XV JENUI. Barcelona, 8-10 de julio de 2009
ISBN: 978-84-692-2758-9
http://jenui2009.fib.upc.edu/

LEARN-SQL: Herramienta de gestión de ejercicios de SQL con autocorrección

A. Abelló X. Burgués M. Casañ C. Martín C. Quer T. Urpí E. Rodríguez
U. Politècnica de Catalunya U. Oberta de Catalunya
{aabello|diafebus|mjcasany|martin|cquer|urpi}@lsi.upc.edu mrodriguezgo@uoc.edu

Resumen

Algunas herramientas de autocorrección existen ya en el ámbito de la docencia informática. No obstante en asignaturas de bases de datos el problema es especialmente complejo debido a la gran variedad de tipos de ejercicios (los sistemas existentes se limitan a consultas) y a que éstos no tienen solución única. Nuestro sistema tiene como objetivo corregir automáticamente cualquier tipo de sentencia SQL (consultas, actualizaciones, procedimientos, disparadores, creación de índices, etc.) y discernir si la respuesta aportada por el estudiante es o no correcta con independencia de la solución concreta que éste proponga. En esta comunicación presentaremos específicamente el módulo encargado de la gestión de ejercicios y todas las tipologías de estos que estamos utilizando en la actualidad.

1. Introducción

El uso de las TIC en educación nos da la oportunidad de revisar y desarrollar nuevas prácticas y métodos. Muchas de las herramientas dan la oportunidad de acceder al conocimiento en cualquier momento y lugar, facilitan la obtención de retroalimentación de forma inmediata, ayudan a desarrollar una actitud proactiva, etc. Estas herramientas también permiten a los profesores pasar de ser un proveedor de contenidos a un facilitador de contenidos, mediante la creación, compartición y reutilización de entidades de aprendizaje digital. En este trabajo, presentamos un sistema que sigue la especificación arquitectónica de IMS QTI en [1] y permite la evaluación de las habilidades con SQL de los estudiantes de una forma automática e interactiva.

Los sistemas de corrección automática no son nuevos en informática. Existen varios en el área de programación (véase [4] para una discusión detallada). Sin embargo, en el ámbito de las Bases de Datos (BD), aparecen dificultades añadidas: en primer lugar, la variedad y diversidad en la tipología de ejercicios conlleva también una diversidad en los sistemas de corrección, pero además, la complejidad de los Sistemas de Gestión de Bases de Datos (SGBD) no hace más que agravar el problema. No obstante, existen también algunos sistemas de autocorrección de SQL. [3, 7, 8, 9] son algunos ejemplos. Por otro lado, [5, 6] son sistemas de tutoría de SQL. Sin embargo, la principal limitación de todos estos sistemas es que se restringen únicamente a consultas.

```
UPDATE DEPARTMENTS
SET BUDGET=0.1*BUDGET
WHERE #DEPT NOT IN
     (SELECT #DEPT FROM EMPLOYEES);
vs
UPDATE DEPARTAMENTS D
SET D.BUDGET=D.BUDGET-D.BUDGET*0.9
WHERE NOT EXISTS
     (SELECT * FROM EMPLOYEES E
     WHERE E.#DEPT=D.#DEPT);
```

Figura 1: Ejemplo de consultas equivalentes

Cuando intentamos automatizar la corrección de ejercicios de SQL, la solución, en general, no es única. Es más, el número de soluciones posibles crece muy rápidamente con la complejidad del ejercicio. Por ejemplo, las consultas en la figura 1 son semánticamente

equivalentes (ambas disminuyen en un 90 % el presupuesto de los departamentos que no tienen empleados asignados), a pesar de ser completamente diferentes desde el punto de vista sintáctico. Por lo tanto, una corrección basada en la comparación de cadenas de caracteres entre la solución del estudiante y todas las soluciones posibles no parece ni eficiente, ni siquiera factible en la mayoría de los casos.

Así pues, necesitamos implementar una estrategia que permita evaluar la corrección de una solución de forma objetiva. Esta estrategia depende del tipo de ejercicio (por ejemplo, consultas y modificaciones de la BD tendrán estrategias diferentes). En el caso del UPDATE anterior, necesitamos aplicar a la solución del estudiante una serie de juegos de pruebas representando diferentes estados de la BD, para las tablas DEPARTMENT y EMPLOYEES. Cada uno de estos juegos de pruebas comprobará si el estudiante ha cometido un cierto error o no, simplemente comparando la salida de la sentencia (que en este caso es simplemente el número de tuplas modificadas). No obstante, esto no es suficiente (por ejemplo, la solución del estudiante, podría haber rebajado el presupuesto únicamente en un 10 %), en un segundo paso, ejecutaremos las consultas necesarias para comprobar que la modificación de los presupuestos sea correcta de los departamentos correctos y en el porcentaje adecuado.

El resto del trabajo se estructura como sigue: la sección 2 muestra la arquitectura general del sistema; la 3 focaliza en la herramienta de gestión de ejercicios; la 4 muestra los diferentes tipos de ejercicios que el sistema es capaz de evaluar; la 5 contiene el resultado de las encuestas realizadas; y finalmente la 6 presenta las conclusiones y el trabajo futuro.

2. Arquitectura

Tal como se puede ver en la figura 2, presentada en [2], IMS QTI trabaja con ejercicios (*assessmentItems*) y cuestionarios (*assessmentTests*). Más concretamente, propone una arquitectura que consiste en un repositorio de ejercicios (*itemBank*) gestionado por *itemBankManager* que guarda los ejercicios

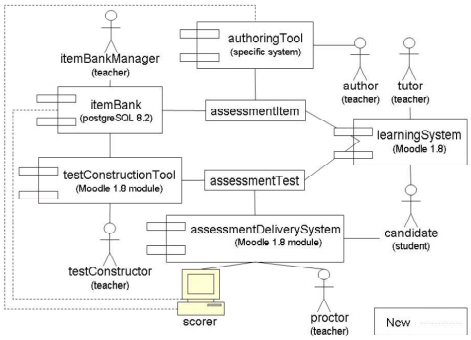

Figura 2: Arquitectura QTI extendida

(de SQL en nuestro caso) que podrán ser incluidos y reutilizados en diferentes cuestionarios en un cierto sistema de aprendizaje (*learningSystem*). También se define en la arquitectura una *authoringTool* para que los profesores gestionen los ejercicios y una *testConstructionTool* para gestionar los cuestionarios. Los profesores propondrán también materiales en el *learningSystem* para los estudiantes, que pueden responder los cuestionarios a través del *assessmentDeliverySystem*.

Hay un elemento más en esta arquitectura: el *scorer*. En [1], se define como una persona o sistema externo responsable del proceso de corrección de las respuestas de los estudiantes. Como nuestro objetivo es automatizar el proceso, lo hemos implementado con un servicio web. Este tipo de implementación facilita por sí sola la independencia y posible distribución geográfica de los componentes.

En nuestro caso, necesitamos extender levemente esa arquitectura, porque el *assessmentDeliverySystem* pide al *scorer* que corrija las respuestas de los estudiantes, pero también la *authoringTool* realizará llamada al servicio web para generar las salidas correctas para la solución del profesor (ver sección 3.3). Además, el *scorer* necesita acceder al repositorio de ejercicios.

3. Gestor de ejercicios

El *authoringTool* (AT) permite a los profesores la creación y gestión de ejercicios y sus

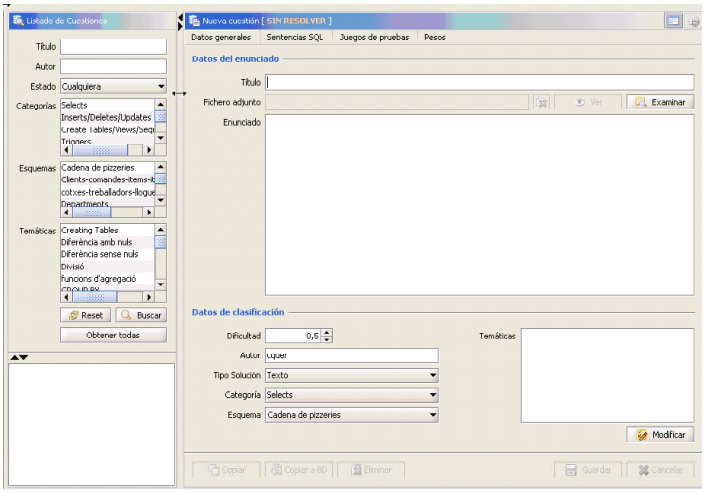

Figura 3: Datos generales de un ejercicio

soluciones. Cada uno de estos ejercicios está relacionado con un cierto esquema de BD y corresponde a un cierto tipo de ejercicio predefinido (ver sección 3.1). Además, para facilitar su gestión, los ejercicios pueden tener asociada una o más temáticas. AT no provee mecanismos para generar ejercicios automáticamente o corregir posibles errores del profesor, pero sí que ayuda a detectarlos al realizar llamadas al *scorer* para generar las salidas para la solución introducida por el profesor (ver sección 3.3).

Para cada ejercicio, el profesor debe definir una serie de juegos de prueba cuya salida indicará la corrección del resultado. La salida para estos juegos de prueba, dada la solución del estudiante, se compara con la salida para la solución introducida en el sistema por el profesor. En caso de coincidir, se considera que el alumno pasa el juego de pruebas y obtiene la puntuación correspondiente. En caso de no coincidir, el alumno recibe un mensaje (introducido por el profesor) indicando un posible error que conllevaría la no consecución de este juego de pruebas.

Sin embargo, tal como se ha explicado anteriormente, esto no es suficiente en algunos tipos de ejercicios que además de generar una salida modifican el estado de la BD. Es por eso que el profesor debe introducir también una serie de comprobaciones (consultas sobre la BD), para las cuales también debe coincidir la salida durante la ejecución del estudiante, con la que se obtiene durante la ejecución del profesor.

En la página principal de AT, existen dos marcos diferentes (ver figura 3). En la izquierda existen una serie de campos que permiten realizar la búsqueda de ejercicios por diferentes criterios como, por ejemplo, el título, autor, etc. En la derecha, existen diferentes pestañas y botones para la gestión del ejercicio seleccionado. Concretamente, en la esquina superior derecha se pueden apreciar dos botones que permiten cambiar de la pantalla de introducción de datos a la pantalla de generación de salidas de los juegos de pruebas para la solución introducida por el profesor (ver sección 3.3).

En la pantalla de introducción de datos, existen cuatro pestañas diferentes. La primera de ellas permite introducir los datos generales del ejercicio (ver sección 3.1). La siguiente pestaña facilita la introducción de las sentencias SQL asociadas al ejercicio en sí (ver sección 3.1). Las dos últimas permiten introducir los datos asociados a cada uno de los juegos de pruebas, así como el peso que se le asigna dentro del ejercicio (ver sección 3.2). Dentro de la pestaña dedicada a los juegos de pruebas, encontraremos también las diferentes comprobaciones asociadas a cada uno de ellos.

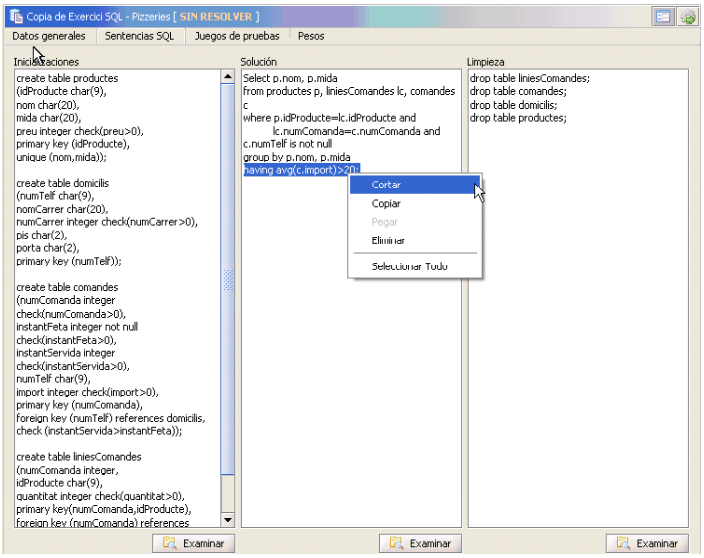

Figura 4: Sentencias SQL de un ejercicio

3.1. Datos generales

Tal como se puede ver en el marco de la derecha de la figura 3, en esta pestaña se introducen, visualizan y modifican los atributos generales de cada ejercicio. Éstos son:

Título: identifica el ejercicio.

Fichero adjunto: contiene información extra que será ofrecida a los estudiantes (habitualmente, el esquema lógico o conceptual al que hace referencia el ejercicio).

Enunciado: indica la tarea a realizar por el estudiante.

Dificultad: intenta estimar la dificultad del ejercicio.

Autor: indica quién generó el ejercicio.

Tipo de respuesta: puede ser tipo texto o la URL de la BD donde el alumno ha realizado el ejercicio.

Temáticas: contiene una lista de conceptos relacionados para su catalogación.

En la siguiente pestaña, que se puede ver en la figura 4, se introducen (es posible cortar y pegar o importar un fichero) y visualizan las sentencias SQL que se asocian al ejercicio propiamente:

Inicializaciones: sentencias que se ejecutarán previamente a la ejecución de los juegos de pruebas (normalmente, corresponde a la creación de las tablas).

Solución: respuesta del ejercicio propuesta por el profesor.

Limpieza: sentencias que se ejecutaran posteriormente a la ejecución de los juegos de pruebas para dejar la BD lista para la próxima ejecución (usualmente, corresponde al borrado de tablas).

3.2. Juegos de pruebas

El *scorer* necesita los juegos de pruebas para corregir los ejercicios. Estos juegos de pruebas corresponden a diferentes estados de la BD que están asociados a diferentes circunstancias en las cuales la solución puede ser ejecutada (es decir, posibles errores que el estudiante podría cometer en la resolución). Estos experimentos permiten obtener la salida generada por el SGBD en cada caso y compararla con la

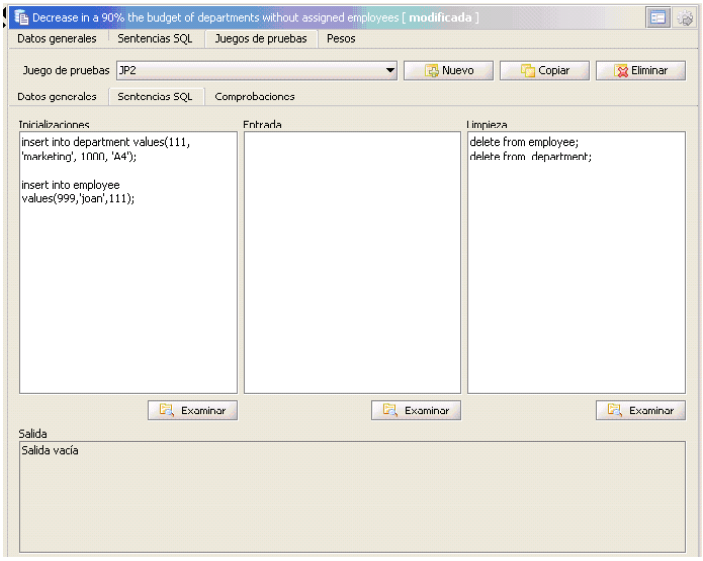

Figura 5: Juegos de pruebas

salida generada por el mismo SGBD, en las mismas circunstancias cuando se ejecuta la solución del profesor. En la pestaña "Pesos" del marco correspondiente a los ejercicios, se listan todos los juegos de pruebas definidos y se les puede asociar un peso diferente a cada uno de ellos (la suma de los pesos de todos los juegos de pruebas ha de ser siempre 1).

En el marco asociado a los juegos de pruebas, existen tres pestañas: "Datos generales", "Sentencias SQL" y "Comprobaciones". Para cada juego de pruebas, se mantienen una serie de datos (que se introducen en la primera de las tres pestañas):

Nombre: identificador del juego de pruebas.

Descripción: texto explicativo del juego de pruebas.

Mensaje: texto asociado, que se le mostrará al estudiante en caso de no coincidir su salida con la del profesor.

Además, tal como se puede ver en la figura 5, a semejanza del ejercicio, cada juego de pruebas contiene unas sentencias SQL asociadas (que se introducen en la segunda pestaña):

Inicializaciones: sentencias que se ejecutaran previamente a la ejecución del juego de pruebas (típicamente, corresponde a la inserción de tuplas) y establecen el estado en el cuál se ejecutará la solución.

Entrada: sentencias para ejecutar y parametrizar el código introducido en la solución, por ejemplo una llamada en caso de procedimientos almacenados (nótese que no se usa en todos los tipos de ejercicios).

Limpieza: sentencias que se ejecutaran posteriormente a la ejecución del juego de pruebas para dejar la BD lista para la ejecución del siguiente juego de pruebas (típicamente, corresponde al borrado de tuplas).

En la parte inferior de la figura 5 se puede ver un campo de texto no editable que muestra cuál es la salida para ese juego de pruebas, dada la solución del profesor.

La tercera de las pestañas del marco de los juegos de pruebas contiene un nuevo marco con las pestañas correspondientes a las comprobaciones: "Datos generales" y "Sentencias SQL". Una comprobación consiste en una consulta que comprueba el estado de la BD pos-

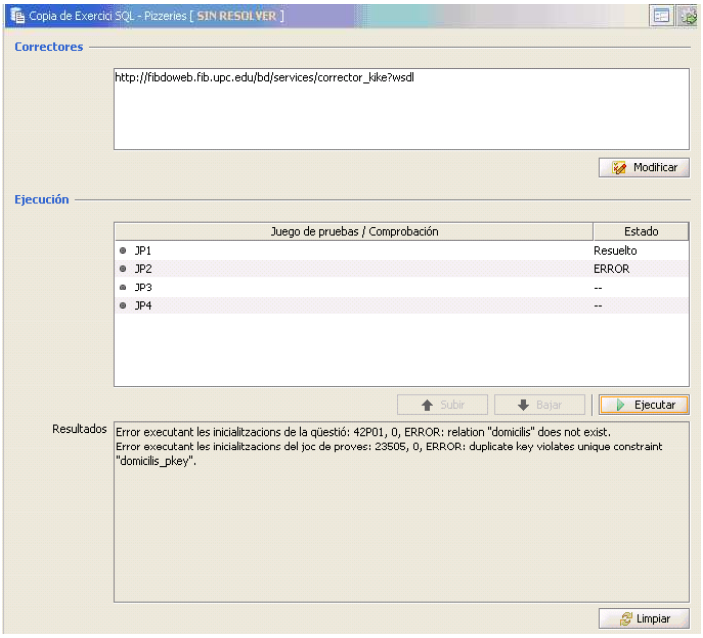

Figura 6: Generación de resultados

teriormente a la ejecución de la solución o entrada y anteriormente a la limpieza del juego de pruebas. A semejanza de los juegos de pruebas, en la primera de esas dos pestañas, cada comprobación tiene asociado un nombre, descripción y mensaje de error que se le mostrará al estudiante en caso de no coincidir su salida con la del profesor. La segunda pestaña contiene únicamente un campo de texto para introducir una consulta y un segundo campo no editable donde aparece la salida de esa consulta para la solución del profesor.

3.3. Generación de salidas

Independientemente de la pestaña o marco que estemos visualizando en cada momento, en la cabecera de la ventana, siempre aparece el título del ejercicio con una etiqueta al lado que indica su estado. Este estado puede ser cualquiera de los siguientes:

1. "Modificada", los datos se han modificado y no se han guardado aún.

2. "No resuelta", los datos en pantalla coinciden con los almacenados, pero las salidas de los juegos de pruebas no se han generado.

3. "Resuelta", el ejercicio está listo para ser utilizado en un cuestionario.

La pantalla que vemos en la figura 6 permite pasar del estado "No resuelta" a "Resuelta". Es decir, permite gestionar la generación de las salidas de los juegos de pruebas y las comprobaciones. Cabe decir sin embargo que no es AT sino el propio *scorer* el encargado de generar siempre dichas salidas, para garantizar el mismo entorno de ejecución tanto para el estudiante como para el profesor. Así pues, en la parte superior de la pantalla tenemos la lista de URLs donde se pueden encontrar los *scorer*s asociados al ejercicio. La solución del profesor se ejecutará en todos y cada uno de ellos para garantizar que siempre se obtienen exactamente las mismas salidas (así evitamos posibles diferencias de versiones o configuración entre ellos).

El campo inmediatamete debajo de ése, contiene la lista de juegos de pruebas y comproba-

ciones (en caso de existir). La ejecución de estos juegos de pruebas puede ser ordenada mediante los dos botones "Subir" y "Bajar". Una vez hecho esto, pulsando el tercer botón ("Ejecutar") generaremos finalmente las salidas de cada uno de ellos y pasaremos al estado "Resuelta". Cualquier modificación en las sentencias SQL o el orden de los juegos de pruebas, devuelve el estado del ejercicio a "No resuelta" y es necesario volver a ejecutar.

También el campo de texto inferior muestra el resultado de la ejecución, indicando si ésta ha sido correcta, o si se ha producido algún error. Si se produce un error en alguno de los juegos de pruebas o comprobaciones, quedará así reflejado junto al identificador del mismo.

Una vez definido el ejercicio y obtenidas todas la salidas para la ejecución del profesor, aquél está listo para su inclusión en un cuestionario. La corrección del ejercicio se hará simplemente comparando las salidas obtenidas para el profesor y el estudiante.

4. Tipología de ejercicios

Con la misma herramienta de gestión de ejercicios, se puede proponer a los estudiantes una gran variedad de ejercicios, simplemente variando el contenido de los diferentes campos de los juegos de pruebas y comprobaciones. Debido a la falta de espacio, no explicitaremos qué se debe introducir en cada uno de los casos, sino que simplemente listaremos los tipos de ejercicios que hemos utilizado hasta el momento en nuestras asignaturas:

Select Realizar una cierta consulta sobre un esquema dado.

Insert/Update/Delete Modificar los datos de una BD dada. Existen dos variantes:

1. Encaminados a alcanzar un estado (se trata de practicar cómo conseguir la transición de un cierto estado de la BD a otro cumpliendo las restricciones de integridad).
2. Encaminados a violar una restricción de integridad (se trata de practicar la violación de restricciones de integridad).

Create table Crear tablas en la BD. Existen dos variantes:

1. Con coincidencia de nombres de tablas y atributos (el punto de partida es un esquema conceptual dado, relativamente pequeño).
2. Sin coincidencia de nombres de tablas y atributos (el punto de partida es un esquema conceptual relativamente grande, donde es fácil cometer errores de sintaxis, o un esquema lógico no normalizado, donde el estudiante deberá generar nuevas tablas).

Create view Crear vistas sobre un esquema dado.

Create procedure Crear procedimientos almacenados dado un esquema y una funcionalidad.

Create trigger Crear disparadores dado un esquema y una funcionalidad.

Grant/Revoke Gestionar los permisos de un cierto usuario en un cierto esquema de BD dada la lista de operaciones a realizar.

Create sequence Creación de secuencias.

Create index Optimizar el rendimiento del SGBD (el estudiante trabaja en su propia cuenta y da acceso al sistema para que compruebe el resultado). Existen dos variantes

1. Con solución única (dada una sola consulta, proponer la mejor estructura posible).
2. Con solución no única (dado un conjunto de consultas, mejorar el rendimiento del SGBD tanto como se pueda).

5. Utilización y resultados

Desde el punto de vista de los profesores, la utilización del sistema ha sido plenamente satisfactoria, destacando su robustez y versatilidad en cuanto a la variedad de ejercicios posibles. Por parte de los estudiantes, lo han aceptado con normalidad y valoran muy positivamente su uso. Esta conclusión la extraemos de las respuestas a una encuesta que les venimos pasando al final de cada semestre.

El sistema se ha utilizado ya en mayor o menor medida durante 3 semestres, en 3 asignaturas diferentes, por más de 400 estudiantes (175 de los cuales respondieron a la encuesta). El promedio de las respuestas obtenidas es el siguiente (1 es el valor mínimo y 5 el máximo):

- Tener el sistema disponible fuera de las clases de laboratorio me ha ayudado a aprender SQL: 4'08
- Saber la nota y poder reintentarlo ayuda a mejorar el resultado: 4'05
- Los mensages obtenidos en caso de error han sido útiles: 2'99
- Ésta es una buena herramienta para aprender SQL: 3'86

También se ha notado una mejoría en las notas, debido a que al saberla de forma inmediata, tienen la posibilidad de reintentar el ejercicio y mejorarla teniendo en cuenta los mensajes de error obtenidos.

6. Conclusiones y trabajo futuro

En este trabajo hemos presentado nuestro sistema de corrección automática de SQL, haciendo énfasis en la herramienta de gestión de ejercicios. Esta herramienta es la que utiliza el profesor no sólo para introducir los enunciados, soluciones y juegos de pruebas, sino también la interfaz que permite generar de forma automática la salida de esos juegos de pruebas. Actualmente estamos desarrollando la corrección de JDBC y, como trabajo futuro, queremos incluir en el sistema y, en concreto, en la herramienta del profesor la gestión de idiomas diferentes para los enunciados y mensajes de error.

Reconocimientos

Este trabajo ha sido financiado con los proyectos de "*Millora de la Docència*" de la *Universitat Politècnica de Catalunya* y 2007MQD00202 de la *Generalitat de Catalunya*. También nos gustaría agradecer su contribución a Adrián Toporcer en la implementación de la herramienta.

Referencias

[1] IMS QTI Specification. Available at: `http://www.imsglobal.org`.

[2] Abelló, A., Rodríguez, E., Urpí, T., Burgués, X., Casany, M., Martín, C., and Quer, C. LEARN-SQL: Automatic Assessment of SQL Based on IMS QTI Specification. In *ICALT'08* (2008), pp. 592–593.

[3] Dekeyser, S., de Raadt, M., and Lee, T. Y. Computer Assisted Assessment of SQL query Skills. In *ADC* (2007).

[4] Douce, C., Livingstone, D., and Orwell, J. Automatic Test Base Assessment of Programming: a Review. *ACM Journal of Educational Resource in Computing 5*, 3 (2005).

[5] Kenny, C., and Pahl, C. Automated tutoring for a database skills training environment. In *SIGCSE'05* (2005), pp. 59–62.

[6] Mitrovic, A. Learning SQL with a computerized Tutor. In *SIGCSE'98* (1998), pp. 307–311.

[7] Prior, J., and Lister, R. The backwash effect on SQL skills grading. In *ITiCSE'04* (2004), pp. 32–36.

[8] Sadiq, S., Orlowska, M., Sadiq, W., and Lin, J. SQLator: an Online SQL Learning Workbench. In *ITiCSE'04* (2004), pp. 223–227.

[9] Soler, J., Prados, F., Boada, I., and Poch, J. Utilización de una plataforma de e-learning en la docencia de Bases de Datos. In *JENUI'06* (2006), pp. 581–588.