Proyecto de fin de Carrera
Ingeniería en Informática
Curso 2009/10

Herramienta para el aprendizaje del álgebra relacional y optimización de consultas

Roberto Yus Peirote

Director: Sergio Ilarri Artigas

Departamento de Informática e Ingeniería de Sistemas
Centro Politécnico Superior
Universidad de Zaragoza

Junio de 2010

Herramienta para el aprendizaje del álgebra relacional y optimización de consultas

RESUMEN

Cada día nos encontramos con actividades que requieren algún tipo de interacción con bases de datos. Por ejemplo, si acudimos al banco a sacar o ingresar dinero, si compramos unos billetes de tren, o si reservamos una habitación de un hotel, estamos interaccionando con distintas aplicaciones de bases de datos. Todas estas interacciones se traducen en consultas a diferentes sistemas gestores de bases de datos. La necesidad de realizar estas consultas de la forma más eficaz y eficiente posible hace que el estudio de las bases en las que se asientan sea muy importante. Debido a que la mayoría de las bases de datos responden al modelo relacional, entender el álgebra relacional es imprescindible para consultar a dichas bases de datos. Con este proyecto se ha desarrollado una aplicación gráfica que pretende sobre todo ser una herramienta para ayudar en el *aprendizaje* del álgebra relacional y la optimización de consultas.

La aplicación desarrollada permite introducir expresiones en álgebra relacional de forma intuitiva, así como árboles de expresiones en álgebra relacional. De esta forma se permite al estudiante aprender a familiarizarse con este proceso. También se encuentra entre sus funcionalidades la introducción de relaciones de ejemplo, ya sea desde la propia aplicación o a través de ficheros externos o incluso desde bases de datos externas. Definir las relaciones con las que va a trabajar ayuda al usuario a comprender de una forma más clara los pasos que va realizando. Una vez definidas las consultas, la aplicación puede ejecutarlas de forma automática o paso a paso tanto en la propia aplicación como en un sistema gestor de bases de datos externo. Además, se permite al usuario realizar la optimización de las consultas introducidas de forma automática o paso a paso aplicando tanto las reglas de transformación del álgebra relacional como las estadísticas de las relaciones, permitiendo al usuario comprender estos procesos por medio de explicaciones. Por último, se permite traducir las expresiones en álgebra relacional a SQL puesto que es el lenguaje estándar de consulta en los sistemas gestores de bases de datos actuales y así el alumno puede observar la relación existente entre ambos lenguajes.

Para el desarrollo de la aplicación se ha utilizado el lenguaje Java, y se ha dividido en dos partes. En la primera se ha creado el entorno gráfico de la herramienta y en la segunda parte se han implementado las funcionalidades principales de la aplicación así como los mecanismos de detección de errores.

Como balance general del proyecto, se ha implementado una herramienta que cumple con todos los requisitos marcados inicialmente, que se traducen en: lograr una herramienta útil, potente, intuitiva y versátil para el aprendizaje del álgebra relacional y la optimización de consultas.

Agradecimientos

Quiero agradecer a mis padres Carmen y Antonio, a mi hermana María y al resto de mi familia la compresión y el afecto que me han demostrado a lo largo de todos estos años y en especial los últimos años de la carrera ya que con su apoyo he podido continuar adelante. Un ciclo de mi vida está a punto de terminar y sé que siempre contaré con ellos para el ciclo que empieza ahora.

También quiero agradecer a mis amigos de carrera con los que he compartido muchos momentos a lo largo de estos años y que se han convertido en mi segunda familia. Creo muy sinceramente que las amistades forjadas perdurarán a lo largo de los años y eso me reconforta.

Y, por supuesto, quiero agradecer a Sergio la posibilidad de finalizar mis estudios trabajando en un aspecto tan interesante como este y también su ayuda y colaboración a lo largo de todo el proyecto.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

El presente documento representa la memoria de trabajo del proyecto de final de carrera "*Herramienta para el aprendizaje del álgebra relacional y optimización de consultas*", realizado por el alumno Roberto Yus Peirote y dirigido por Sergio Ilarri Artigas. Se ha desarrollado dentro del marco del Departamento de Informática e Ingeniería de Sistemas del Centro Politécnico Superior de la Universidad de Zaragoza.

En este proyecto se ha desarrollado una herramienta gráfica que permite definir expresiones en álgebra relacional o árboles de expresiones en álgebra relacional y posteriormente ejecutar dichas consultas sobre relaciones de ejemplo extraídas de ficheros externos, bases de datos externas o introducidas por el usuario en la propia aplicación. También se permite realizar la optimización de estas consultas así como la traducción de las consultas en álgebra relacional a SQL.

La herramienta tiene principalmente una finalidad educativa, e incluye en consecuencia modos de ejecución u optimización paso a paso, detección de errores, etc.

1.1. Motivación

En la sociedad actual, las bases de datos están presentes en muchas de nuestras actividades diarias. Desde la compra de entradas o billetes, al ingreso u extracción bancaria o incluso la consulta de paginas web, a lo largo de cada jornada consultamos bases de datos de muchas formas. Dado que a día de hoy todavía no hay herramientas serias para consultar a bases de datos en lenguaje natural, es necesario que estas consultas se realicen en un lenguaje comprensible por los sistemas gestores de bases de datos. El lenguaje estándar para esto es *SQL* (*Structured Query Language*), pero antes de aprender a realizar consultas en este lenguaje es necesario comprender las bases en las que se asienta. Por ello, el estudio y la comprensión del álgebra relacional y sus principios son un aspecto muy importante en la formación en bases de datos de los alumnos de *Ingeniería en Informática.*

El plan actual de Ingeniería en Informática de la Universidad de Zaragoza sólo contempla una asignatura obligatoria de bases de datos: *Ficheros y Bases de datos*. Y el próximo plan de estudios con la aparición de los grados también tiene una sola asignatura obligatoria de bases de datos: *Bases de datos*. Por lo tanto, y dado que los alumnos que no amplíen sus conocimientos en la materia cursando optativas solamente van a dedicar un cuatrimestre al estudio de las bases de datos, es importante que puedan disponer de herramientas para simplificar dicho estudio y practicar (en este caso la construcción de consultas en álgebra relacional) de la forma mas eficaz posible.

Existen multitud de libros e información acerca de la construcción de consultas en álgebra relacional y de su optimización, pero debido a su componente matemático, la forma mas ilustrativa para su aprendizaje son los ejemplos. Esta herramienta pretende servir al alumno para realizar su aprendizaje en función a relaciones dadas o a relaciones introducidas por él (por lo tanto más familiares).

1.2. Objetivo y alcance

El objetivo del proyecto es desarrollar una herramienta que permitiera:

- Definir y manipular expresiones en álgebra relacional y árboles de expresiones en álgebra relacional a través de un interfaz gráfico fácil de usar, de tal forma que el usuario pueda comprobar la corrección de las mismas, así como obtener el árbol asociado a la expresión que introduzca y viceversa.
- Interpretar las expresiones en álgebra relacional permitiendo la evaluación de las mismas sobre relaciones de ejemplo ayudando al usuario a entender el proceso. Además, esta ejecución puede realizarse de forma local (a través de la propia aplicación) así como remota (a través de la conexión a un gestor de bases de datos externo). El usuario podrá también ejecutar las consultas paso a paso para una mejor comprensión del proceso.
- Traducir las expresiones en álgebra relacional a SQL de tal forma que el usuario pueda entender la relación entre ambos lenguajes y ejecutar sus consultas posteriormente en cualquier sistema gestor de bases de datos relacionales.
- Optimizar las consultas introducidas por el usuario aplicando las propiedades de las operaciones del álgebra relacional, así como datos estadísticos de las relaciones involucradas. Esta optimización se realizará de forma automática o interactiva mostrándose en este último caso los pasos seguidos.
- Definir relaciones de ejemplo e introducir datos para esas relaciones (ya sea de forma manual o automática) que ayuden al usuario en la comprensión del álgebra relacional.

Esta herramienta tiene carácter docente lo que implica que todas sus funcionalidades están pensadas para facilitar la compresión del álgebra relacional.

1.3. Contenido de la memoria

La división de la memoria contempla los siguientes apartados:

- Capítulo 1, introducción a la problemática abordada y los objetivos marcados.
- Capítulo 2, descripción de la tecnología empleada en el desarrollo del proyecto.
- Capítulo 3, análisis de las herramientas existentes relacionadas con el proyecto.
- Capítulo 4, descripción del proceso de desarrollo de la herramienta.
- Capítulo 5, resumen de los resultados del proyecto, conclusiones del autor y las posibles aportaciones de este PFC a líneas futuras de trabajo.

Además se incluyen los siguientes anexos:

- Anexo A - manual de usuario de la herramienta.
- Anexo B - estudio del contexto del proyecto.
- Anexo C - análisis y diseño de la aplicación.
- Anexo D - ejemplos de uso.

1.4. Marco temporal del proyecto

En el diagrama de la Figura 1.1 se resume el desarrollo temporal del proyecto. También se incluye en la tabla 1.1 un resumen de las horas que han sido necesarias para completar cada una de las partes del proyecto.

Nombre	Fecha de inicio	Fecha de fin
Estudio Previo	15/12/09	20/01/10
Documentación tema	15/12/09	20/01/10
Planificación	15/01/10	20/01/10
Análisis y Diseño	20/01/10	27/02/10
Estudio Estado del Arte	20/01/10	28/01/10
Especificación de Requisitos	25/01/10	11/02/10
Diagrama de clases	15/02/10	25/02/10
Casos de uso	25/02/10	27/02/10
Prototipado ventanas	1/03/10	2/03/10
Implementación	1/03/10	20/05/10
Interfaz Gráfica	1/03/10	3/04/10
Funcionalidades Aplicación	29/03/10	18/05/10
Pruebas	17/05/10	25/05/10
Documentación	20/05/10	12/06/10
Elaboración manual	20/05/10	25/05/10
Ejemplos y tutoriales	25/05/10	29/05/10
Memoria	31/05/10	12/06/10

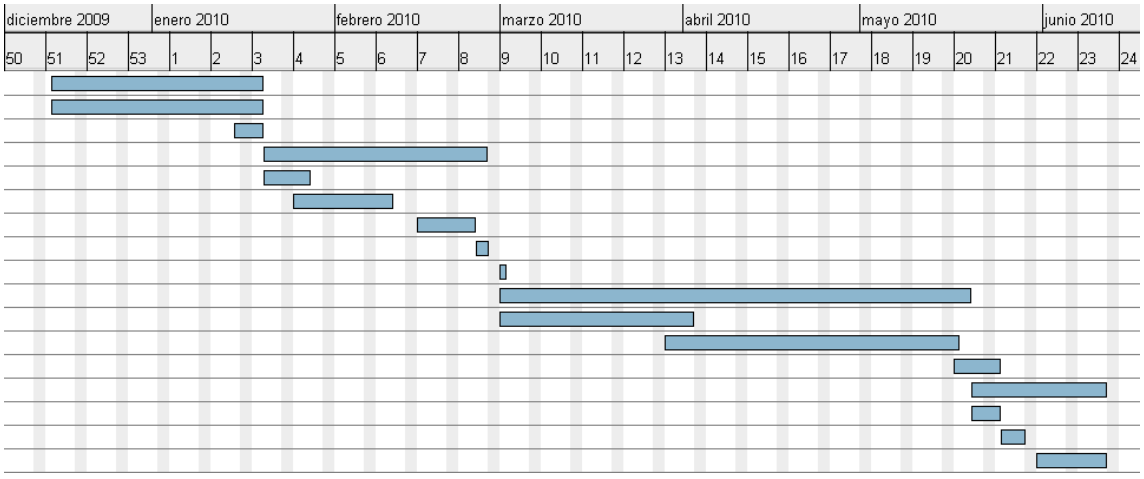

Figura 1.1: Diagrama de Gantt de desarrollo de Proyecto

Tiempo dedicado por tarea

Tarea	Tiempo
Estudio Previo	20h
Análisis y Diseño	35h
Implementación	395h
Pruebas	45h
Documentación	110h
Total	**605h**

Tabla 1.1: Resumen temporal

Capítulo 2

Contexto Tecnológico

En este capítulo se detallan las diferentes tecnologías que fueron utilizadas en el desarrollo del presente PFC. En primer lugar se explican algunos conceptos referentes al modelo relacional, necesarios para entender el contenido de proyecto (estos conceptos están explicados de una forma más extensa en el Anexo B). A continuación se introduce brevemente el lenguaje utilizado para la implementación de la aplicación así como el software utilizado durante otras fases del desarrollo del proyecto.

2.1. El modelo relacional

Para ser capaces de comprender con exactitud el álgebra relacional es necesario realizar un repaso acerca de modelo relacional.

Antes de definir una relación necesitamos definir en primer lugar otros dos términos, dominio y atributo. Cada atributo de una relación se caracteriza por un nombre y por un dominio. El dominio indica qué valores pueden ser asumidos por una columna de la relación. A menudo un dominio se define a través de la declaración de un tipo para el atributo (por ejemplo diciendo que es una cadena de diez caracteres), pero también es posible definir dominios más complejos y precisos.

2.2. El álgebra relacional

El álgebra relacional es un lenguaje procedural de consulta y está formado por cinco operaciones fundamentales. Estas operaciones son la proyección, selección, producto cartesiano, unión y diferencia. Toda consulta puede ser generada con esas cinco operaciones, pero se incluyen además una serie de operaciones adicionales que simplifican la definición de consultas usuales. Estas operaciones adicionales incluyen la intersección, el natural join, outer joins y la división. El operador de renombre

algunas veces es incluido como operación auxiliar aunque podría considerarse una operación fundamental.

Todas los operadores del álgebra relacional utilizan como operandos relaciones. Además, estos operadores producen como resultado una nueva relación. Los operadores pueden unirse para expresar consultas más complejas.

Para las operaciones del álgebra relacional, una relación es vista como un conjunto de tuplas. Estas operaciones del álgebra relacional son pues las operaciones utilizadas en la teoría de conjuntos con algunos operadores adicionales.

2.2.1. Operaciones Fundamentales

A continuación se explicarán las operaciones fundamentales del álgebra relacional.

Proyección

La proyección es una operación unaria que se denota por la letra griega pi (π). De una forma intuitiva podemos definir la proyección sobre una relación como la misma relación omitiendo algunas columnas. Definiendo la operación de una forma más formal, la operación de proyección copia la relación que utiliza como argumento dejando algunas columnas atrás. Las columnas que queremos conservar se pasan a la operación como una lista de atributos.

Selección

La selección es una operación unaria que selecciona las tuplas que satisfacen un predicado dado. Al igual que la proyección selecciona solamente un subconjunto de atributos, la selección selecciona un subconjunto de tuplas. Esta operación se denota por la letra minúscula griega sigma (σ). La relación pasada como argumento se expresa entre paréntesis y la condición de selección se anota al lado del operador.

Producto Cartesiano

El producto cartesiano genera como resultado una relación cuyas tuplas son formadas combinando cada posible par de tuplas: siendo uno de ellos de la relación R y el otro de la relación S. Si hay n_1 tuplas en R y n_2 tuplas en S, entonces hay $n_1 n_2$ tuplas en la relación resultante de aplicarles el producto cartesiano.

Unión

La operación binaria de unión se denota, al igual que en teoría de conjuntos, por $\cup$. La unión se utiliza para unir el contenido de sus argumentos, sin embargo, la unión del modelo relacional no es tan general como la unión del modelo matemático ya que la del modelo relacional solamente une relaciones compatibles.

Diferencia

La última de las operaciones fundamentales del álgebra relacional es la diferencia. Esta operación se denota con el símbolo $-$. Para poder aplicar este operador, las dos relaciones tienen que ser compatibles. El resultado de la expresión R $-$ S, es una relación obtenida al incluir todas las tuplas de R que no aparecen en S.

Renombre

Como se ha mencionado anteriormente, la operación de renombre se omite algunas veces de la lista de operaciones fundamentales o, en ocasiones, se clasifica como una operación auxiliar. La operación de renombre, denotada por la letra minúscula griega rho (ρ), es una operación unaria. El resultado de aplicar el operador de renombre a una relación es una relación idéntica a la original excepto por el hecho de que los atributos tienen nuevos nombres.

2.2.2. Construcción de expresiones

Los cinco operadores fundamentales dan al álgebra relacional el poder para formular consultas complejas. El álgebra relacional es definida por las siguientes reglas:

1. Si R es una relación, entonces es una expresión algebraica.

2. Si *E1* y *E2* son expresiones algebraicas, entonces también lo son *E1*$\cup$*E2*, *E1* – *E2* y *E1* $\times$ *E2*. Para poder aplicar la unión y la diferencia, *E1* y *E2* deben ser compatibles.

3. Si E es una expresión algebraica, entonces también lo son $\pi_L(E)$, $\sigma_P(E)$, $\rho_S(E)$, siendo L una lista de nombres de atributos contenidos en el esquema de E, P un predicado con los atributos de E y S una lista de nuevos nombres.

4. Si E es una expresión, también lo es (E).

2.2.3. Operadores adicionales

Hemos introducido las cinco operaciones fundamentales del álgebra relacional. Con estos cinco operadores disponemos del potencial necesario para expresar cualquier consulta en álgebra relacional. Sin embargo, algunas consultas muy utilizadas se convierten en expresiones muy complejas cuando nos vemos restringidos a utilizar estas operaciones. Es por esto que surgen un conjunto de operaciones adicionales que simplifican las consultas.

Es necesario notar que los operadores adicionales no incrementan el potencial del álgebra relacional. Cualquier consulta que pueda ser expresada utilizando los operadores adicionales puede ser expresada también solamente con operadores fundamentales. La única razón para su introducción es pues la simplificación de consultas comunes. A continuación se explican algunos de ellos.

Intersección

La primera operación que añadimos al álgebra relacional es la intersección, la cual es denotada por $\bigcap$ y se trata de una operación binaria. Después de aplicar el operador de intersección obtenemos una relación que contiene solamente aquella tuplas de R que aparecen también como tuplas en S.

División

La división es una operación binaria denotada por $\div$. El resultado de aplicar la operación de división $R \div S$ consiste en las tuplas de R, en las cuales sólo se seleccionan los atributos únicos de R (que aparecen en la cabecera de R pero no en la de S), tales que existen todas sus combinaciones con las tuplas de S en R.

Natural Join

Otra operación adicional que vamos a describir es el *natural join* (o reunión natural). Es una operación binaria denotada por $\bowtie$. El natural join es una operación muy utilizada que nos permite combinar relaciones que tienen algún atributo común sin necesidad de especificar nada más.

Outer Joins

Un outer join es similar al natural join explicado anteriormente. La única diferencia es que el outer join mantiene la información que habría sido perdida reemplazando la información faltante por *nulls*.

Hay tres tipos de outer joins, left, right y full. El *left outer join* se denota por $^{+}\bowtie$. Esta operación aplicada a dos relaciones R y S es el conjunto de todas las combinaciones de tuplas de R y S que son iguales en sus atributos comunes, además de las tuplas de R que no tienen pareja en los atributos de S.

En cuanto al *right outer join*, su estructura es muy similar al left outer join. Se denota por $\bowtie^{+}$. Esta operación aplicada a dos relaciones R y S es el conjunto de todas las combinaciones de tuplas de R y S que son iguales en sus atributos comunes, además de las tuplas de S que no tienen pareja en los atributos de R.

Y por último el *full outer join*, denotado por $^{+}\bowtie^{+}$ y cuyo resultado aplicado a dos relaciones R y S es el conjunto de todas las combinaciones de tuplas de R y S que son iguales en sus atributos comunes, además de las tuplas de S que no tienen pareja en los atributos de R y las tuplas de R que no tienen pareja en los atributos de S.

Semijoin

El semijoin es similar al natural join y se expresa mediante el símbolo $\rhd <$. El resultado de la operación de semijoin $R \rhd < S$ es el conjunto solamente de aquellas tuplas de R para las cuales hay una tupla en S que es igual en cuanto a sus atributos comunes.

2.3. Optimización de consultas

Las consultas pueden ser representadas por un árbol, donde:

- Los nodos internos son operadores.
- Las hojas son relaciones.
- Los subárboles son subexpresiones.

El objetivo principal es transformar los árboles de expresiones en árboles de expresiones equivalentes, donde el tamaño medio de las relaciones fruto de las subexpresiones del árbol son menores de lo que eran antes de la optimización. A continuación se presentan un conjunto de reglas, que pueden ser utilizadas en dichas transformaciones.

- La proyección es idempotente.
- La selección es idempotente.

- La selección es conmutativa.
- La selección es conmutativa respecto a la proyección.
- La proyección es conmutativa respecto al join y al producto cartesiano.
- La selección es conmutativa respecto a las operaciones de conjuntos.
- La proyección es conmutativa respecto a la unión.
- Join y producto cartesiano son asociativas.
- Join es conmutativa.
- Conmutatividad de las operaciones de conjuntos: la unión y la intersección son conmutativas pero la diferencia no.
- Asociatividad de las operaciones de conjuntos: la unión y la intersección son asociativas pero la diferencia no.

2.4. Java

Para el desarrollo del proyecto se ha utilizado el lenguaje de programación Java [5], desarrollado por Sun Microsystems [6] a principios de los años 90. Java es un lenguaje de programación orientado a objetos, con una sintaxis similar a la de C++. Los programas Java se ejecutan sobre una máquina virtual a través del bytecode, un código intermedio más abstracto que el código máquina. En 2007 Sun liberó el código fuente de la plataforma Java, convirtiéndolo en Software Libre casi en su totalidad. Java también presenta las siguientes características:

- Portabilidad. Puesto que los programas Java se ejecutan sobre la máquina virtual de Sun, las aplicaciones funcionan en cualquier sistema operativo para el que esté implementada la plataforma Java. Esto hace que el código sea independiente de la plataforma y evita crear código específico para cada sistema operativo que se quiera soportar. Por este mismo motivo, las aplicaciones Java también son independientes de la plataforma hardware.
- La gestión de memoria dinámica en Java se realiza de manera transparente al programador. Cuando se instancia un nuevo objeto, automáticamente se reserva el espacio en memoria necesario para este. En cuanto a la liberación, Java cuenta con un sistema de recolección de basura, que se encarga de detectar los objetos ya inaccesibles por el programa, y liberar el espacio que ocupan. Esto evita que el programador tenga que preocuparse de la gestión de memoria, evitando sus problemas derivados.

- Java cuenta con varios entornos de desarrollo, que integran múltiples características para facilitar ciertas tareas en el proceso de programación.
- Bibliotecas externas. El reciente auge de Java ha concentrado en gran medida desarrollo para esta plataforma, lo que hace que existan numerosas bibliotecas para extender la funcionalidad de las aplicaciones que se desarrollen. Además, gracias las características de la máquina virtual comentadas anteriormente, la integración de bibliotecas externas a una aplicación en desarrollo es sencilla.

Todas estas características, además de tratarse de un lenguaje utilizado a lo largo de la carrera, han motivado la elección de Java como lenguaje para la implementación de este proyecto.

2.5. Software de apoyo

Durante el desarrollo del proyecto se utilizaron además las siguientes aplicaciones:

- *NetBeans IDE 6.8 [7]* como entorno de desarrollo Java.
- *Gantt Project [8]* para elaborar el diagrama de Gantt del proyecto.
- *Dia 0.97.1* [9] para la elaboración de diagramas de casos de uso, diagramas de clases y trazas de eventos durante la fase de diseño.
- *Adobe Photoshop CS4* [10] para la generación y edición de los diferentes elementos gráficos tanto de la aplicación como de la memoria.
- *LaTeX* [11] para la elaboración de la documentación. Los componentes utilizados fueron:
  - *MiKTeX 2.8* [12] como distribución *TeX/LaTeX* para Microsoft Windows XP.
  - *LyX 1.6.5* [13] como entorno gráfico para escribir documentos en *LaTeX*.

Capítulo 3

Estado del arte

En este capítulo se estudia en primer lugar lo que se quiere conseguir con la herramienta y posteriormente se comentan las herramientas similares existentes con objeto de comprobar que ninguna de ellas cumple con todos los requerimientos.

3.1. Análisis de requerimientos

Para ser capaces de evaluar las distintas alternativas existentes debemos realizar en primer lugar un análisis de los requerimientos básicos que una herramienta de este tipo debería tener. Estos requerimientos han sido fruto del estudio detallado de las actividades realizadas por el alumno que comienza el aprendizaje del álgebra relacional, y nos da una lista de funcionalidades que deberían ser imprescindibles en nuestra aplicación. Así pues, los requerimientos que vamos a tener en cuenta para la comparación son los siguientes:

1. *Introducción y manipulación de consultas en álgebra relacional.* Es la característica fundamental ya que el alumno deberá aprender a base de introducir consultas y editarlas.

2. *Introducción y manipulación de árboles de expresiones en álgebra relacional.* Consideramos también fundamental la posibilidad de trabajar con los árboles de las expresiones puesto que representan una gran ayuda en cuanto a la comprensión de consultas no triviales.

3. *Optimización de consultas.* Es uno de los puntos que se tratan siempre que se habla del álgebra relacional y una vez comprendido es fácil aplicarlo a la elaboración de consultas en cualquier otro lenguaje.

4. *Ejecución de consultas (en modo automático o manual) en la propia aplicación o en SGBDs.* Para comprender las diferentes operaciones del álgebra relacional

es esencial que las consultas que introduzca el usuario se ejecuten y muestren resultados. Sería una gran limitación que solo se pudieran ejecutar estas consultas sobre SGBDs externos a la aplicación puesto que limitaría mucho su uso, por lo tanto también consideramos esencial la posibilidad de ejecutar las consultas sobre la propia herramienta.

5. *Traducción álgebra relacional - SQL.* Puesto que SQL es utilizado en la actualidad como lenguaje estándar de consulta, pensamos que sería de gran ayuda para el alumno la posibilidad de traducir sus consultas a este lenguaje y poder observar la relación existente entre ambos.

6. *Disponibilidad de interfaz gráfica.* El objetivo principal de la herramienta es que el alumno realice la labor de aprendizaje del álgebra relacional de una forma fácil e interactiva. Por lo tanto es necesario que la aplicación disponga de una buena interfaz gráfica.

Estos requerimientos se analizarán ahora para cada una de las herramientas con objeto de compararlas de una manera rigurosa.

3.2. Herramientas actuales

El álgebra relacional en la actualidad es utilizada en la mayor parte de los casos en entornos formativos. Por esto mismo no existen muchas herramientas en el mercado que la utilicen, y las que podemos encontrar se tratan en su mayoría de proyectos educativos que han sido desarrollados centrándose solamente en las funcionalidades que necesitan en dicho entorno. A continuación vamos a explicar brevemente algunas de ellas para, posteriormente, hacer una comparativa con la herramienta desarrollada en este proyecto.

3.2.1. Relational

Desarrollada como herramienta educativa en la *Facoltà di Scienze Matematiche Fisiche e Naturali dell'Ateneo di Catania*, *Relational [14]* dispone de una interfaz gráfica que permite cargar/guardar relaciones, ejecutar consultas sobre ellas y mostrar los resultados. Está desarrollada en Phyton y tiene una Licencia Publica General de GNU (GNU General Public License). Permite la optimización de consultas simples pero no la realiza en función de estadísticas provenientes de las relaciones. No permite la traducción entre álgebra relacional y SQL ni tampoco la obtención o definición de árboles de expresiones en álgebra relacional. Está disponible para Windows, MacOS y Debian (cada sistema operativo necesita un instalador). No se adjunta demasiada documentación y la ayuda al usuario es bastante pobre, lo que hace complicado el uso. En la Figura 3.1 se puede ver un ejemplo de esta herramienta.

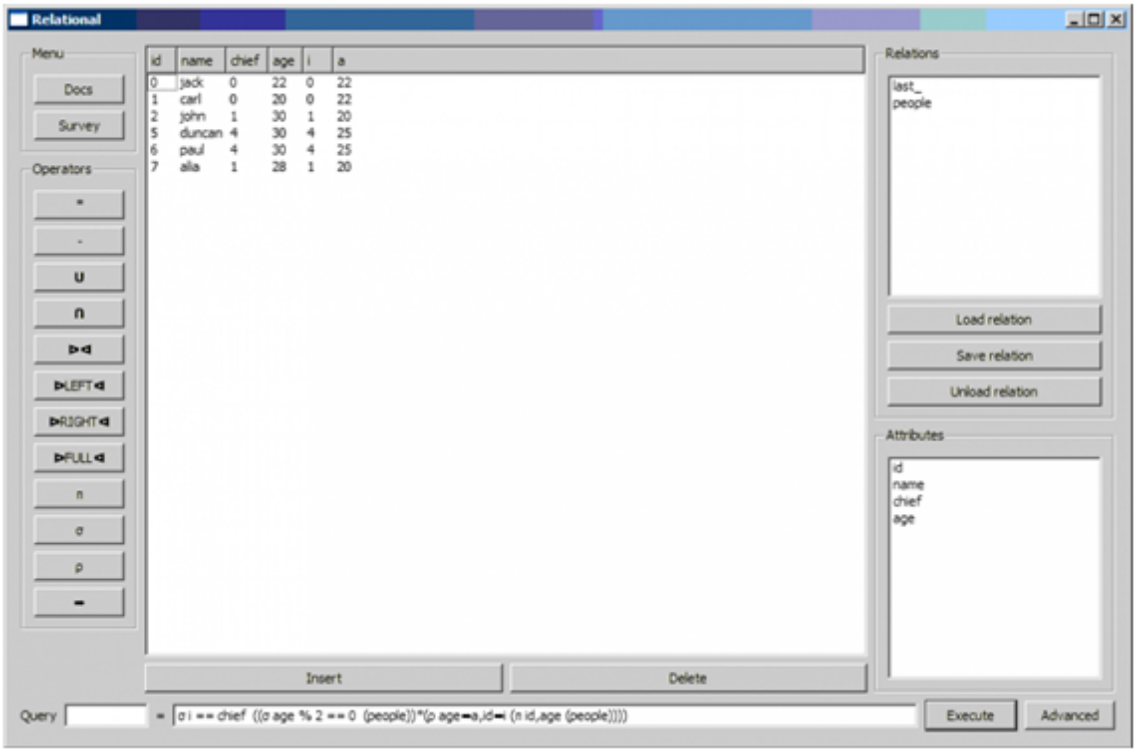

Figura 3.1: Aplicación *Relational*

3.2.2. LEAP

LEAP [15] se trata de una herramienta orientada a la educación desarrollada como proyecto fin de carrera en la *Oxford Brookes University*. Está desarrollada en C y tiene una Licencia Publica General de GNU (GNU General Public License). Se distribuye el código fuente y es necesario compilarlo lo cual puede representar algún problema. No tiene interfaz gráfica y se basa en el paso de comandos, el uso es difícil y cuesta identificar los resultados obtenidos. Además, presenta pocas ayudas para el usuario y no permite la optimización de consultas. Permite la obtención de árboles de expresiones en álgebra relacional, pero al carecer de interfaz gráfica son mostrados en modo texto, resultando confusos si las consultas son complejas. La documentación adjunta es bastante completa, lo cual es un punto muy favorable. La última versión de la herramienta data de 2005 y el proyecto parece estar abandonado.

3.2.3. WinRDBI

WinRDBI [16] es una herramienta educacional desarrollada en la *Arizona State University*. Su primera versión fue implementada en *Quintus Prolog*, pero posteriormente se pasó a *Java* con objeto de crear una interfaz gráfica amigable. Para poder descargar esta aplicación debe registrarse en su web para la obtención de una

autorización de descarga. Tiene interfaz gráfica que permite cargar/guardar relaciones, cargar/guardar consultas y, en comparación con las demás aplicaciones, la interfaz gráfica es muy completa y simplifica el uso. No dispone de ayudas para la introducción de las consultas. Permite la carga de relaciones a través de diferentes formatos de archivo. No permite la optimización de consultas. La documentación es muy completa y bien redactada. Trae algunos ejemplos incluyendo una base de datos y consultas. Dispone de instalador y al haber sido desarrollada en Java es multiplataforma. Se puede observar una captura de la aplicación en la Figura 3.2.

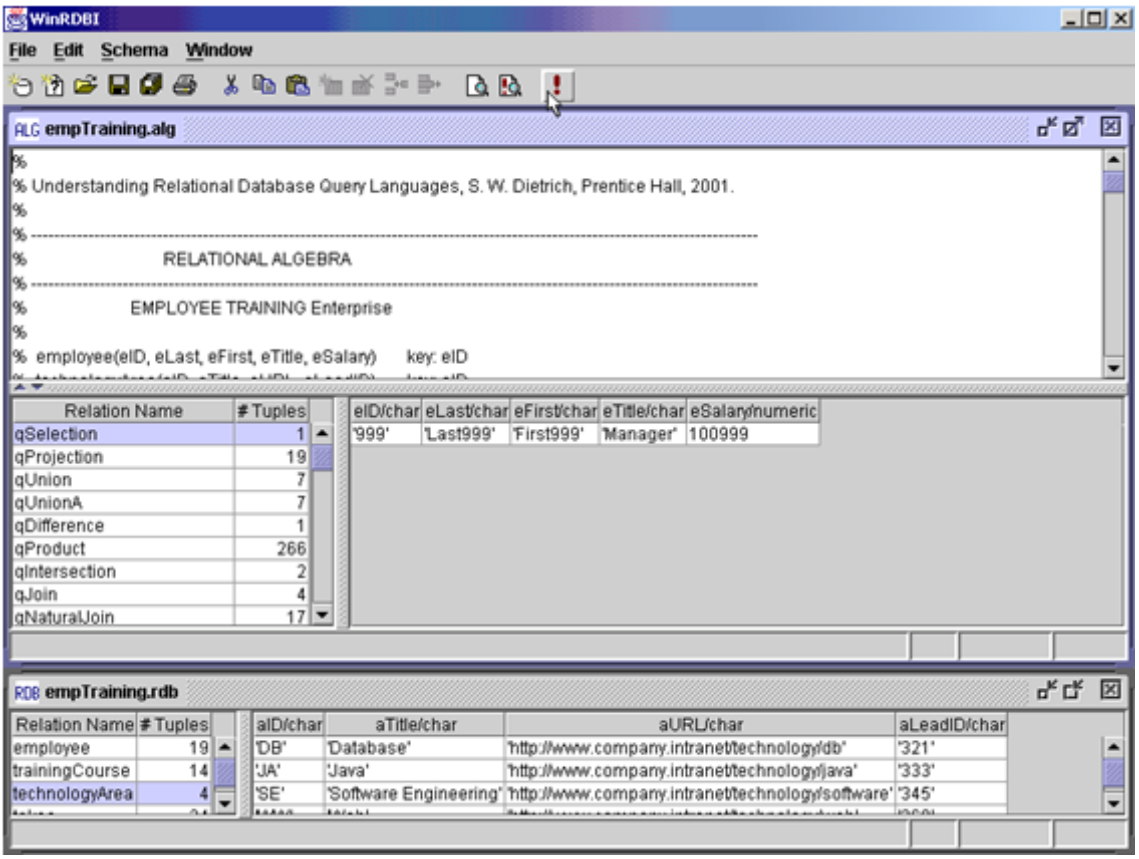

Figura 3.2: Aplicación *WinRDBI*

3.2.4. RelationalQuery

Desarrollada en la *Universidad de Sevilla*, RelationalQuery [17] es una herramienta educativa que posee una interfaz gráfica que permite almacenar/recuperar consultas y elegir el lenguaje en el que se va a trabajar (álgebra relacional, TRC o SQL). Permite la realización de consultas sobre una base de datos SQL (trae incorporado un driver JDBC para Oracle[4] aunque pueden utilizarse otros drivers). Básicamente se trata de un traductor de consultas entre diferentes lenguajes. Es multiplataforma

al ser desarrollada en Java. Actualmente el proyecto está abandonado. Puede verse la interfaz de la herramienta en la Figura 3.3.

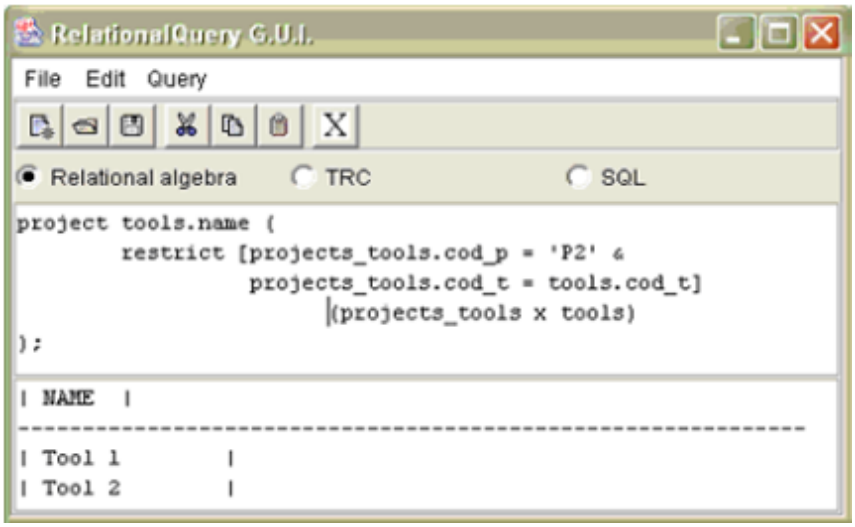

Figura 3.3: Aplicación *RelationalQuery*

3.2.5. Comparativa

Del estudio de las herramientas actuales se obtuvo bastante información interesante sobre los puntos fuertes y débiles de cada aplicación. Esta información fue trasladada al desarrollo de nuestra herramienta. Así pues, estamos en disposición de comparar los aspectos que se han tratado en la definición de los requerimientos, así como otros que se consideran deseables para una aplicación de este tipo, entre las herramientas estudiadas y la herramienta desarrollada en este proyecto. En las tablas de la Figura 3.4 se puede observar un resumen de dicha comparación. Es necesario explicar los resultados de las tablas, ya que es difícil representar gráficamente toda la información y el grado de cumplimiento de cada característica puede ser diferente en cada una de las aplicaciones.

En primer lugar, se consideró un punto fundamental del proyecto que la herramienta contara con una interfaz gráfica intuitiva y que ayudara al usuario en todo lo posible. Este punto es cumplido por casi todas las herramientas, exceptuando a *LEAP*, pero la interfaz de herramientas como la de este proyecto o la de *WinRDBI* resulta mucho más atractiva y fácil de utilizar que el resto.

Entrando en las funcionalidades, todas las herramientas permiten la introducción de expresiones en álgebra relacional, si bien es cierto que los operadores deben ser introducidos en forma textual en todas menos en la nuestra aplicación y en *Relational*. La introducción de los operadores en modo textual dificulta la lectura de las relaciones y puede confundir al usuario que esté aprendiendo el álgebra relacional.

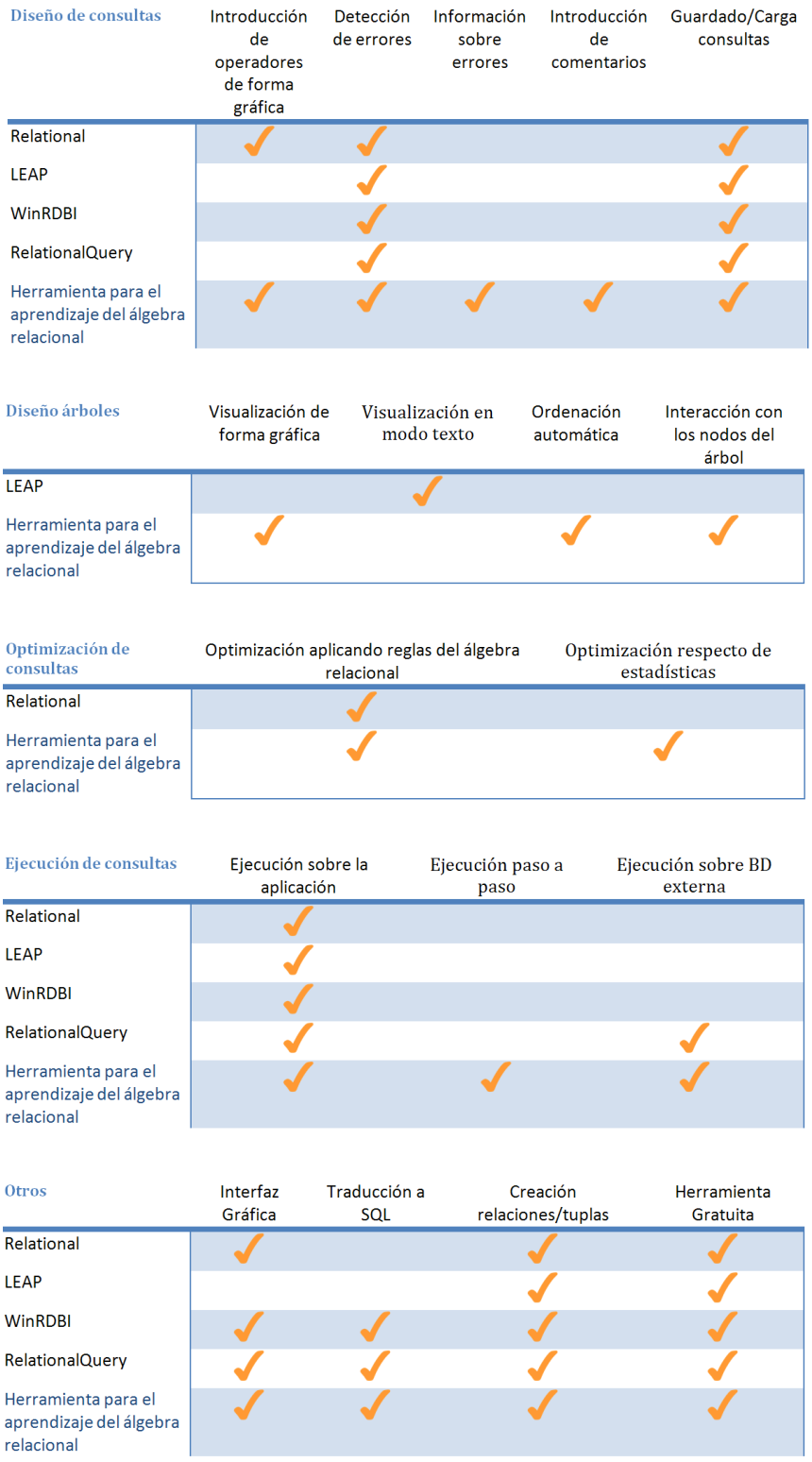

Figura 3.4: Tabla comparativa entre las herramientas

Otra de las funcionalidades importantes es la introducción y manipulación de árboles de expresiones en álgebra relacional. Solamente nuestra aplicación y *LEAP* permiten este aspecto, pero *LEAP*, al no disponer de interfaz gráfica, hace que la lectura de los mismos sea difícil y no se parezcan mucho a los árboles que los usuarios realizan en papel. También hay que decir que no es posible modificar los árboles en *LEAP* como ocurre en nuestra aplicación.

Las únicas aplicaciones que permiten la optimización de consultas son *Relational* y nuestra herramienta. *Relational* optimiza automáticamente consultas simples sin tener en cuenta las estadísticas de las relaciones. Nuestra aplicación sin embargo permite la optimización automática así como la interactiva, y tiene en cuenta estas estadísticas.

A la hora de definir las relaciones con las que el usuario va a trabajar, todas las herramientas permiten la obtención de los datos desde ficheros externos, pero el acceso a gestores de bases de datos externos solamente es posible en nuestra aplicación. Tampoco es posible la definición de relaciones o tuplas a través de la interfaz en todas las herramientas, *Relational*, *WinRDBI* y nuestra aplicación son las únicas que disponen de esta funcionalidad.

Capítulo 4

Desarrollo de la herramienta

Como se ha explicado en la sección 1.2, el principal objetivo del proyecto es diseñar una herramienta gráfica que permita al usuario:

- Definir y manipular expresiones en álgebra relacional y árboles de expresiones en álgebra relacional.
- Interpretar dichas expresiones permitiendo la ejecución de las mismas sobre relaciones.
- Traducir las expresiones en álgebra relacional a SQL.
- Optimizar las consultas de manera automática y guiada.
- Definir relaciones de ejemplo e introducir datos en dichas relaciones.

A la vista de estos objetivos y aplicando la metodología para el desarrollo software aprendida durante la carrera, se realiza el análisis de requisitos, diseño de la arquitectura del sistema, prototipado e implementación del mismo. En este capitulo se hace un resumen de las etapas seguidas, que son explicadas con más detalle en el Anexo C.

4.1. Análisis de requisitos

Después del estudio de las herramientas actuales y la naturaleza del problema se procedió a realizar el análisis de requisitos según el estándar *830-1988 IEEE* [18]. A continuación se muestran los requisitos de la aplicación (extracto del documento de *Especificación de Requisitos Software (ERS),* que puede ser consultado en el Anexo C):

1. Definición de expresiones de álgebra relacional. De forma "textual" o a través de ayudas gráficas.
2. Definición de arboles sintácticos de álgebra relacional.
3. Obtención de arboles sintácticos de expresiones de álgebra relacional, mostrados de forma gráfica y permitiendo su posterior manipulación.
4. Guardado/carga de expresiones de álgebra relacional.
5. Interpretación (automática o paso a paso) de expresiones de álgebra relacional sobre relaciones de ejemplo.
6. Guardado/carga de relaciones de ejemplo.
7. Conexión a bases de datos externas para la obtención de relaciones de ejemplo.
8. Introducción de datos en las relaciones.
9. Traducción de álgebra relacional a SQL.
10. Optimización (automática o paso a paso) de consultas introducidas por el usuario.
11. Imprimir árboles, relaciones y consultas.
12. Comprobación de la corrección de consultas y árboles.

4.2. Arquitectura propuesta

El diseño de la arquitectura del sistema se realiza de forma modular en función de los requisitos previamente analizados. En la Tabla 4.1 se puede ver la división de los requerimientos en diferentes categorías. De ellos se obtienen tres grandes secciones que deberán corresponder a los módulos que integrarán la aplicación. Estas secciones son:

- El sistema gestor de bases de datos externo.
- Los ficheros donde se guarden/carguen las consultas, relaciones y tuplas.
- Los elementos del álgebra relacional (relaciones, operadores, tuplas, árboles...)

Teniendo en cuenta que en una herramienta de este tipo la interfaz gráfica es muy importante, puesto que cuanto más intuitiva y fácil de utilizar más eficaz será en la ayuda al aprendizaje de los alumnos, se decide que este apartado tendrá un modulo propio en la arquitectura del sistema. También se decide que la categoría de

SGBD	Ficheros	Álgebra relacional
Importar/exportar relaciones	Cargar y guardar relaciones	Creación/edición de árboles
Ejecutar consultas	Cargar y guardar tuplas	Creación/edición de consultas
	Cargar y guardar consultas	Optimizar consultas
	Imprimir árboles/relaciones/consultas	Ejecución de consultas
		Comprobación corrección consultas

Tabla 4.1: Tabla de requisitos divididos por categoría

álgebra relacional (que abarca tanto las expresiones como los árboles de expresiones) estará compuesta a su vez por los módulos de ejecución, optimización y creación, puesto que dicha categoría comprende la mayor parte de las funcionalidades de la herramienta. Así pues, en la Figura 4.1 se puede ver el esquema gráfico de la arquitectura propuesta.

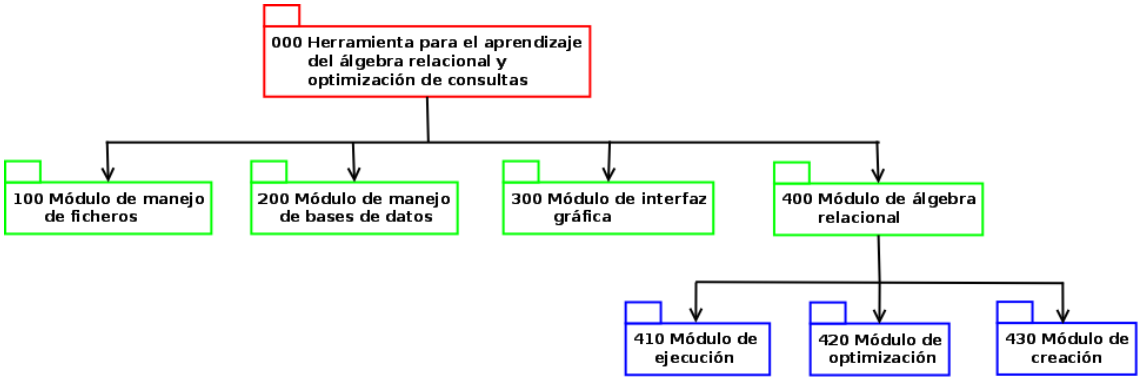

Figura 4.1: Arquitectura general de sistema

También se adjunta en la Figura 4.2 un diagrama que permite observar la comunicación entre los módulos diseñados. Comentando dicho diagrama podemos decir que el módulo de interfaz gráfica servirá como conexión entre los diferentes módulos del sistema. Se ha optado por esta decisión puesto que un control central de la comunicación nos permitirá gestionar de una forma mas adecuada los recursos y los posibles errores. De esta forma, el modulo de álgebra relacional no accederá directamente al sistema gestor de bases de datos ni a los diferentes ficheros externos, sino que será necesaria la comunicación con el módulo de interfaz gráfica para la extracción de la información. Es decir, la obtención de las relaciones (y sus tuplas) ya sea desde la base de datos o desde un fichero externo, será realizada por el módulo de interfaz gráfica (puesto que el usuario confirmará todas estas acciones a través de componentes visuales), y a continuación serán transferidos dichos datos al módulo de álgebra relacional correspondiente.

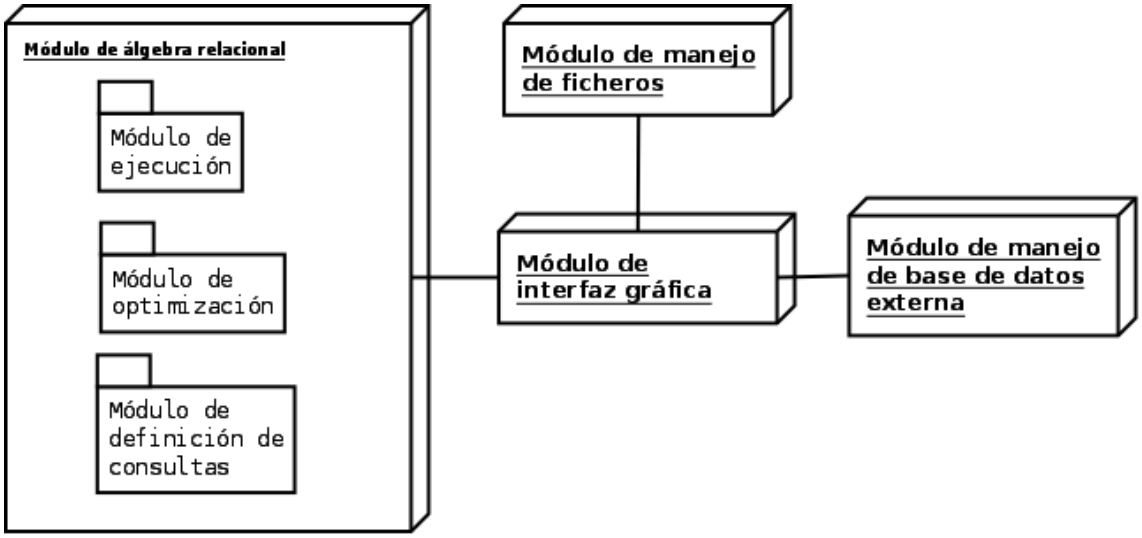

Figura 4.2: Comunicación entre los módulos del sistema

4.3. Prototipado

Durante el proceso de análisis se elaboran varios prototipos de la ventana principal con diferentes propuestas. Estos prototipos se comentan con el director del proyecto y tras un proceso de análisis de sus características se decide realizar un prototipo final que satisface nuestras expectativas. La Figura 4.3 recoge el prototipo final, y el resto pueden ser observado en el Anexo C.

4.4. Implementación

Tras el estudio previo y el proceso de análisis y diseño de la solución, se procedió a la fase de implementación. Dicha fase queda dividida en las siguientes etapas:

Interfaz gráfica

Se decide implementar en primer lugar la interfaz gráfica del sistema aprovechando para ello el estudio de las herramientas actuales, de tal forma que se seleccionan las características que durante el estudio parecieron positivas y se añaden aquéllas que se consideran necesarias para esta aplicación. Durante el desarrollo de la interfaz se tienen en cuenta aspectos de usabilidad de tal forma que se presenta la información de la forma mas accesible para el usuario. También se decide, en la medida de lo posible, presentar varias formas de acceder al contenido para que el usuario elija aquella con la que se sienta mas cómodo. Se incluyen elementos gráficos para acompañar las diferentes opciones y se disponen los elementos de forma lógica

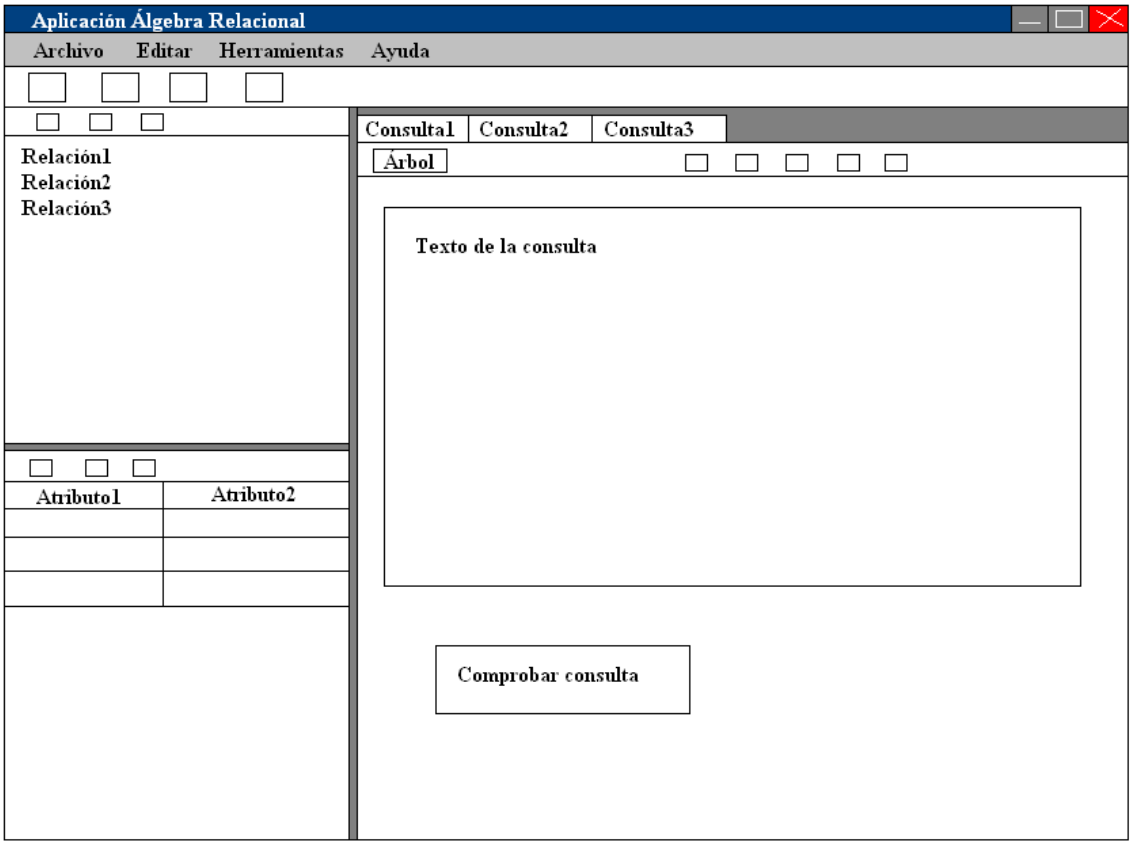

Figura 4.3: Prototipado de la ventana principal

y coherente entre las diferentes partes de la interfaz. Un aspecto importante es la posibilidad de permitir al usuario personalizar el aspecto de la ventana principal, permitiéndole de esta forma maximizar/minimizar/cerrar las diferentes partes de la ventana principal. Para aumentar la robustez se precisa de confirmación en las acciones destructivas como la eliminación de relaciones/tuplas/consultas o la apertura de nuevos proyectos.

Diseño de árboles

Una vez la interfaz gráfica era lo suficientemente potente como para poder cargar relaciones y tuplas, se decidió comenzar en paralelo a implementar las funcionalidades referentes a la definición y modificación de los árboles de expresiones en álgebra relacional. Así pues, se realizó la definición de las clases que harían falta para esto como pueden ser las clases encargadas de representar los árboles o sus nodos. Ninguna de las herramientas estudiadas permiten la creación de árboles en modo gráfico. Por lo tanto se diseño su interfaz totalmente desde cero, pensando en todo momento que el proceso debía ser similar a como se realiza en las clases y en los exámenes. Por ello se pensó que permitir al alumno ir insertando elementos en un espacio que sirviera

como lugar de trabajo, para posteriormente editarlos, relacionarlos, etc., podría ser una idea interesante. También se pensó desde un principio que en función de la complejidad de los árboles diseñados podría ser muy útil que la aplicación reordenara el trabajo del usuario para expresarlo de forma más clara. Durante esta etapa de diseño, se implementaron también los métodos necesarios para la extracción de las consultas de los árboles definidos, así como su ejecución.

Diseño de expresiones

El diseño de expresiones parecía a priori más fácil que el diseño de los arboles, pero resultó complicarse debido a la necesidad de analizar las consultas introducidas por el usuario para comprobar su corrección y poder ejecutarlas u obtener su árbol asociado. Utilizando los conocimientos adquiridos a lo largo de la carrera en el análisis léxico y sintáctico de expresiones, se realizaron los métodos que permitían completar las tareas antes descritas. Una vez realizado este proceso se comenzó a trabajar en las funcionalidades que permitían guardar/cargar consultas desde ficheros externos, así como imprimir dichas consultas, sus árboles, o las relaciones que intervinieran.

Optimización

Para la optimización de las consultas ha sido necesaria la definición de métodos capaces de aplicar a los árboles definidos reglas de optimización. Gracias a que en la fase de análisis se percibió que un buen diseño de las clases que representan los árboles y sus componentes facilitaría la tarea de optimización, la definición de estos métodos que implementan las reglas de optimización no ha sido costosa. Para la optimización en base a estadísticas de las relaciones ha sido necesario crear métodos que estimaran la carga que supone cada operación sobre las relaciones definidas, puesto que hasta el momento de la ejecución no se sabe con exactitud el tamaño de datos que va a mover cada una de ellas.

Traducción a SQL

La última funcionalidad importante implementada ha sido la traducción de las consultas a SQL. Una vez más gracias al diseño de las clases encargadas de gestionar los árboles, ha bastado con recorrerlos implementando métodos capaces de traducir cada una de las operaciones del álgebra relacional a su equivalente en SQL. También se ha abordado el tema de la creación de *vistas* puesto que es una parte importante en la formación que el alumno recibe en la asignatura de *Ficheros y Bases de Datos* y dependiendo de la complejidad de las consultas ayuda al usuario a aclararse durante su estudio.

4.5. Errores y problemas encontrados

Durante el desarrollo de la aplicación se han realizado pruebas de los diferentes componentes por separado y a la hora de su integración. También se realizaron pruebas de carácter general una vez la implementación hubo terminado, para comprobar el correcto funcionamiento de la aplicación. Gracias a las pruebas desarrolladas a lo largo de la implementación se resolvieron los problemas o errores que fueron encontrados.

La mayoría de los problemas encontrados tenían que ver con las diferentes tecnologías empleadas en el proyecto. El uso de bases de datos externas y ficheros externos hicieron que el control de errores fuera muy completo, para que en caso de fallo de alguno de ellos no se provocara el fallo total del sistema. También el uso de clases y métodos de Java, que no habían sido utilizados anteriormente durante la carrera, supusieron que se encontraran problemas con diferentes partes de la interfaz.

En las reuniones con el director del proyecto se encontraron también algunos problemas de usabilidad de la interfaz gráfica así como en las pruebas realizadas por algunos usuarios sobre la herramienta. Estos problemas se resolvieron de forma satisfactoria.

Capítulo 5

Conclusiones

En este capítulo se reflexiona sobre las conclusiones generales del desarrollo del proyecto, se apuntan algunas posibilidades de mejoras y trabajo futuro, y se termina con una pequeña conclusión a nivel personal.

5.1. Conclusiones generales

La conclusión más importante del desarrollo de este proyecto es que se ha conseguido, en el tiempo esperado, desarrollar una herramienta para la docencia que permite:

- Crear y modificar consultas en álgebra relacional a través de una interfaz gráfica sencilla e intuitiva.
- Crear y modificar árboles de expresiones en álgebra relacional de una forma visual y semejante al proceso realizado en papel.
- Ejecutar consultas (incluyendo un modo de ejecución paso a paso), tanto en la aplicación como en bases de datos externas.
- Optimizar consultas (incluyendo la optimización paso a paso) teniendo en cuenta tanto las reglas del álgebra relacional como las estadísticas de las tablas.
- Traducir consultas en álgebra relacional a SQL.
- Introducir relaciones y tuplas de ejemplo sobre las que ejecutar las consultas.

En resumen, se ha conseguido implementar todos los requisitos que se habían propuesto además de algunos que fueron surgiendo durante el estudio detallado del contexto de proyecto.

5.2. Posibilidad de mejoras y trabajo futuro

A continuación se exponen algunos de los posibles trabajos a realizar continuando con la línea del proyecto:

- La aplicación ayuda a la comprensión del álgebra relacional a través de explicaciones y de la interacción con relaciones de ejemplo. Para ello se requiere previamente un cierto conocimiento del tema. Se podría desarrollar alguna funcionalidad capaz de interpretar las consultas que el usuario expresara en lenguaje natural (o semi-natural), de tal forma que se permitiera al usuario introducir la consulta de una forma más familiar para él y esto facilitara su aprendizaje.

- No era el objetivo de esta aplicación tratar las bases de datos distribuidas, pero podrían implementarse las funcionalidades necesarias para realizar la optimización de consultas teniendo en cuenta que se podría dar el caso de que la información que va a ser consultada se encontrase en múltiples bases de datos lógicamente relacionadas, de tal forma que con una buena gestión se pudiera optimizar el tiempo de ejecución.

- Se ha implementado la optimización automática y paso a paso de las consultas introducidas por el usuario, pero no era el objetivo de este proyecto permitir la selección de las diferentes heurísticas a aplicar y podrían realizarse las modificaciones oportunas para darle la oportunidad al usuario de seleccionarlas.

5.3. Conclusión personal

El desarrollo de este proyecto me ha aportado conocimientos que serán útiles en mi vida profesional. Conocimientos técnicos en cuanto al problema tratado, el álgebra relacional y las bases de datos, y en cuanto a las herramientas utilizadas, y otros conocimientos como son el desarrollo de software y el trato con el cliente.

En cuanto a los conocimientos técnicos me gustaría destacar que gracias al estudio realizado a lo largo del proyecto he llegado a comprender mucho mejor la importancia del álgebra relacional. Después de realizar este proyecto percibo mejor la relación existente entre el área matemática y las bases de datos. Sin comprender los conocimientos propios del álgebra es muy difícil entender correctamente ya no sólo el álgebra relacional sino las bases de datos relacionales en general, puesto que para su definición se utilizaron conceptos matemáticos. También he aprendido bastante acerca de Java, durante la carrera lo había utilizado para la realización de varias prácticas, pero no había necesitado hacer un uso tan exhaustivo de la documentación como en este caso. La necesidad de utilizar clases y métodos que no conocía me ha

hecho aprender a consultar correctamente la documentación disponible y he aprendido bastante acerca de un lenguaje que creo que será fundamental los primeros años de mi vida profesional.

Como decía, este proyecto también me ha aportado una serie de conocimientos como podrían ser el trato con el cliente (en este caso el papel de cliente ha sido interpretado por el director de proyecto y los usuarios que han realizado pruebas de la herramienta) y el desarrollo de un proyecto software de tamaño medio (en el que se han empleado casi siete meses). El trato con el cliente es una de esas cosas que no se estudian en la carrera pero que considero muy importantes de cara a mi vida laboral. Se han realizado reuniones semanales con el director en las que se ha presentado el trabajo de forma que entre los dos se han encontrado problemas y posibles soluciones.

En resumen, el proyecto ha hecho que consiguiera adquirir conocimientos interesantes de cara a mi futuro profesional y además desarrollar ciertas habilidades, como dar soluciones a problemas que aparecen en el transcurso de un proyecto real, que a priori no se dan en trabajos desarrollados en la formación de la carrera.