ROCA
REVISTA CIENTÍFICO - EDUCACIONAL
DE LA PROVINCIA GRANMA.

RPNS: 2090 | ISSN: 2074-0735

VOLUMEN 17 (2021)
Número 1 (enero-marzo)

ORIGINAL

Recibido: 09/04/2020 | **Aceptado**: 12/12/2020

Softar, software educativo de álgebra relacional de bases de datos.

Softcar, educational software for relational algebra as a support for the database subject.

MSc. Ángel Enrique Figueredo León, Prof. Auxiliar. [afigueredol@udg.co.cu]
Universidad de Granma. Bayamo, Cuba.

MSc. Edel Ángel Rodríguez Sánchez, Asistente. [edel.rodriguez@utc.edu.ec]
Universidad Técnica de Cotopaxi. Cotopaxi, Ecuador.

MSc. Yanet Esmeralda Silva Pérez, Asistente. [deyanetsilva@gmail.com]
Universidad Técnica de Cotopaxi. Cotopaxi, Ecuador.

Resumen

El presente trabajo propone el estudio y aprendizaje del lenguaje de consultas estructuradas del álgebra relacional de bases de datos. Dicha investigación se basa en el problema existente en la carrera de Ingeniería Informática de la Universidad de Granma en cuanto a la dificultad de los estudiantes para comprender dicho tema de la asignatura Sistemas de Bases de Datos. Es por tal motivo que el objetivo de este trabajo es desarrollar una herramienta que implemente las funcionalidades necesarias para garantizar la recepción y asimilación del conocimiento de manera más fácil para el estudiante. Un estudio realizado acerca de las tendencias tecnológicas actuales sustenta la decisión de optar por desarrollar dicha herramienta con software libre, utilizando para ello Eclipse SDK como IDE y Java como lenguaje de programación. Para el modelado se utiliza la herramienta case Rational Rose, y todo el proceso de desarrollo está basado en la metodología RUP. Lo anteriormente planteado dio como resultado una aplicación de escritorio que responde a la problemática actual.

Abstract

The current paper proposes the study and learning of structured algebra relational data bases consultation language. Such research is based on the existing problem in the Informatic Engineering major at the University of Granma concerning the difficulty of the students to understand this topic of the Database Systems subject. It is for this reason that the objective of this work is to develop a tool that implements the necessary functions to guarantee the knowledge reception and assimilation in an easier and friendlier way for the students. A study about the current technological trends sustains the decision to choose to develop such tool with free software, using Eclipse SDK as IDE and Java as the programming language. For modelling, the Rational Rose case tool is used, and the whole development process is based on the RUP methodology. The above stated resulted in a desktop application that responds to the current problem.

Palabras claves: sistemas de bases de datos; analizador sintáctico; álgebra relacional; técnicas de compilación; software educativo.

Keywords: database systems; syntactic analyzer; relational algebra; compilation techniques; educational software.

Introducción

Desde sus inicios el hombre ha buscado la forma de mejorar su calidad de vida y su forma de trabajo. Para ello ha buscado métodos adecuados tales como la tecnología que ha desarrollado a través de la ciencia. Esto ha permitido llegar a grandes inventos científicos desde la calculadora hasta la computadora. Este gran avance ha llevado a la humanidad a tener un enorme desarrollo social. La computadora se ha convertido en pocos años en parte integrante de la vida cotidiana y

ha permitido que se exploren de manera más amplia y exactas disciplinas cuyos avances no serían posible de ningún otro modo.

El mundo de hoy, está inmerso en una nueva revolución tecnológica basada en la informática, que encuentra su principal impulso en el acceso y en la capacidad de procesamiento de información sobre todos los temas y sectores de la actividad humana.

El sistema educativo actual se encuentra sumido en una época de grandes cambios. La necesidad de innovación tecnológica y los cambios que se realizan en la docencia propician a su vez un importante avance en las estructuras docentes. El uso de las nuevas tecnologías aplicadas a la docencia y la rápida implantación de herramientas software permiten dinamizar y compartir contenidos de modo que amplían el espectro de trabajo en el aula y trasladan el entorno docente a otros lugares y otra división de tiempos distintos a los de la docencia tradicional (Meza y Cantarell ,2006).

Los Software Educativos (SE), se definen de forma genérica como aplicaciones o programas computacionales que facilitan el proceso de enseñanza-aprendizaje. Algunos autores los conceptualizan como cualquier programa computacional, cuyas características estructurales y funcionales sirven de apoyo al proceso de enseñar, aprender y administrar, o el que está destinado a la enseñanza y el autoaprendizaje, y además permite el desarrollo de ciertas habilidades cognitivas (Vidal, Gómez y Ruiz, 2010).

Las instituciones educativas en Cuba están particularmente integradas a las computadoras como una herramienta interactiva para el aprendizaje. Los programas de Educación Asistida por Computadora (CAE), pueden solicitar retroalimentación del usuario y responder de manera apropiada. En forma similar, programas interactivos de aprendizaje pueden enseñar, hacer pruebas de comprensión y repaso basado en lo aprendido por el estudiante.

Las universidades cubanas han estado totalmente inmersas en un proceso de informatización, con el objetivo de lograr una mayor eficiencia tanto en la gestión académica como administrativa, ya que es de vital importancia para el correcto funcionamiento de una universidad, y en general de cualquier centro de educación.

La Universidad de Granma es uno de los centros de educación superior del país que se encuentra enfrascada en la automatización del proceso docente educativo, tanto de pregrado como de postgrado.

Por tanto, es de especial interés desarrollar un software educativo que proporcione un mecanismo mediante el cual los estudiantes puedan explorar las diferentes posibilidades del lenguaje de consultas estructuradas, de manera que vean el resultado de la consulta propuestas en ese mismo momento y puedan realizar las oportunas modificaciones. Actualmente la carrera de Ingeniería Informática no cuenta con un software que permita a los estudiantes comprender y comprobar resultados obtenidos en un ejercicio de álgebra relacional. Solo se cuenta con el criterio del profesor encargado de impartir la asignatura; esto dificulta la forma en que los estudiantes pueden adquirir mayor profundidad en los conocimientos necesarios y suficientes con relación al tema.

Población y Muestra

Inicialmente, se hizo un estudio en la Universidad de Granma para constatar las deficiencias en el proceso docente-educativo relacionados con los contenidos de álgebra relacional de la asignatura de Base de Datos impartida en el 2do año de la carrera de Ingeniería Informática de la Universidad de Granma y determinar la mejor vía para solucionarlo. Se utilizaron métodos y técnicas como el análisis y síntesis para recopilar y procesar la información necesaria; histórico-lógico en el estudio de trabajos similares relacionados la asignatura antes mencionada para

tomarlos como punto de partida en la investigación; la revisión documental para conocer con claridad los datos que son de interés, y la entrevista para obtener datos detallados sobre su procesamiento.

Se realizó una búsqueda de sistemas informáticos que pudieran emplearse para resolver las deficiencias detectadas, y al no encontrarse ninguno, se desarrolló una aplicación con este objetivo que se ajustara a sus necesidades accesible desde nuestra intranet.

El álgebra relacional es una disciplina matemática construida a partir de 8 operadores básicos sobre los que se apoya una serie de axiomas y teoremas. Es un conjunto de operaciones que describen paso a paso cómo computar una respuesta sobre las relaciones, tal y como estas son definidas en el modelo relacional. Se denomina de tipo procedimental, a diferencia del Cálculo Relacional que es de tipo declarativo (Fernández, 2007).

Museros y Sanz (2010) dicen que el álgebra relacional está constituida por una colección de operadores de alto nivel que, aplicados a las relaciones, dan como resultado nuevas relaciones.

Si $R_1, R_2, \ldots, R_j$ *y R'* son relaciones y Op un operador del álgebra relacional, una operación consiste en aplicar Op a la relación o relaciones $R_1, R_2, \ldots, R_j$ que da como resultado la relación *R'*:

$$Op(R_1, R_2, ..., R_j) = R'$$

El álgebra relacional cumple la propiedad de cierre (clausura), ya que el resultado de una operación es otra relación. Si $Op_1, Op_2, \ldots, Op_n$ son operadores del álgebra, se cumple:

$$Op_n(...(Op_2(Op_1(R)))) = R'$$

Un lenguaje de consulta es un lenguaje con el que el usuario solicita información de la base de datos: se construye una expresión que contesta interrogantes sobre la instancia actual de la base

de datos. Básicamente, una consulta (query) es una forma de buscar, encontrar y exhibir determinada información, extrayéndola del cúmulo de datos que almacena la base de datos.

Los datos que responderán a la consulta pueden provenir de una o varias tablas. Las consultas no contienen información de la base de datos, sino tan solo las instrucciones necesarias para seleccionar los registros y campos requeridos de una base de datos.

Las consultas se crean para encontrar todos los registros que contienen una entrada de carácter específica. Se puede utilizar la coincidencia exacta u operadores relacionales cuando se realiza la búsqueda, puesto que se deben encerrar las cadenas de caracteres entre comillas (Quiroz, 2009).

Análisis de los resultados

Los desarrolladores claves, requieren un firme conocimiento del contexto donde se emplazará el sistema, por lo cual, la captura de los requisitos correctos para construir el sistema correcto es sumamente importante. Existen solo dos formas utilizables de representar el entorno en el que se desarrolla el sistema a representar; el modelado del negocio y el modelado del dominio. El modelo del dominio captura los tipos de objetos más importantes en el contexto del sistema. Los objetos del dominio representan los elementos existentes y los eventos en el entorno en donde trabajará el sistema. Por esa razón, se propone un modelo del dominio con el objetivo de definir los tipos más importantes de objetos para representar los elementos y los eventos en el entorno en el cual trabaja el sistema.

El diagrama de la figura 1, visualiza y relaciona los principales conceptos del dominio, el cual va a contribuir posteriormente a identificar algunas de las clases que se utilizarán en el sistema a desarrollar.

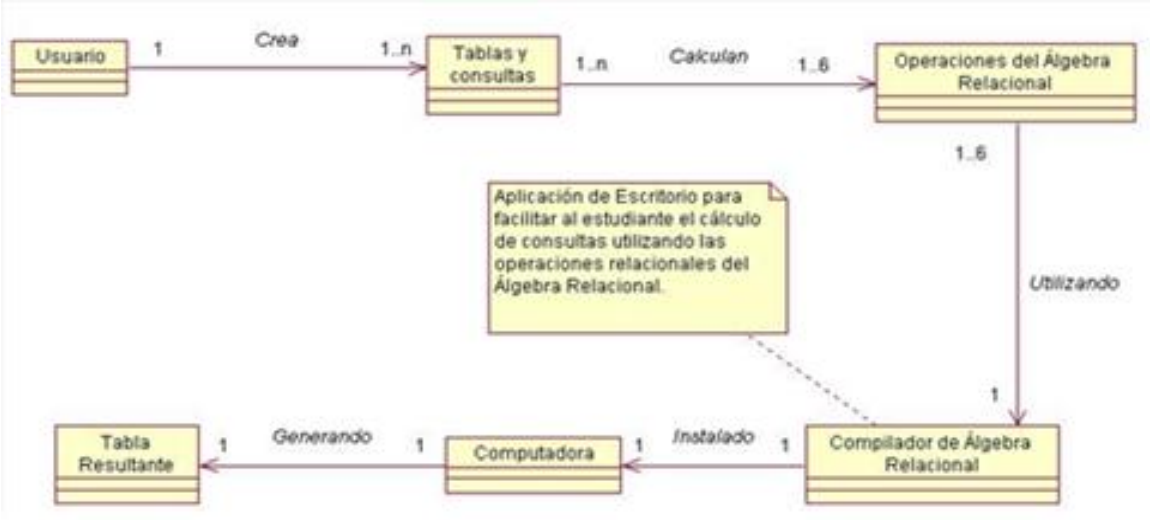

Figura 1. Diagrama de clases del dominio.

Los requisitos funcionales son las acciones que deberá ser capaz de realizar el sistema para satisfacer las necesidades del usuario. Definen los límites de la aplicación y por lo general, se describen mejor a través del modelo de caso de uso y los casos de uso como tal. De acuerdo con los objetivos planteados, el sistema debe de ser capaz de:

R1- Crear tablas.

R2- Introducir datos en las tablas.

R3- Guardar tablas.

R4- Eliminar tablas.

R5- Modificar tablas.

R6- Mostrar tablas.

R7- Introducir consulta.

R8- Procesar los datos.

R9- Guardar el resultado final.

R10- Mostrar ayuda.

Técnicas de compilación utilizadas

En la comunicación hombre-máquina existe una dificultad real; las computadoras operan sobre bits y registros, y los hombres se entienden por medio de idiomas. Los lenguajes de programación son el vehículo de comunicación entre el hombre y la computadora. Los compiladores o traductores son programas encargados de transformar el programa fuente en un programa objeto equivalente. Este programa objeto debe estar expresado, en última instancia, en un lenguaje que la máquina entienda directamente (Mora, 1983).

Implementar un intérprete es un proceso difícil, ya que requiere de un gran esfuerzo durante su desarrollo debido a que es preciso diseñar e implementar algoritmos y estructuras de datos muy complejos (Toro, Corchuelo y Troyano, 1997).

Gramáticas *LL(1)*

Las gramáticas *LL(1)* permiten construir de forma automática un analizador determinista descendente con tan solo examinar en cada momento el símbolo actual de la cadena de entrada (símbolo de preanálisis) para saber qué producción aplicar (Aho, Sethi, y Ullman, 1990).

L: método direccional, se procesa la entrada de izquierda a derecha.

L: se obtiene derivación más a la izquierda.

1: se usa un símbolo de preanálisis para decidir la producción a seguir.

Teorema 1: Una gramática *LL(1)* no puede ser recursiva a izquierdas y debe ser factorizada.

Teorema 2: Una gramática *LL(1)* es no ambigua.

La condición *LL(1)*

Para que una gramática sea *LL(1)* se debe cumplir que:

:Para las producciones de la forma $A \rightarrow \alpha_1|\alpha_2|...|\alpha_n$ se debe cumplir que

$PRIMEROS\ (\alpha_i) \cap PRIMEROS\ (\alpha_j) = \emptyset$ *para todo i, j* $(i \neq j)$.

Esta regla permite decidir qué alternativa elegir conociendo solo un símbolo de la entrada.

Si A Є V_n, tal que {ε} Є PRIMEROS (A), entonces:

PRIMEROS (α_i) ∩SIGUIENTES (A) = Ø

Esta condición garantiza para aquellos símbolos que pueden derivar de la cadena vacía, el primer símbolo generado y el siguiente detrás de ellos sean distintos, sino no sabría cómo decidir si se va a empezar a reconocer el no-terminal o si ya lo hemos reconocido (Louden, 1997).

Además, se utilizó un analizador sintáctico descendente predictivo no-recursivo sin llamadas recursivas a procedimientos, que utiliza una pila auxiliar de símbolos terminales y no-terminales. Para determinar qué producción debe aplicarse en cada momento, se busca en una tabla de análisis sintáctico (parsingtable) en función del símbolo de preanálisis que se está observando en ese momento y del símbolo en la cima de la pila se decide la acción a realizar (Vivancos, 2000). En la figura 2 se muestra un analizador sintáctico descendente dirigido por tabla.

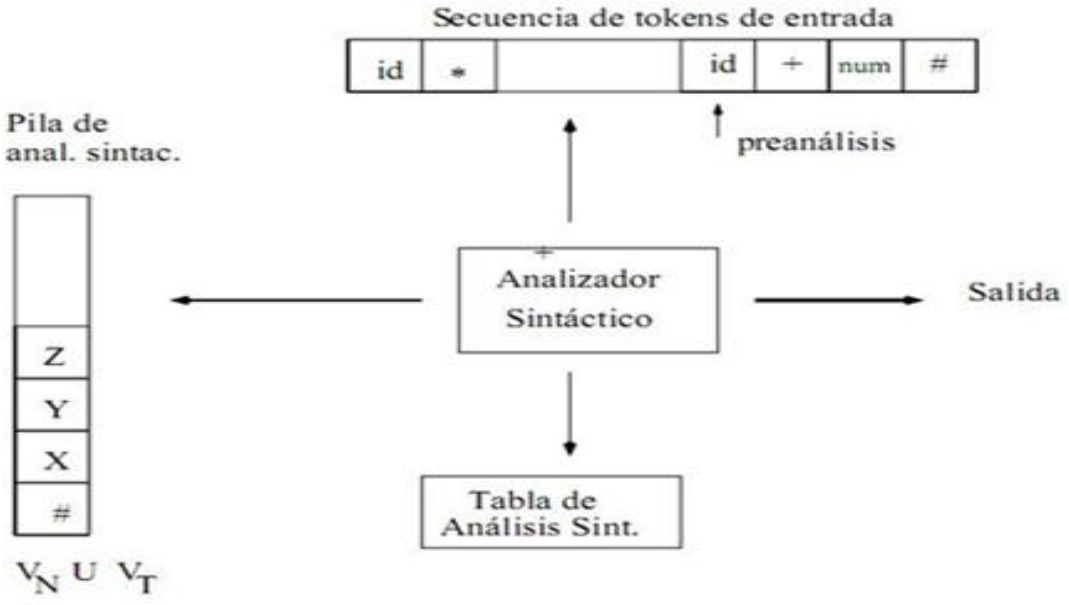

Figura 2. Esquema de un analizador sintáctico descendente dirigido por tabla.

Buffer de entrada**:** contiene la cadena de componentes léxicos que se va a analizar seguida del símbolo # de fin de cadena.

La pila de análisis sintáctico: contiene una secuencia de símbolos de la gramática con el símbolo # en la parte de abajo que indica la base de la pila.

Una tabla de análisis sintáctico**:** es una matriz bidimensional *M [X, a] con X Є Vnt y α Є Vt incluido {#}*. La tabla de análisis sintáctico dirige el análisis y es lo único que cambia de un analizador a otro (de una gramática a otra).

Salida: serie de producciones utilizadas en el análisis de la secuencia de entrada.

El analizador sintáctico tiene en cuenta en cada momento, el símbolo de la cima de la pila y el símbolo en curso de la cadena de entrada (el símbolo de preanálisis). Estos dos símbolos determinan la acción a realizar, que pueden ser, según Luengo y Cueva (1996):

Si X == a == #, el análisis sintáctico ha llegado al final y la cadena ha sido reconocida con éxito.

Si X == a ≠ #, el terminal se desapila (se ha reconocido ese terminal) y se avanza en la entrada obteniendo el siguiente componente léxico.

Si X es no-terminal, entonces se consulta en la tabla M [X, a]. La tabla contiene una producción a aplicar o si está vacía significa un error. Si es una producción de la forma $X \rightarrow Y_1Y_2\ldots Y_n$, entonces se meten los símbolos $Y_n\ldots Y_1$ en la pila (notar el orden inverso). Si es un error el analizador llama a una rutina de error.

El análisis sintáctico dirigido por tabla es más eficiente en cuanto a tiempo de ejecución y espacio, puesto que usa una pila para almacenar los símbolos a reconocer en vez de llamadas a procedimientos recursivos (Herrera, 1982).

A continuación se expone el algoritmo de análisis sintáctico predictivo dirigido por tabla.

Entrada: una cadena w, una tabla de analizador sintáctico y una gramática G.

Salida: Si w ∈ L(G), la derivación más a la izquierda de la cadena w, sino una indicación de error

Método:

Como configuración inicial se tiene en el fondo de la pila el símbolo #, el axioma S en la cima y la cadena w# en el buffer de entrada.

Iniciar preanálisis con el primer símbolo de w#.

repetir

Sea X el símbolo de la cima de la pila

si X es un terminal o # entonces

si X == preanalisis entonces

extraer X de la pila y obtener nuevo componente léxico

sino error();

sino

si M[X,preanalisis] = $X \rightarrow Y_1\ Y_2 \ldots Y_n$ entonces

extraer X de la pila

meter $Y_n \ldots Y_2 Y_1$, con Y_1 en la cima de la pila

sino error();

hasta_que X == # (* la pila está vacía y hemos procesado toda la entrada *)

Cadena reconocida con éxito

Para el desarrollo de la aplicación informática se hizo uso de la metodología del Proceso Unificado de Desarrollo de Software (RUP). Este permite definir un sistema de software; para detallar los artefactos en el sistema; para documentar y construir. Durante todos los pasos del

ciclo de vida del software y facilitando el mejoramiento de este, se hizo uso como herramienta CASE, el Rational Rose Enterprise Edition ya que proporciona excelentes facilidades de interoperabilidad con otras aplicaciones. JAVA fue lenguaje de programación escogido para implementar el software, porque es un lenguaje multiplataforma, con el cual se pueden desarrollar programas que se ejecuten sin problemas en sistemas operativos como Windows, Linux, Mac y Unix.

Una vez definido el lenguaje de programación, se procedió a definir el entorno de desarrollo, en este caso se hizo uso del Eclipse SDK 3.2. Esta es una poderosa y completa plataforma de programación, desarrollo y compilación de elementos tan diversos como sitios Web, programas en C++ o aplicaciones Java. Incluye todas las herramientas y bibliotecas estándar de Java necesarias para crear applets y aplicaciones.

El sistema garantiza de manera sencilla, dinámica y atractiva la gestión de los datos y se logró gracias a la aplicación de algunos principios de diseño. Uno de estos ha sido optar por mantener un mismo tamaño para las formas, para obtener de esta manera la uniformidad en el diseño.

La interfaz de usuario, es la categoría de diseño que crea un medio de comunicación entre el hombre y la máquina. Además, el diseño identifica los objetos y acciones de la interfaz y crea entonces un formato de pantalla que formará la base del prototipo de interfaz de usuario.

El diseño de la interfaz es un punto fundamental a tener en cuenta a la hora de presentar la aplicación, porque se considera que es la cara mostrada al usuario y debe ser lo más amigable y comprensible posible.

En el diseño de las pantallas se deben tener en cuenta varios aspectos: organización de los elementos en la pantalla, dónde se coloca la información y cómo se estructura. Las ventanas de la interfaz deben ser diseñadas de forma uniforme garantizando:

✓ El equilibrio en la organización de la información, por ejemplo, todas las ventanas que muestran información siempre la mostrarán en el mismo orden.

✓ La optimización de la cantidad de elementos en la pantalla, ayudando al fácil manejo y mejor comprensión de la información mostrada en pantalla.

✓ La unidad, donde cada elemento de la pantalla se debe diseñar siguiendo un patrón de tamaño, colores y formas.

En la figura 3 se muestra la interfaz que permite realizar todas las operaciones relacionadas con las tablas de una base de datos.

Figura 3. Interfaz del Usuario.

El sistema analiza sintáctica y semánticamente una consulta; la ejecuta y devuelve los resultados, los mismos deben ser claros y precisos para que el usuario pueda comprender la forma en que el sistema los muestras, dando la posibilidad de guardarlos. A continuación en la figura 4 se presenta una vista del sistema mostrando los resultados obtenidos luego de calcular una consulta.

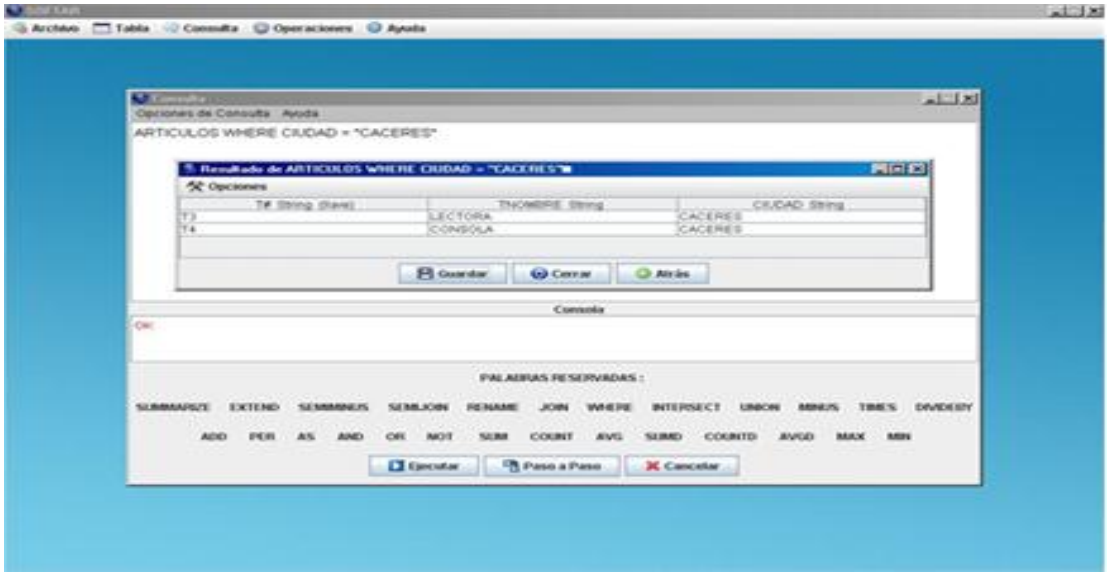

Figura 4. Ejemplo de cómo se muestran los resultados dentro del sistema.

El tratamiento de los errores es uno de los elementos fundamentales para el buen funcionamiento de un sistema, pues garantiza la armonía y facilidad de uso de la aplicación. Es importante partir del hecho que los usuarios son altamente propensos a cometerlos, por grandes o pequeños que estos sean. El sistema debe prevenir y ayudar a corregirlos mediante el uso de mensajes altamente descriptivos sobre el origen de los mismos y la posibilidad de su corrección. También pueden ser generados por los usuarios en el momento de entrar los datos o generados por el sistema frente a algunas acciones y de manera inesperada. Estas tareas de detección de los mismos se realizan con el objetivo de garantizar al usuario el máximo de calidad y fiabilidad. Mediante la figura 5 se muestra un ejemplo de tratamiento de errores.

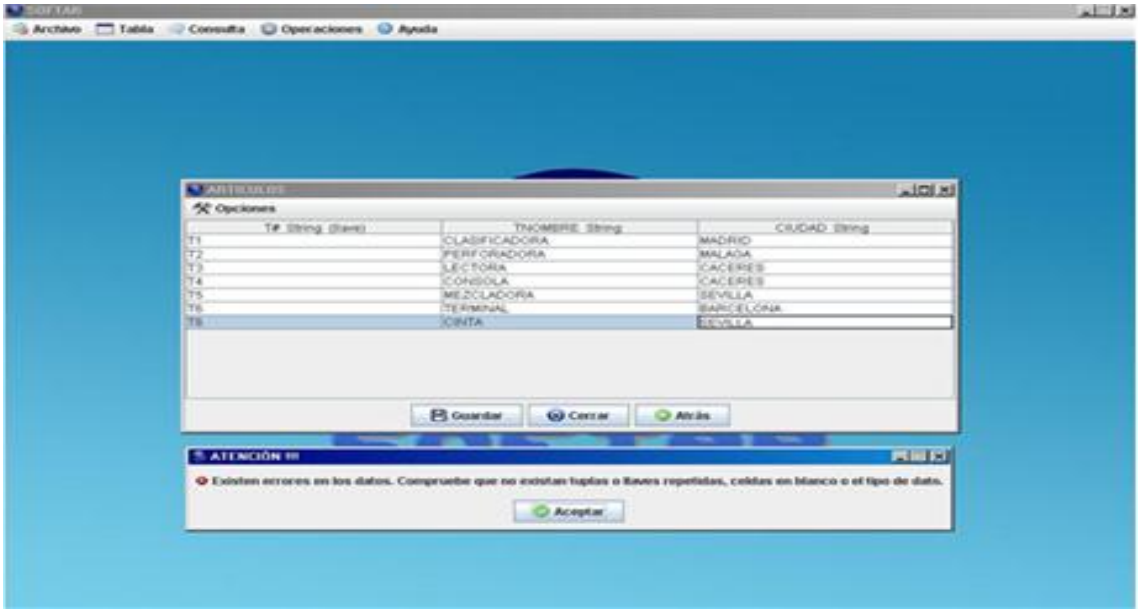

Figura 5. Ejemplo del tratamiento de las excepciones dentro del sistema.

El éxito de un software se puede definir como calidad del sistema, de los datos y del servicio. La aplicación propuesta en esta investigación impacta de manera positiva en la preparación de los estudiantes de la carrera de Ingeniería en Informática en la asignatura de Bases de Datos evidenciados en los resultados obtenidos por los estudiantes. Con su implantación, no es necesario mejorar el equipamiento disponible, pues se desarrolló en función de los requerimientos técnicos existentes. Las tecnologías requeridas para su funcionamiento están basadas en software libre por lo que puede ser modificado en caso de cualquier error técnico o necesidad de ponerles nuevas prestaciones.

Para evaluar su calidad se empleó el criterio de expertos mediante el método Delphi. Este consiste en la utilización sistemática del juicio intuitivo de un grupo de expertos para obtener un consenso de opiniones informadas. Es considerado como uno de los métodos subjetivos de pronosticación más confiables (Blanco, López y Mengual, 2010; García y Suárez, 2013).

Los expertos que evaluaron la calidad de la aplicación fueron 20. Respondieron una encuesta que se confeccionó a partir de los siguientes indicadores generales de evaluación:

1. Resulta una interfaz amigable y fácil de operar.
2. Garantiza la disponibilidad de la información actual e histórica.
3. Facilita el análisis de la información actualizada.
4. Garantiza la seguridad de la información.
5. Constituye una herramienta útil para la preparación de los estudiantes en la asignatura de Bases de Datos.
6. Eleva la calidad de la preparación de los estudiantes.
7. Garantiza la obtención correcta de la información.

Después de realizado el procesamiento de la encuesta, los resultados arrojaron que la aplicación cumple con todos los indicadores antes mencionados.

Con el propósito de valorar la efectividad de la propuesta, se seleccionó un grupo de expertos sobre la base de la labor profesional desarrollada por estos. Se utilizó la metodología de preferencia debido a la imposibilidad de réplica. Las categorías son de la siguiente forma: excelente 5, bien 4, regular 3, deficiente 2, mal 1 y descartado 0.

Tabla 1
Resultados de Opinión de Experto.

NO	Indicadores	EXPERTOS					
		A	B	C	D	F	Media
1	Capacidad de motivación al estudiante	5	5	5	5	5	5
2	Adecuación a los destinatarios	5	4	5	4	4	4,4
3	Fomento del autoaprendizaje	4	4	5	4	5	4,4
4	Estímulo a la creatividad del estudiante	5	5	4	4	4	4,2
5	Operatividad del medio soporte del sistema de ejercicios	5	5	4	4	5	4,6
Media		4,8	4,6	4,6	4,2	4,6	

La media general de los expertos fue 4,56.

Conclusiones

1. Se desarrolló un sistema informático que automatiza el proceso del álgebra relacional, facilitándole a la carrera de Ingeniería en Informática de la universidad de Granma una herramienta eficiente como apoyo al proceso docente-educativo en la asignatura de Bases de Datos.
2. Se utilizaron métodos y criterios que corroboraron la eficacia de dicho sistema.

Referencias Bibliográficas

Aho., A., Sethi., V., & J, U. (1990). Compiladores: principios, técnicas y herramientas. Barcelona, España: Addison Wesley Iberoamericana S.A

Blanco, J.E., López, A. y Mengual, S. (2010). Validación mediante método Delphi de un cuestionario para conocer las experiencias e interés hacia las actividades acuáticas con especial atención al Windsurf. ÁGORA, 12(1), 75-96.

Fernández, C. (2007). *El modelo relacional de datos: de los fundamentos a los modelos deductivos*: Ediciones Díaz de Santos.

García, M. y Suárez, M. (2013). El método Delphi para la consulta a expertos en la investigación científica. Revista Cubana de Salud Pública, 39(2), 253–267.

Herrera, E. (1982). Técnicas de Compilación. Granada, España: E.T.S. de Ingenierías Informática y de Telecomunicación de la Universidad de Granada

Louden, K. C. (1997). *Compiler Construction: Principles and Practice*. San Jose State University, USA

Luengo, M. C., & Cueva, J. M. (1996). *Análisis Sintáctico en Procesadores del Lenguaje*. Ovideo, España: Departamento de Matemáticas de la Universidad de Oviedo

Meza, A. M., & Cantarell, L. (2006). Importancia del Manejo de Estrategias de Aprendizaje para el uso Educativo de las Nuevas Tecnologías de Información y Comunicación en Educación. *MISTICA*.

Mora, K. (1983). Lenguajes de Programación y Técnicas de Compilación. Habana, Cuba: Félix Varela.

Museros, L., & Sanz, I. (2010). Documentación Explicativa del Tema 4: El Modelo Relacional. Algebra Relacional y SQL. *DSpace*. Castellón de la Plana, España: Departamento de Ingeniería y Ciencia de los Computadores, Universidad Jaume I

Quiroz, J. (2009). Las consultas en el álgebra relacional. *Boletín de Política Informática* (6), 53-61.

Toro, M., Corchuelo, R., & Troyano, J. A. (1997). Apuntes de Compiladores. Departamento de Lenguajes y Sistemas Informáticos Facultad de Informática y Estadística, Universidad Hispalense, Sevilla, España

Vidal, M., Gómez, F., & Ruiz, A. M. (2010). Software educativos Scielo, 24(1), 97-110.

Vivancos, E. (2000). Compiladores I: Una introducción a la fase de análisis. Valencia, España: Servicio de Publicaciones / Universidad Politécnica de Valencia