiserslautern. URL https://sqlcoach.informatik.hs-kl.de
5. Anderson, L.W., Krathwohl, D.R. (eds.): A Taxonomy for Learning, Teaching, and Assessing. A Revision of Bloom's Taxonomy of Educational Objectives. Allyn & Bacon (2001)
6. Derr, K., Fried, T., Hornung, B., Saller, D.: Mathx^3 – online-selbsttest zur basiskompetenz mathematik. Zeitschrift für Hochschulentwicklung **4**(1) (2009)
7. Garmann, R., Fricke, P., Reiser, P., Bersuch, C., Bott, O.J.: Moodle, grappa, asqlg, graja - neue entwicklungen bei der grading-software der hochschule hannover(new developments in the grading software of the hanover university of applied sciences and arts). In: S. Strickroth, O. Müller, M. Striewe (eds.) Proceedings of the Third Workshop Äutomatische Bewertung von Programmieraufgaben"(ABP 2017) (2017)
8. Gruttmann, S.J., Böhm, D., Kuchen, H.: E-assessment of mathematical proofs: Chances and challenges for students and tutors. In: International Conference on Computer Science and Software Engineering, CSSE 2008 (2008)
9. Kratzke, N.: Smart like a fox: How clever students trick dumb automated programming assignment assessment systems. In: H. Lane, S. Zvacek, J. Uhomoibhi (eds.) Proceedings of the 11th International Conference on Computer Supported Education, CSEDU 2019, pp. 15–26 (2019)
10. Priss, U., Striewe, M. (eds.): Proceedings of the First Workshop Äutomatische Bewertung von Programmieraufgaben"(ABP 2013), *CEUR Workshop Proceedings*, vol. 1067 (2013)
11. Schildgen, J., Deßloch, S.: SQL-Grundlagen spielend lernen mit dem Text-Adventure SQL Island . In: 16. GI-Fachtagung Datenbanksysteme für Business, Technologie und Web (BTW 2015), Demo-Programm (2015)
12. Stöcker, A., Becker, S., Garmann, R., Heine, F., Kleiner, C., Bott, O.J.: Evaluation automatisierter Programmbewertung bei der Vermittlung der Sprachen Java und SQL mit den Gradern aSQLg und Graja aus studentischer Perspektive. In: A. Breiter, C. Rensing (eds.) DeLFI 2013 - Die 11. E-Learning Fachtagung Informatik der Gesellschaft für Informatik e.V. (GI), LNI (2013)