Capítulo 6

Relational Algebra Translator

Contenido de este capítulo

Funcionamiento del RAT 1
Componentes de un compilador 1
Componentes del software RAT 2
Pre-analizador de asignación 3
Analizador Sintáctico 5
Conexión con una base de datos 6
Instalar el RAT 7
Herramientas del RAT 8
Ejemplos de sentencias 9

"Con números se puede demostrar cualquier cosa."

Thomas Carlyle

1 FUNDAMENTOS DEL RAT

¿Qué es el RAT? Son las siglas del software “Relational Algebra Translator“ este software fue desarrollado en la Universidad Nacional de Costa Rica. El RAT es básicamente como su nombre lo dice, un traductor de álgebra relacional (A.R.) a sentencias SQL, el software viene acompañado de una seria de herramientas que facilitan al estudiante el aprendizaje del hasta entonces teórico tema del algebra relacional.

Este capítulo está destinado a exponer brevemente las técnicas usadas para programar este software, también en este capítulo se profundiza sobre el funcionamiento del software como herramienta pedagógica; su descarga no tiene ningún costo y está disponible en la dirección http://www.slinfo.una.ac.cr/rat/rat.html.

Para la creación de este software se requiere de conocimientos básicos en el desarrollo de compiladores/traductores, si el lector quiere profundizar en este tema puede visitar la bibliografía [1], también se recomienda leer [2] para los temas de gramáticas libres de contexto y derivaciones.

El producto de software Relational Algebra Translator (RAT) implementa los operadores originales del álgebra relacional (pi. sigma, producto cartesianos, producto natural y ro) además cuenta con operadores lógicos como el Y y el O lógicos, por ultimo implementa los operadores conjuntistas (diferencia, unión y la intersección); observe en la imagen 1 del programa la consulta ingresada en el primer campo de texto; en el segundo campo de texto se encuentra la traducción para esa misma consulta ahora en SQL.

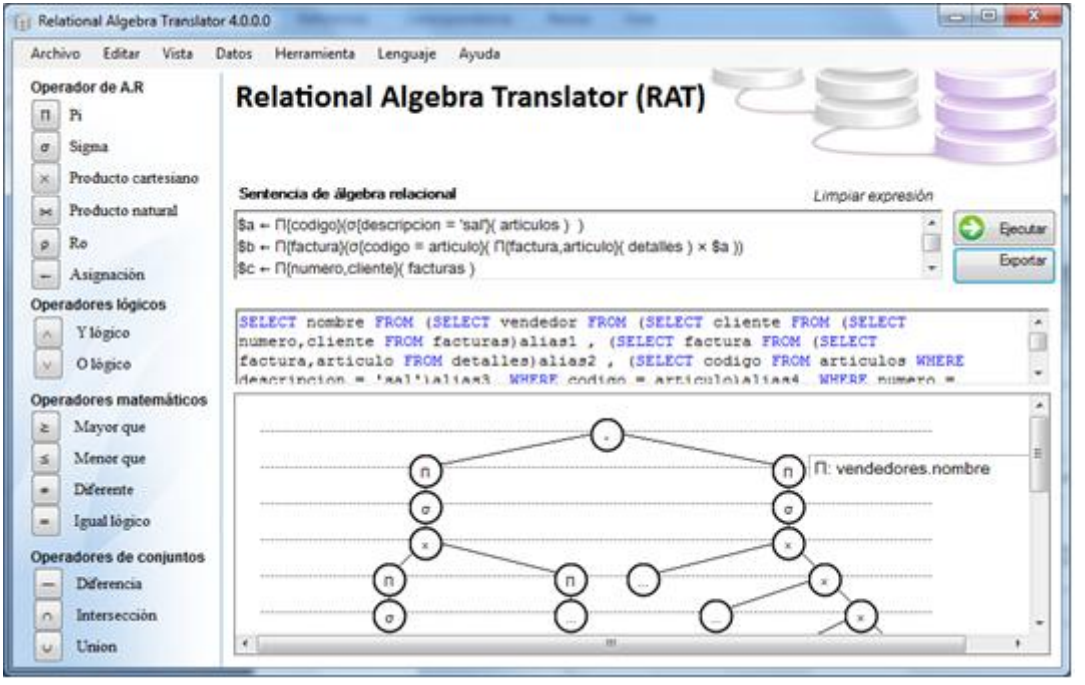


Figura 6.1: RAT 4.0

Debajo de la traducción al SQL, se encuentra el árbol de parser grafico, nótese que los nodos de las hojas están siempre equilibrados y representa el crecimiento de la consulta desde las tablas originales hasta los nodos de los operadores ubicados en la parte superior.

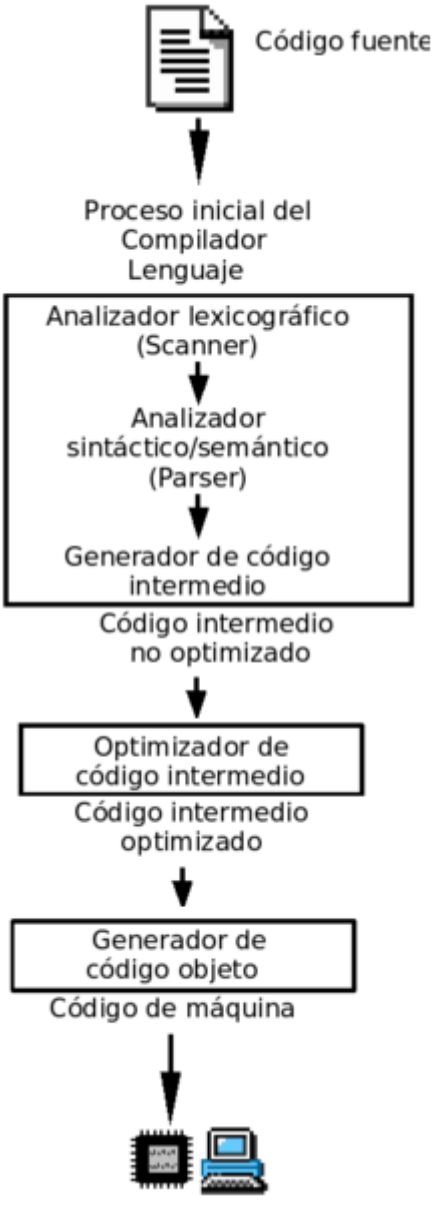
Este software puede ser utilizado para enseñar álgebra relacional de forma muy especializada, pero también podría ser utilizado como una valiosa herramienta cuando se enseñe teoría de conjuntos; pues facilita realizar operaciones conjuntistas como la intersección, unión o diferencia de conjuntos o combinaciones de ellas.
También es útil porque puede aplicársele productos cartesianos y así formar nuevos conjuntos a partir de los ya existentes. Finalmente el operador sigma facilita la escritura de conjuntos mediante el axioma de compresión, es decir declarando una propiedad que tengan los elementos del conjunto.

1.1 Origen del RAT

El problema de enseñar álgebra relacional es su alto nivel de abstracción, prácticamente es un lenguaje de consulta teórico; mientras que otros lenguajes de consulta como el SQL sí permiten realizar pruebas y obtener resultados de forma natural en las bases de datos, el álgebra relacional sin embargo, no dispone de un mecanismo que permita interactuar con las bases de datos comerciales de la actualidad. La idea de crear una herramienta que facilitara a los estudiantes del curso Diseño e Implementación de Bases de Datos de la carrera Ingeniería en Sistemas de la Universidad Nacional de Costa Rica parte de la tesis que, los estudiantes logran un aprendizaje significativo en el tema si prueban las soluciones a los problemas que se les presentan contra datos reales, esto disminuye el nivel de abstracción y reafirma los principios teóricos del modelo relacional que es uno de los ejes principales del curso.

Se establecen los requerimientos de la herramienta, tomando en cuenta la tecnología actual. La herramienta deberá ser capaz de traducir el álgebra relacional a un lenguaje que sea entendido no sólo por los estudiantes sino por las bases de datos, se establece por lo tanto el lenguaje SQL como el lenguaje objetivo a ser traducido. El formalismo y la simbología de los operadores del álgebra relacional se deben garantizar, por lo que la herramienta necesariamente tendrá los operadores oficiales (sigma, pi, ro, uniones, intersecciones y diferencia de conjuntos) a disposición del usuario. El último requerimiento se relaciona en la representación gráfica de las operaciones en el álgebra relacional, el sistema debe ser capaz de representar mediante un árbol de parser la construcción de las sentencias durante la traducción.
Una vez establecidos los requerimientos del sistema, se escoge el lenguaje C# para implementar la solución informática. Se descomponen los módulos del sistema en cuatro grandes partes: analizador lexicográfico, analizador sintáctico, analizador semántico y el traductor. Los tres primeros componentes son los que garantizan que las operaciones e instrucciones del álgebra relacional que se introducen en el sistema estén formadas correctamente y no procesar sentencias con errores de semántica.

2 COMPONENTES DE UN COMPILADOR



En general se puede decir que todo compilador es un traductor, puesto que al compilar un software el sistema se ve obligado a realizar una traducción a otro lenguaje (por ejemplo de Java a Ensamblador) para que la computadora lo "entienda", el proceso de la compilación inicia con un lenguaje determinado y finaliza cuando es traducido a lenguaje maquina.

Hagamos la traza del proceso de compilación para un simple ejemplo escrito en C++, todo el proceso inicia cuando se tiene un archivo fuente, en este archivo (.CPP o .H) están escritas las instrucciones de alto nivel que se desean compilar, recordemos que normalmente son escritas en Entornos de Desarrollo Integrados (IDEs), que son en el trasfondo editores de texto que facilitan la labor de la escritura, estos pueden ser tan simples como el Block de Notas o tan complejos como editores comerciales como Visual Studio de Microsoft. Es decir, el lenguaje C++ es independiente del IDE, por lo tanto las reglas gramaticales son definidas por una gramática formal y no por un compilador, de ahí la existencia de tantos compiladores para un mismo lenguaje.

Una vez se le da la orden de compilar un archivo .CPP o .H, el primer paso es ejecutar el analizador lexicográfico, cuya función es trasformar el flujo de *string* (cadenas de texto) en tokens (objetos que alimentan al proceso de compilación) para su posterior análisis en la traducción, para ello usa reconocimiento de patrones como gramáticas regulares o autómatas finitos. En esta fase también se verifica que los caracteres pertenezcan al alfabeto de la gramática, por ejemplo en C++ no está definido el símbolo µ por lo que una expresión como 4 µ 5 generaría un error lexicográfico. Hace años los programadores debían tener mucho cuidado con los identificadores de variables, pues el

compilador sólo aceptaba caracteres ASCII dejando sin funcionamiento variables como "niño", "año" o "día", pues no representaban caracteres ASCII. Sin embargo hoy en día la mayoría de compiladores usa UNICODE permitiendo efectivamente programar sin problemas en muchos idiomas como el español.

La segunda etapa, se denomina analizador sintáctico, la función es este analizador es verificar la sintaxis es decir las reglas definidas en un leguajes, mediante algoritmos aplicados a **arboles de análisis sintácticos** o para casos más simples **matrices gramaticales** un ejemplo trivial, es determinar las reglas pasa saber si una expresión es un número, como se muestra seguidamente en el formato BNF:

```
<numeral>       ::=  0|1|2|3|4|5|6|7|8|9
<exp_numerica>  ::=  <exp_numerica><exp_numerica> | <numeral>
```

Esta regla es capaz de generar todos los números naturales. Una tercera etapa llamada analizador semántico verifica la semántica de los tokens, es decir el ámbito para determinar la presencia de errores. Tomemos como ejemplo el siguiente segmente de código ***semánticamente*** correcto pero ***sintácticamente*** inválido.

```
int x = "hola mundo";
```

Como puede verse, efectivamente esta asignación es válida semánticamente porque está formada de: <tipo><nombre> = <valor>, pero por otro lado, su ámbito sintáctico no permite que un int le sea asignado una variable de cadena.

Las dos últimas etapas de la compilación son la optimización y generación de código, la mayoría de compiladores optimizan el código para realizar ejecutables más eficaces y eficientes sin embargo no es condición necesaria, por lo que algunos compiladores no optimizan sus sentencias pero todo compilador debe generar código traducido a lenguaje maquina. La diferencia con los traductores como lenguajes HTML o JavaScript es que no realizan ninguna de estas dos etapas.

3 COMPONENTES DEL SOFTWARE RAT

El RAT comparte muchos de los pasos típicos de un compilador, existen básicamente cuatro etapas bien definidas que usan técnicas aplicadas al desarrollo de un compilador. Como se puede deducir por la imagen 6.3 el RAT aplica teoría de conjuntos, matrices gramaticales, geometría analítica, recursividad, gramáticas formales, notación BNF y algoritmos recursivos para resolver los problemas de traducción.

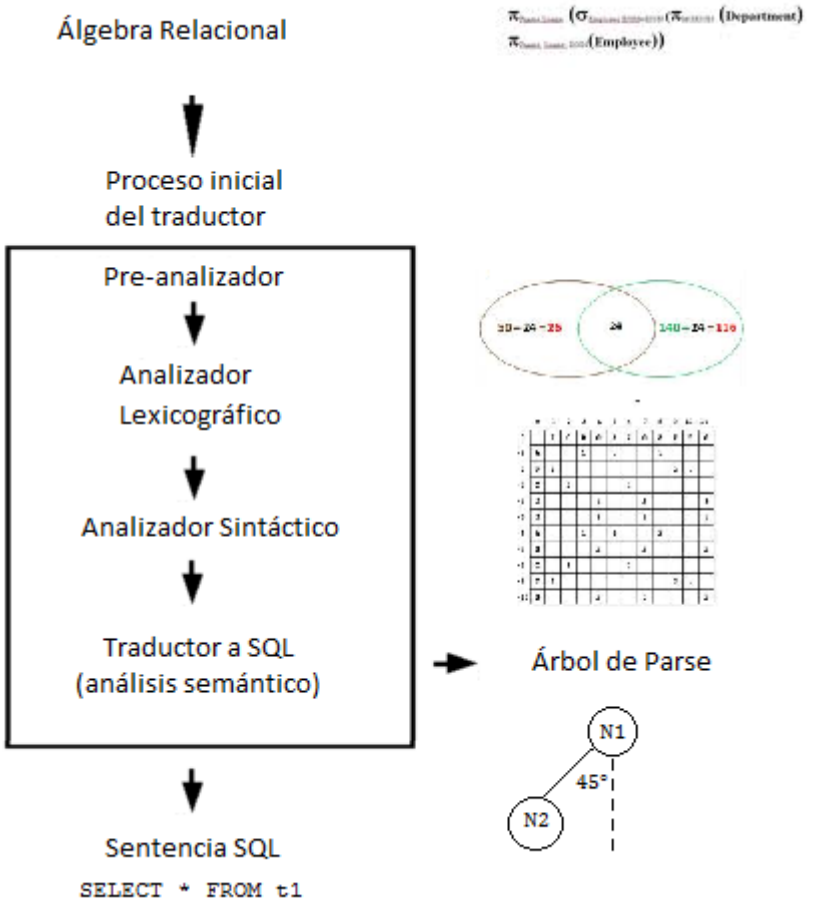


Imagen 6.3: Traducción del RAT

Seguidamente se definirán las etapas que discrepan de un proceso de compilación explicado anteriormente, por ejemplo el analizador lexicográfico no será explicado.

3.1 Pre-analizador de asignación

Toda sentencia en el Álgebra Relacional, está formada a lo sumo de una línea, esto es muy diferente a los demás lenguajes de programación donde la tendencia es tener muchas líneas de código para hacer la misma función que una línea en el SQL. Lo primero que se debe hacer es transformar las sentencias de múltiples líneas a una única sentencia evaluable.

Tomemos por ejemplo, la sentencia de asignación en A.R:

```
X ← tabla1
Y ← tabla2
W ← X × Y
```

Es evidente, que en el A.R. es una expresión valida, sin embargo no lo es para el SQL cuya traducción es `SELECT * FROM tabla1, tabla2`,esto significa encontrar una solución algorítmicamente simple para este problema. El RAT destina el Preanalizador para lograr crear una única línea que sea definida en término de otras variables. La solución propuesta resulta de la estructura de datos más eficiente que existe, las tablas Hash (también llamadas Maps) pues si complejidad algorítmica es O(1).

Recordemos que las tablas Hash asocian una única llave con su respectivo contenido, el RAT usa como llave el nombre de las variables y el contenido será otro string desde la variable hasta el salto de línea (**\n**), es decir tendría la forma de la imagen 6.4.

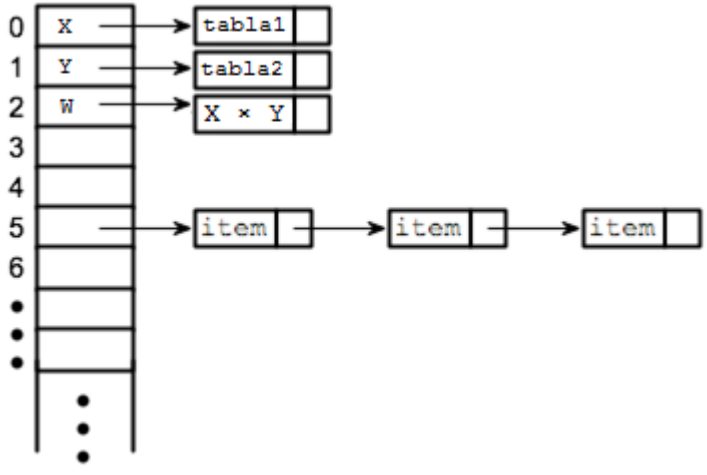


Imagen 6.4: Hash Tables

3.2 Conceptos teóricos del RAT

La propuesta es un software que combina las técnicas de compiladores, matemáticas discretas, estructura de datos y técnicas avanzadas de diseño de algoritmos como la recursividad sobre árboles binarios.

Las técnicas de compiladores, estudian el diseño de aquellos programas que permiten traducir un lenguaje origen a un lenguaje destino [1], normalmente cuando se hablan de compiladores el lenguaje destino suele ser el lenguaje maquina (código binario), sin embargo la definición no limita la existencia de un lenguaje diferente a este último. Se plantea entonces un software con características de compilador que permita traducir de álgebra relacional a un lenguaje que las bases de datos entiendan (SQL); cabe rescatar que las bases de datos que usan el SQL son sistemas basados en la teoría de relaciones e-narias, funciones y teoría de conjuntos; motivo por el cual la traducción entre los lenguajes es natural y evita la perdida de funcionalidad.

Existen dos tipos básicos y reconocidos de lenguajes: los lenguajes naturales y los lenguajes formales; los primeros se fueron construyendo con el paso del tiempo son por ejemplo, el español o el ingles. Por otro lado los lenguajes formales se basan en normas matemáticas tales como la lógica y teoría de conjuntos.

Definición 1. Se define un lenguaje como un conjunto de palabras. Cada lenguaje está formado por secuencias de símbolos (palabras) tomados de alguna colección finita llamada alfabeto.

Definición 2. Se define un alfabeto como un conjunto no vacio y finito de símbolos, entonces escribimos $\sum$ un alfabeto y $\sigma \in \sum$. Una secuencia finita de símbolos de un alfabeto, se llama palabra, expresión o cadena.

Definición 3. Una palabra es un conjunto finito de símbolos $\{\sigma_1, \sigma_2, \dots, \sigma_n\}$, con la concatenación de palabras formaremos un conjunto denotado por $\sum^*$, luego el conjunto de palabras que tengan significado se llama diccionario.

Desde el punto de vista técnico, un compilador además se ser un traductor es un programa que verifica que la sentencia de entrada sea válida. Una sentencia valida es aquella expresión lexicográficamente, semánticamente y sintácticamente correcta. El software propuesto debe cumplir estas validaciones.

Definición 4. Una expresión es lexicográficamente correcta si, y solo $\forall x, (x \in exp)(x \in \sum)$, con exp una palabra.

Ejemplo 1. Se define el alfabeto del álgebra relacional por T = {0, 1, 2, …, 8, 9, _, a, b, c, …, A, B, C ,…, Z, <, >, ≤, ≥, ≠, =, ∨, ∧, ∩ ∪, -, ⋈ ×, σ, Π, "," ρ, ε} donde ε representa un carácter nulo.

La utilidad del analizador lexicográfico, es la detección temprana de errores de tipo alfabético; por ejemplo la expresión exp = (valor = 4)(@) no es una expresión lexicográficamente valida, pues exp ⊄ T, en particular @ ∉ T.

Definición 6. Se establece una gramática usando la definición de Noam Chomsky [3], compuesta por la siguiente cuaterna:

1. Un conjunto finito de símbolos, que representa la totalidad del alfabeto de un lenguaje. Los componentes reciben el nombre de terminales o símbolos terminales.
2. Un conjunto de variables, a estas variables se les denomina no terminales o categorías sintácticas, estos no terminales se forman por otros símbolos no terminales o no terminales.
3. Existe un símbolo inicial, esta variable representa el inicio de la gramática, además debe ser un símbolo no terminal.
4. Un conjunto finito de producciones o reglas, que representa una definición recursiva de lenguaje. Cada regla está conformada de:
    a. Una variable llamada cabeza, es un identificador de la regla en particular.
    b. El símbolo de producción (→).
    c. Una cadena de cero o más símbolos terminales o no terminales, a este bloque se le llama cuerpo de la producción, en la cual cada elemento se sustituye recursivamente para formar una derivación.

Estas cuatro componentes definen una tupla llamada, gramática libre de contexto, también recibe el nombre de gramática o CFG. De la forma G = (V,T,P,S) donde V son las variables no terminales, T son los símbolos terminales, P son las reglas de Producción y la S es el símbolo inicial.

Ejemplo 2. Se define la gramática para el álgebra relacional la tupla G = (V, T, P, S) con V = {literal, numeral, símbolos numéricos, símbolos lógicos, símbolos conjuntos, variable, numero, parámetro, proyección, condicional, renombramiento, expresión, expresión lógica, expresión numérica}. El conjunto T o alfabeto del lenguaje se define como T = {0, 1, 2, …, 8, 9, _, a, b, c, …, A, B, C ,…, Z, <, >, ≤, ≥, ≠, =, ∨, ∧, ∩ ∪, -, ⋈ ×, σ, Π, "," ρ, ε} donde ε representa un carácter nulo. Tenemos también el símbolo inicial S = expresión, definido en las reglas de producción de P.

Para el conjunto P se definen las siguiente reglas de producción, para efectos de simplificar la notación se usara el formato Backus-Naur form (BNF) [2], donde los símbolos de producción (→) y sus respectivos cuerpos se agrupan con el símbolo ::= y separamos los cuerpos de las producciones por el símbolo (|) para identificar que son de la misma regla. A continuación se detalla el conjunto de Producciones:

```
<literal>            ::= _|a|b|c|…|A|B|C|…|Z
<numeral>            ::= ,|-|0|1|2|…|8|9
<simbolosNumericos>  ::= +|-|*|/
<simbolosLogicos>    ::= ∨|∧|<|>|≤|≥|≠|=
<simbolosConjuntos>  ::= ∩|∪|-|⋈|×
<variable>           ::= <literal> <variable> | <variable>
<número>             ::= <literal> <variable> | <variable>
<parámetros>         ::= <variable> ,<parámetros> | <variable>
<proyección>         ::= Π ( <parámetros> ) ( <expresión> )
<renombramiento>     ::= ρ ( <literal> ) ( <expresión> )
<condicional>        ::= [<proyección>]σ(<expresiónLogica>)(<expresión>)
<expresiónLogica>    ::= <expresiónNumerica> <simboloLogicos> |
                         <expresiónNumerica>
<expresiónNumerica>  ::= <variable> <simbolosNumericos> <número> |
                         <número> <simbolosNumericos> <variable>
<expresión>          ::= [(]<expresión>[)]
                         <simbolosConjuntos>[(]<expresión>[)]      |
                         <proyección>  | <condicional> | <variable>
```

■

Definición 8. Una expresión es semánticamente correcta si, y solo la expresión es producida por alguna regla de producción.

La importancia del analizador semántico, es la detección de errores estructurales; esto le permite al software saber que la expresión que introduce el usuario está formada correctamente; pongamos un ejemplo πy + +8, esta expresión es lexicográficamente valida, pues todos los símbolos de la expresión son elementos del alfabeto; el error se encuentra en el "+ +" pues se esperaba un número o una variable después del primer signo de mas.

Definición 9. Se define una matriz gramatical o matriz de adyacencia, como una matriz cuadrada que permite representar una relación binaria.

Ejemplo 3. Matriz Gramatical del álgebra relacional

Para efectos del verificador sintáctico, se desarrolla una matriz gramatical para crear y evaluar todos los posibles escenarios. Esta matriz booleana está formada por todos los símbolos no terminales tanto horizontal como verticalmente. La matriz completa del álgebra relacional al ser tan grande no se adjuntará, sin embargó vamos a ejemplificarla con los símbolos Pi, Sigma, paréntesis, literal y número (constante).

		SIGMA	LITERAL	CONSTAMTE	PARENTESIS ABIERTO	PARENTESIS CERRADO
PI		0	0	0	1	0
SIGMA		0	0	0	1	0
LITERAL		0	0	0	0	1
CONSTANTE		0	0	0	0	1
PARENTESIS ABIERTO		1	1	1	1	0
PARENTESIS CERRADO		0	0	0	1	1

Tabla 1 Matriz gramatical (adyacencia) de ejemplo.

Nótese que esta matriz también recibe el nombre de matriz de adyacencia, permite modelar un grafo, es decir determina si existe un camino que permita pasar de un símbolo a otro, por ejemplo Sigma | Pi → paréntesis abierto o después de un literal se puede cerrar un paréntesis (Literal → paréntesis cerrado).

■

Definición 10 Un árbol consta [4] de un conjunto finito de elementos, llamados nodos y un conjunto también finito de líneas dirigidas o vértices que conectan los nodos. También se puede definir como:

1. Una estructura de datos vacía.
2. Una conjunto con uno o más nodos tales que:
   a. Hay un nodo llamado raíz
   b. Todos los demás son subárbols de la raíz, que son arboles a la vez.

Observe que la definición del árbol es recursiva, también los algoritmos de esta estructura se suelen manejar recursivamente. La verdadera importancia de los arboles binarios son su capacidad para modelar las expresiones de álgebra relacional.

■

4 MANUAL DE USUARIO DEL RAT

4.1 Requerimientos de instalación

El software requiere tener un sistema operativo Windows 2000 SP4, Windows Xp, Windows Vista o Windows 7, para poder ejecutarse la aplicación el sistema debe tener instalado el Framework 2.0 o superior el cual se pude descargar del sitio web en el caso que la computadora no lo tenga ya instalado. Desde Windows Vista ya viene el Framework pre instalado.

Los requerimientos mínimos del software son:

a. Procesador a 1Hghz
b. 250 Mb de memoria RAM
c. 5 Mb de disco duro libre

Los requerimientos recomendados del software son:

a. Procesador a 1Hghz
b. 250 Mb de memoria RAM
c. 5 Mb de disco duro libre
d. Una base de datos instalada
e. Conectores ODBC para la base de datos que requiera

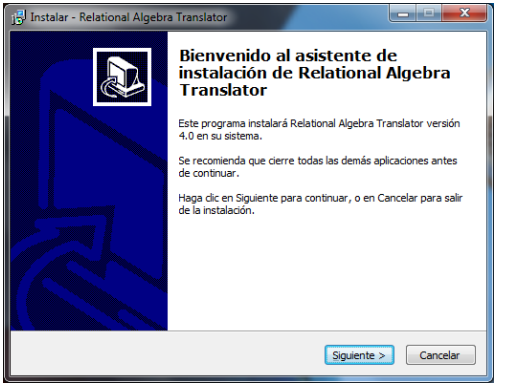


Imagen 6.5: Instalador del RAT

4.2 Instalación del Software

Para poder instalar el software primero es necesario descargarlo sin costo alguno del sitio oficial: **`http://www.slinfo.una.ac.cr/rat/descargas/descargas.html`**, una vez que tanga descargado la aplicación, deberá ejecutar el archivo descargado. Iniciara entonces el instalador de la aplicación como se ve en la figura siguiente. Finalice el proceso y ejecute la aplicación, para iniciar la configuración.

4.3 Sobre las funcionalidades del RAT

La primera ventana que aparece cuando se ejecuta la aplicación, es el menú principal el sistema el cual permite llamar a las demás aplicaciones del software. El RAT está compuesto por cuatro aplicaciones principales.

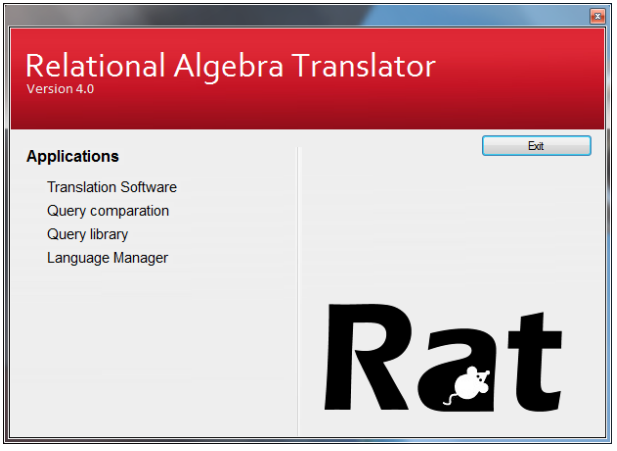


Imagen 6.6: Menú principal del RAT

Observe que la primera vez que se ejecuta el software, va aparecer traducido al ingles; sin embargo el RAT tiene soporte oficial para cuando idiomas (ingles, español, italiano y alemán) basado en el número de países que lo descargan.

La primera aplicación que le va salir es "Translation Software" esté es el núcleo de todo el software, es esencialmente el subprograma que va traducir las consultas de álgebra relacional a consultas SQL, además es el encargado de coger las sentencias SQL y realizar el llamado a las bases de datos.

Para fines académicos y de producción resulta conveniente saber cual consulta es más eficiente entre varias consultas que se tengan, para ello el RAT incorpora una métrica que permite comparar la eficiencia entre dos consultas de Álgebra Relacional. Para ello puede ingresar al segundo programa llamado "Query comparation".

Como el RAT va ser una herramienta que usted use diariamente para sus estudios o para optimizar consultas en su trabajo, es necesario incorporar la posibilidad de guardar las consultas que usted efectué. Se incorporó entonces una especie de biblioteca de consultas, una vez guardada la sentencia es fácilmente recuperable en el futuro. Para poder observar la biblioteca (inicialmente vacía) debe ingresar a "Query library".

Finalmente y por ser el RAT una herramienta académica sin fines de lucro, se espera que sea utilizada por la mayor cantidad de universidades e instituciones a nivel mundial. Como es un proyecto sin financiamiento se provee una herramienta para poder traducir el software entero al idioma que el usuario final desee. Se les agradecería mandarnos sus traducciones para mejorar las actuales y agregar nuevas, se respeta altamente los derechos de autor por lo que si colabora con el proyecto, su nombre va estar en el sitio web de

descargar como colaborador. Ingrese a "Language Manger" para poder agregar nuevos idiomas.

4.6.1 Elementos de interfaz del RAT

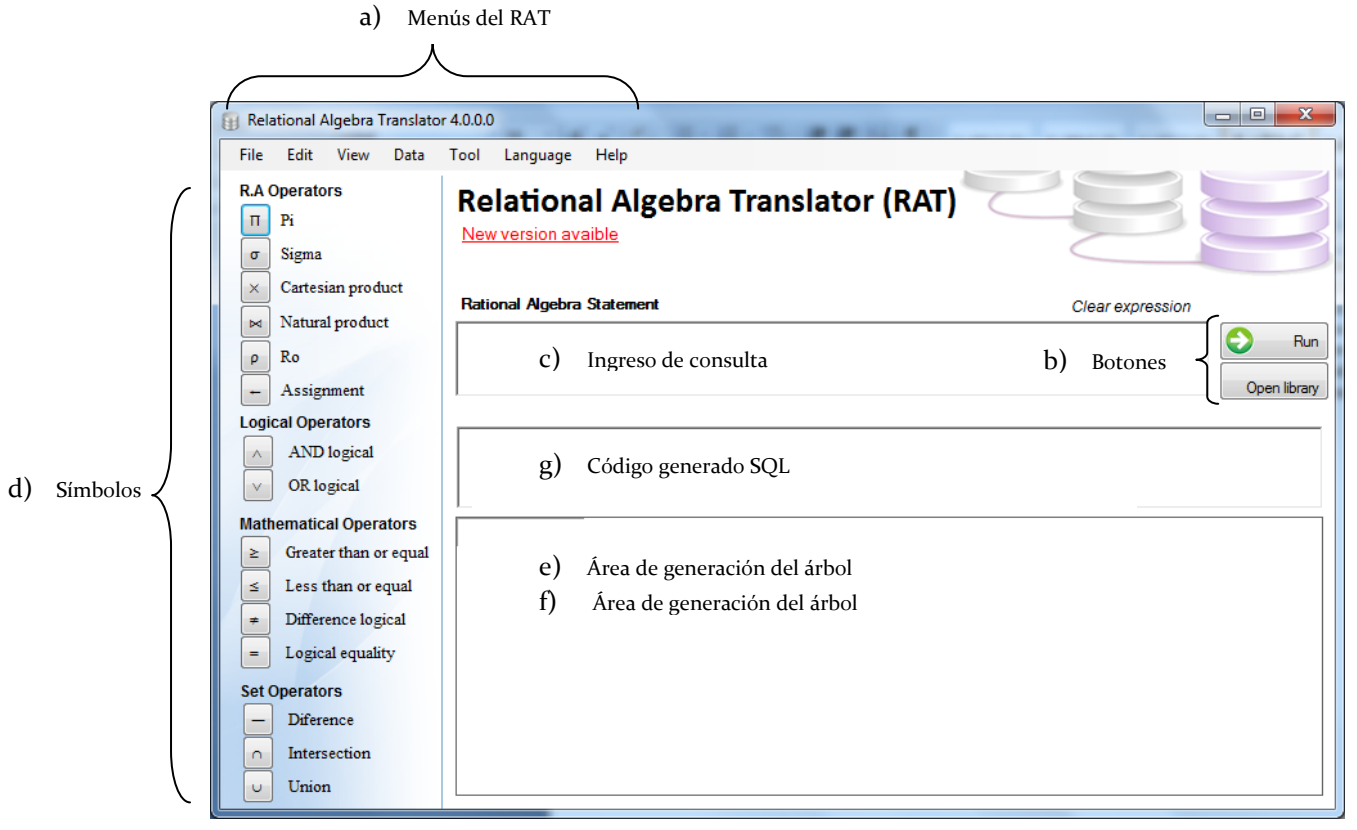
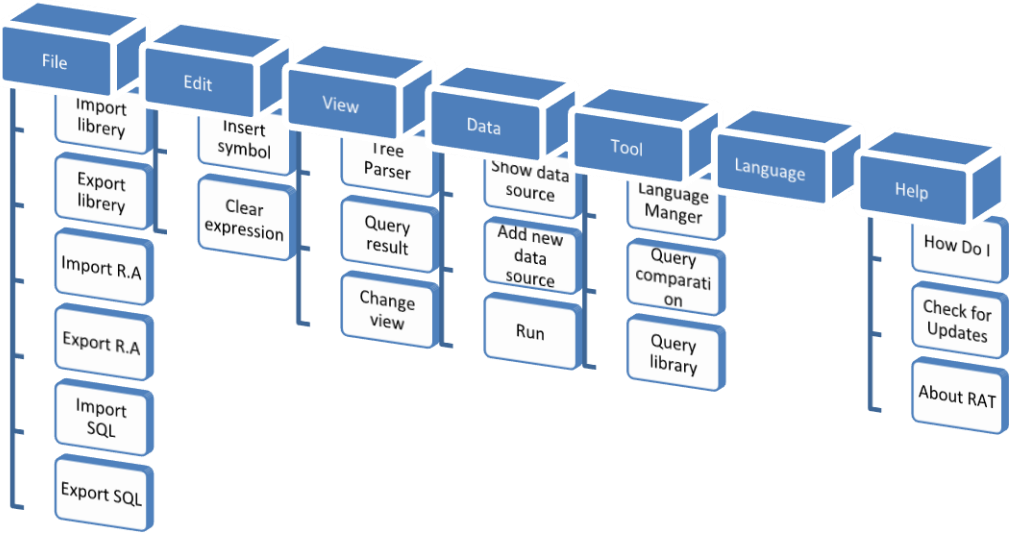


Imagen 6.7: Elementos de la ventana de traducción

Como se puede ver, la interfaz del RAT tiene muchas opciones para facilitar el procesamiento de las consultas. Los menús del RAT son convencionales, tiene las siguientes opciones:



4.3.2 Funcionalidad de exportación e importación

Seguidamente se detallan las funciones de cada menú y submenú, partiendo por la opción File.

a. Import libreray: El RAT permite guardar consultas que el usuario considere importante, estas consultas van a dar a una biblioteca común para el programa, si quiere puede importar bibliotecas de otros usuarios a la suya. **NOTA: El importar la biblioteca dará como resultado la eliminación de su vieja biblioteca.**

b. Export library: tiene la función de exportar la biblioteca común del sistema.

c. Import R.A: importa sentencias de álgebra relacional, en este caso sólo importa una sola sentencia.

d. Export R.A: exporta sólo una sentencia de álgebra relacional

e. Import SQL: importa sólo una sentencia de SQL.

f. Export SQL: exporta sólo una sentencia de SQL

4.4.3 Accesos rápidos

En el caso del menú Edit, su función principalmente es agrupar submenús cuyas funciones sean ayudarle a escribir sentencias de álgebra relacional, ella se divide en dos submenús, el primero de ellos es el de insertar un símbolo. El programa tiene dos formas de insertar símbolos, la primera de ellas es mediante el mouse con el punto d, en la imagen 6.7 o mediante el menú de insertar símbolo. En este mismo submenú esta la opción de limpiar por completo la pantalla con la función "clear expression". En la tabla 6.2 se encuentra un resumen de los accesos rápidos, para ser llamados debe presionar la tecla Control (Ctrl) izquierda más el código propio del acceso rápido.

Nombre de símbolo	Forma del símbolo	Acceso rápido
Assignment (asignación)	$variable ← relación	Ctrl + A
Pi	Π{columnas} (relación)	Ctrl + P
Sigma	σ{condición} (relación)	Ctrl + S
Ro	ρ{nueva} (relación)	Ctrl + R
Cartesian product (producto cartesiano)	Relacion1 × relación2	Ctrl + Q
Natural product (producto natural)	Relacion1 ⋈ Relacion2	Ctrl + N
Unión	Relacion1 ∪ Relacion2	Ctrl + U
Intersection (Intersección)	Relacion1 ∩ Relacion2	Ctrl + I
Diference (Diferencia)	Relacion1 – Relacion2	Ctrl +M
AND logical (Y lógico)	Condicion1 ∧ Condicion2	Ctrl + Y
Or logical (O lógico)	Condicion1 ∨ Condicion2	Ctrl + O

Tabla 6.2 Accesos rápidos

4.4.4 Vistas de la aplicación

El RAT tiene dos formas de visualizar una consulta de álgebra relacional, la primera de ellas es mediante un árbol grafico, separado en niveles. Para ello se va utilizar una consulta generara para este fin:

$$\Pi_{\{descripcion\}}(\sigma_{\{identificador = id\_consultor\}}(pf\_diccionario \times pf\_consultor))$$

Consulta 6.1 Ejemplo de consulta del RAT

Observe en la imagen 6.8 como se construye un árbol automáticamente con sólo ingresar la consulta en el programa interpretador. Sin embargo el RAT también disponen de otra vistas más tradicionales como la representación de tablas, en la imagen 6.9 se observa el resultado de la consulta contra una base de datos real.

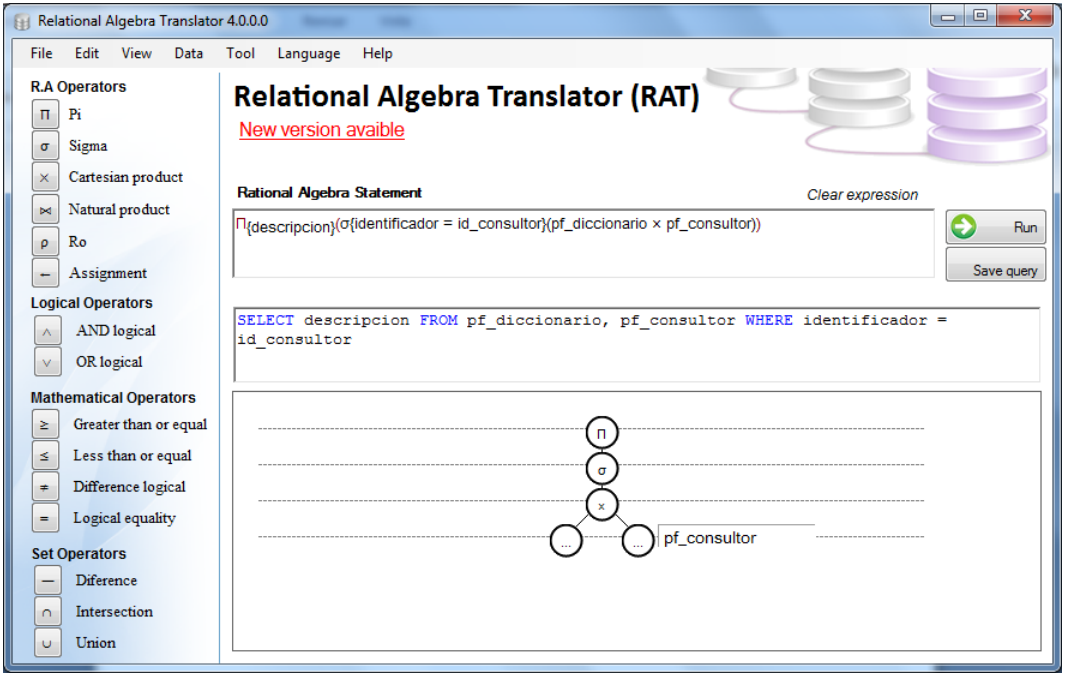


Imagen 6.8: Árbol de Parser la consulta 6.1

Para poder cambiar de vista en vista, el RAT provee los submenús de “Change view” que cambia de modo, por otro lado se puede ir explícitamente a una vista en particular mediantes las opciones de “Tree parser” o “query result”.

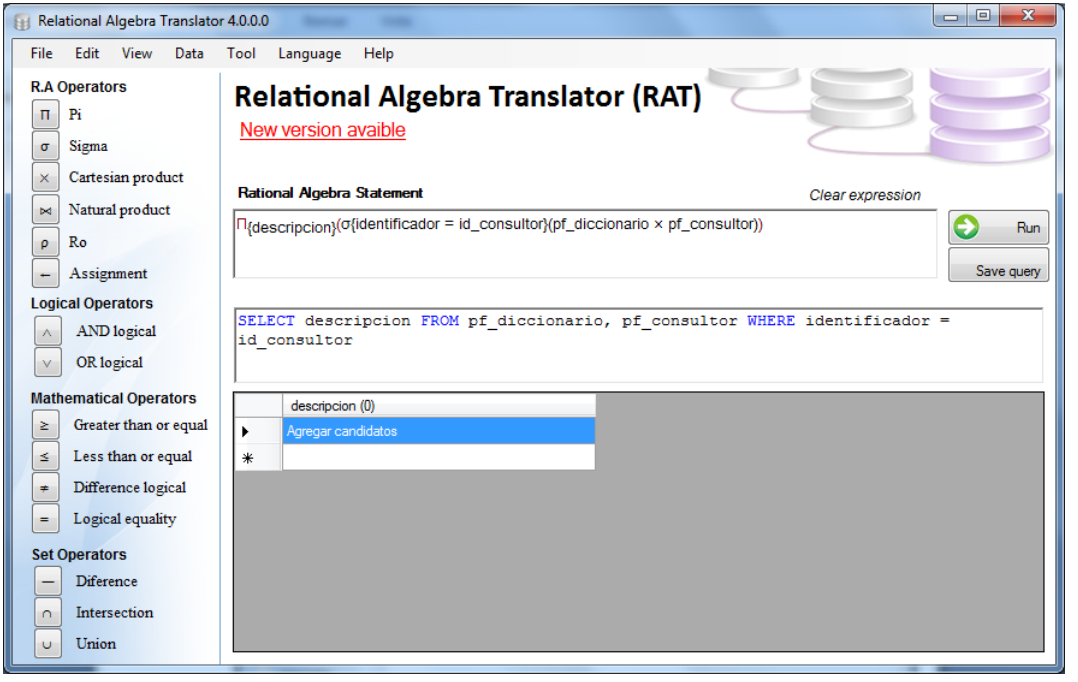


Imagen 6.9: Resultado de la consulta, sobre la base de datos real

4.4.5 Cambiar idioma

El RAT actualmente tiene soporte para cuatro idiomas, que son español, ingles, alemán e italiano, estos idiomas fueron escogidos basados en la cantidad de descargas del programa, desde que el sitio fue lanzado a la red. Para poder cambiar de idioma, basta con ir al menú de “Language” y seleccionar el nuevo idioma. El software cambiará toda la interfaz sin reiniciar el programa, este proceso pude demorar varios segundos.

4.4.6 Establecer conexiones con la bases de datos

Para permitir que la mayor cantidad de bases de datos sean compatibles con el RAT, se utilizan dos conectores, uno es genérico llamado ODBC y otro es desarrollado para Oracle, en general se puede conectar a cualquier base de datos si existe un conector ODBC.

4.4.6.1 Conexión ODBC

Las siglas de ODBC significan Open Data Base Conector, es un estándar establecido por Microsoft para permitir mediante una capa intermedia poder tener comunicación con diversas bases de datos, eso significa que para poder conectarse por ejemplo con MySQL, es necesario descargar el conector ODBC.

Lo primero que tenemos que hacer, es descargar el conector de la base de datos a la que queramos conectarnos. Después de descargarlo, procedemos a agregar nuestra nueva conexión, para ello es necesario abrir el administrado de origines de datos ODBC de Windows; presione en el menú DATA la opción de ADD NEW DATA SOURCE, con lo que nos aparece la ventana de la imagen 6.10.

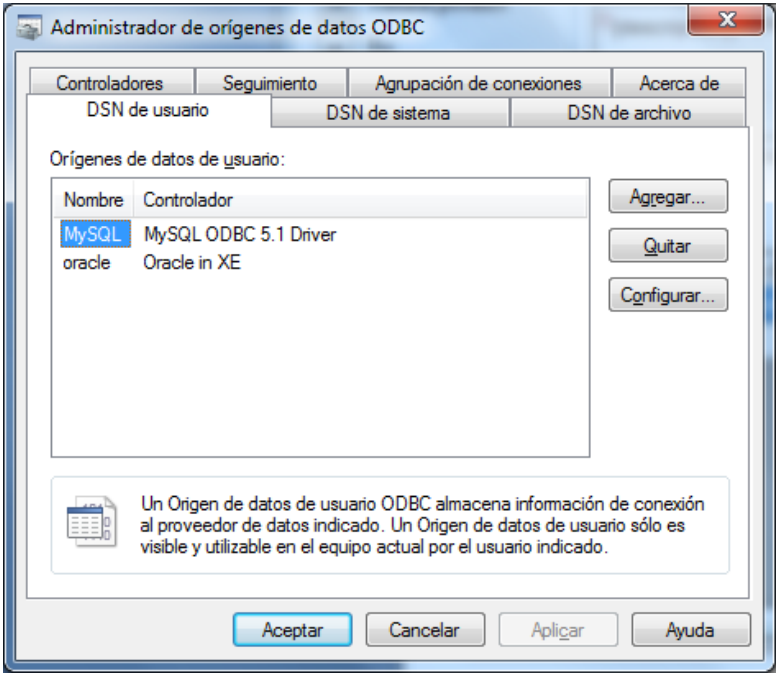


Imagen 6.10 Administrador de orígenes de datos ODBC

Una vez aquí, se debe agregar una nueva conexión, para ello de click en “Agregar” a la derecha de la ventana, lo que nos saldrá son las bases de datos que tienen conectores

instalados en la computadora, por ejemplo en la ventana 6.11 está instalado el conector para MySQL sin embargo es posible que cuando vean su ventana, no tengan el conector disponible, puede descargar el conector desde el mismo sitio donde descargó el RAT.

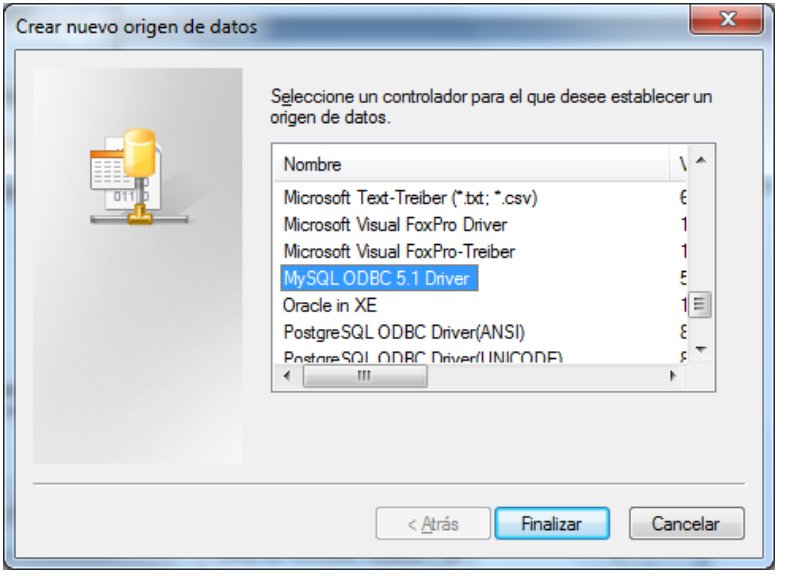


Imagen 6.11 Lista de orígenes instalados

Damos click a finalizar y según sea el conector nos solicitará determinada información, en general siempre es la misma información para todas las bases de datos, en la imagen 6.12 se completan los siguientes datos:

a. Data sourcce Name: este es el nombre con el que vamos identificar la conexión, puede ser cualquier nombre.

b. Server: se establece la dirección IP de donde se encuentra el servidor, si está en la misma computadora es localhost.

c. Database: es el nombre del schema de la base de datos, si usa otra base de datos, este campo es posible no aparezca.

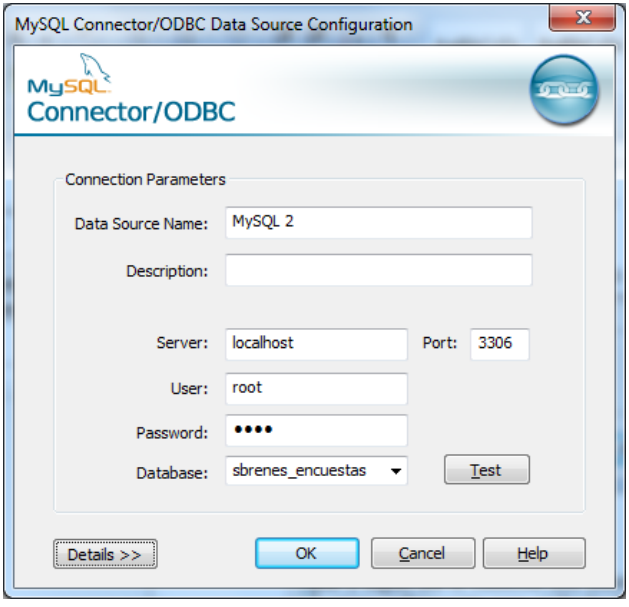


Imagen 6.12 Lista de orígenes instalados

Cuando le damos OK, nos llevara a la ventana del Administrador de orígenes de datos ODBC, donde nos debe aparecer una nueva conexión agregada con el nombre que se le había puesto en la ventana anterior.

4.4.6.2 Conexión final

Si bien es cierto, Oracle tiene su conector ODBC muchas veces resultar conveniente (en particular con Windows de 64bits) utilizar el conector propio Oracle. En la imagen 6.13 se completan los datos necesarios para realizar la conexión.

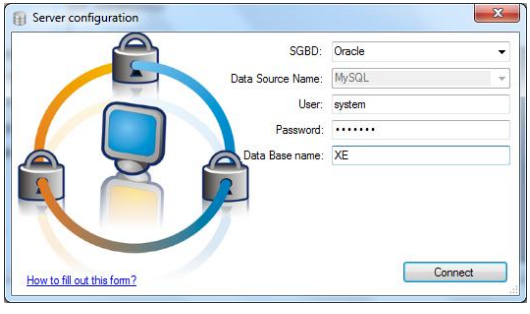


Imagen 6.13 Ventana de conexión del RAT

Detalles de conexión para Oracle

Detalle	Valor
SGBD	Oracle
Data Source Name	[vacio]
User	System [usuario con el que se va conectar]
Password	manager [clave del usuario]
DataBase name	XE [en el caso de la versión express]

Detalles de conexión para MySQL y PostGresSQL

Detalle	Valor
SGBD	ODBC
Data Source Name	MySQL [nombre que se ingreso en el administradore de ODBC]
User	[vacio]
Password	[vacio]
DataBase name	[vacio]

Cuando se confirme los datos, el sistema se conectará (no indicará nada) o en su defecto le indicará que hubo un fallo y fue imposible levantar la conexión. Si no hubo fallo en el proceso, ya puede escribir consultas y evaluarlas mediante la tecla F5, o con el botón "Run" a la derecha del cuadro de texto donde se ingresan las consultas. Los datos que usted ingreso serán guardados de forma automática, de manera que la próxima vez que ingrese a la ventana sólo es darle click al botón "Conectar".

4.4 Traducción de álgebra relacional al RAT

Procedemos ahora a traducir algunas sentencias de álgebra relacional al RAT con la intensión de ver las particularidades en la traducción.

Sentencia de álgebra relacional	Sentencia en el RAT
$\boldsymbol{\pi}_{\boldsymbol{esqema}}\,(relacion)$	$\boldsymbol{\pi}\{esquema\}(relacion)$
$\sigma_{\boldsymbol{esqema}}\,(relacion)$	$\sigma\{esquema\}(relacion)$
$A \cap B$	$[A] \cap [B]$
$A \cup B$	$[A] \cup [B]$
$A - B$	$[A] - [B]$

Todos los demás operadores, tiene una traducción directa; es decir no requiere agregarle patentéis para su funcionamiento.

4.5 Otras funciones del RAT

4.5.1 Biblioteca de consultas

El RAT da la posibilidad de guardar las consultas que se van generando, para ello es necesario dar click en el botón de "save query" ubicado debajo del botón "Run", una vez hecho esto va aparecer una segunda ventana donde le va solicitar completar la información de la consulta, que será el nombre y una descripción de la consulta. Cuando le dé OK, aparece la biblioteca con la nueva consulta. Una vez registrada la consulta es muy simple reutilizaras o eliminarlas, tan sólo debe dar click en la flecha verdad para usarla, o en la equis roja para eliminarla. Para exportar/importar revisé la documentación de importación y exportación de bibliotecas, ubicada en el tema de interfaz del RAT.

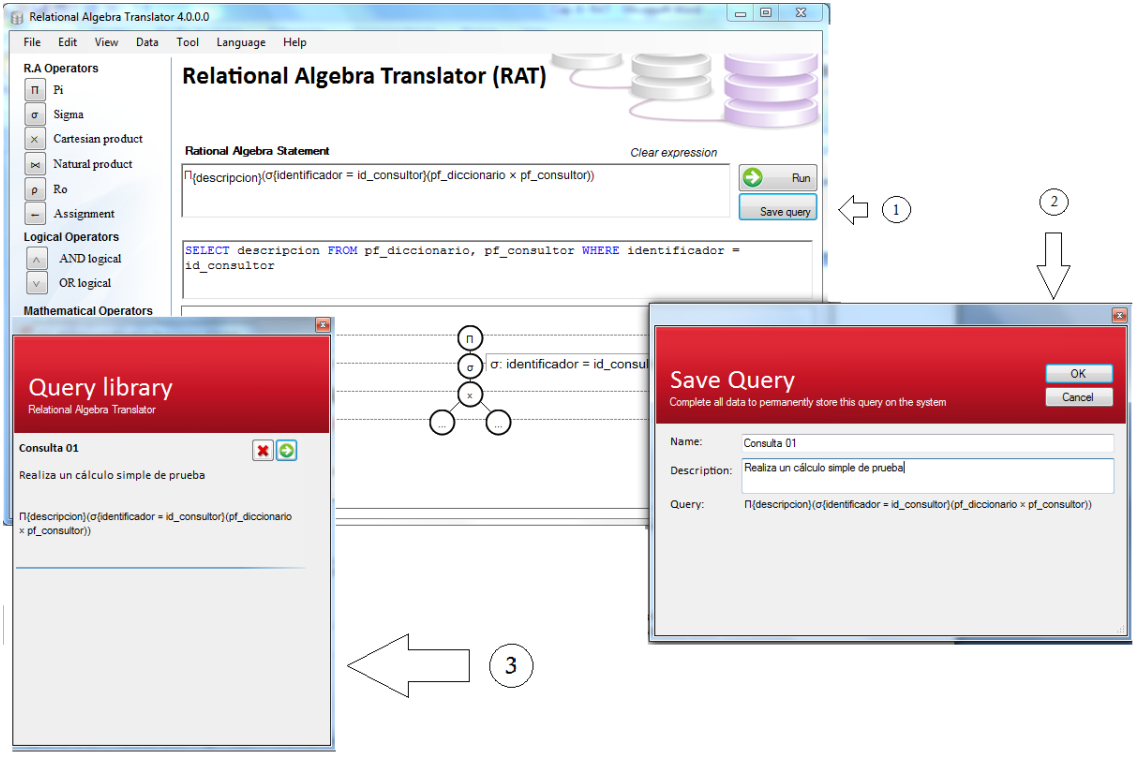


Imagen 6.14 Ciclo de vida para agregar consultas a la biblioteca

4.5.2 Comparador de consultas

Para determinar entre varias consultas cuál de ellas es la mejor, el RAT incorporar una métrica basada en la cantidad de los productos cartesianos de una consulta. En toda consulta, existen operadores unarios y binarios, si calculamos la cardinalidad (cantidad de elementos) de los conjuntos, y sumamos todos estos productos vamos a obtener un gran total con las poblaciones intermedias que representan el gasto de memoria de una consulta, a mayor valor es más ineficiente la consulta.

$$\sum_{i=0}^{A} |\, a_i \times b_i| = Sumatoria\ de\ todos\ los\ productos\ cartesianos$$

La otra métrica es el tiempo que dura en ser ejecutada la consulta, está representada por Δt, es decir el cambio del tiempo desde que se inicia hasta que termina de ejecutarse la consulta. Para realizar estas pruebas es suficiente abrir la librería y seleccionar las dos consultas a competir y dar click a “Compare”.

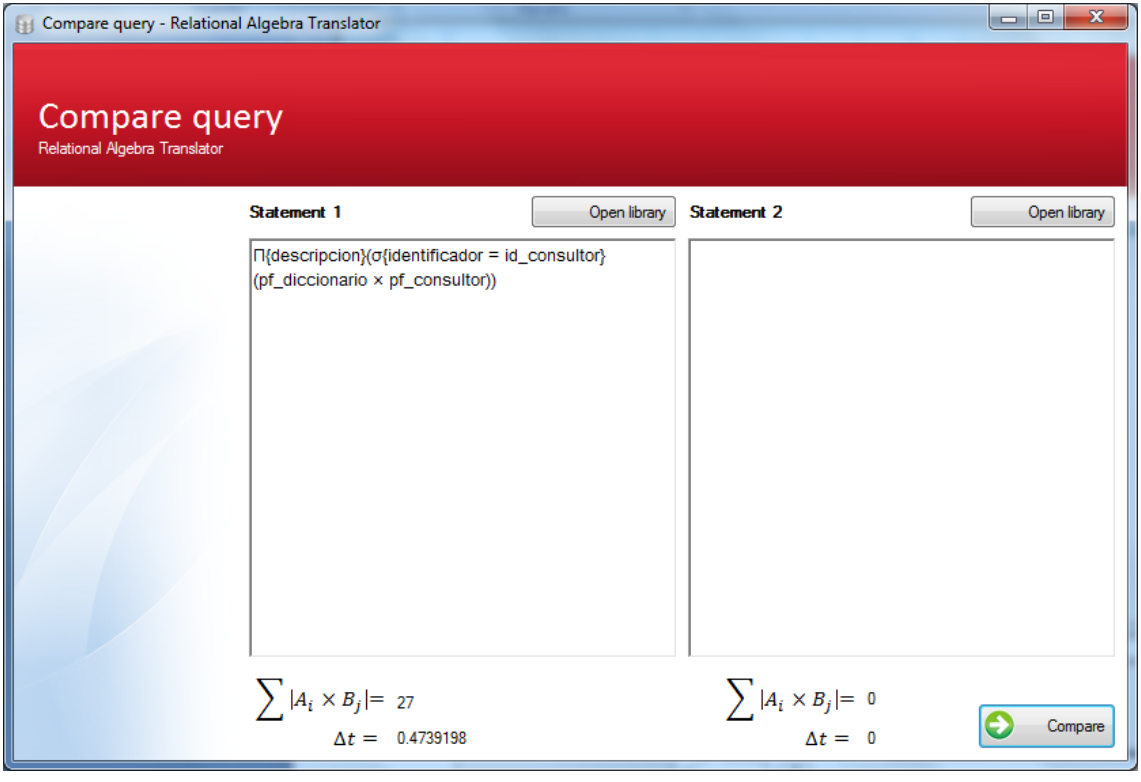


Imagen 6.15 Comparador de consultas del RAT

4.5.3 Administrador de idiomas

Aunque le RAT tenga sólo cuatro idiomas instalados, los usuarios pueden modificar un idioma existente para crear otro completamente nuevo, esta herramienta es sumamente fácil de usar, consiste en una serie de cajas de texto que se cambian por la nueva frase traducida.

Para instalar el nuevo idioma, es necesario dar click en "Add language", esto nos abrirá una ventana de guardar archivo, escogemos el nombre del idioma y la ruta. La ruta debe ser la raíz de la aplicación y dentro de ella existe una carpeta que dice "language" ahí es donde debemos copiar el archivo generado, la ruta por ejemplo puede quedar como `C:\Program Files\Relational Algebra Translator\lenguage`

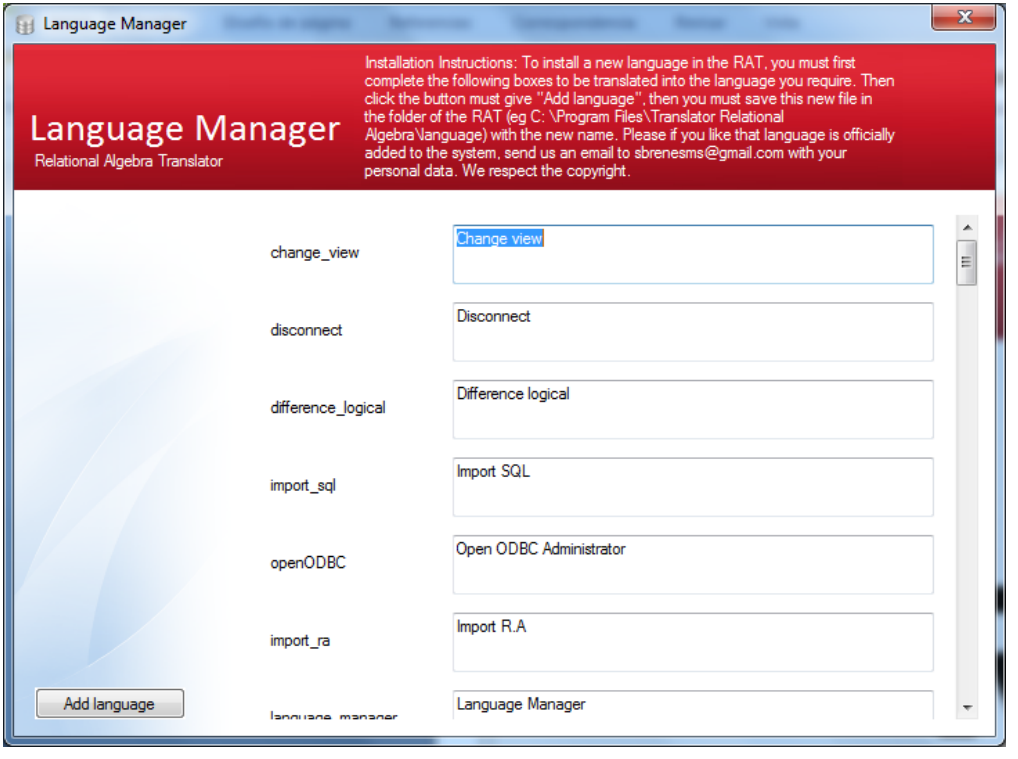


Imagen 6.16 Administrador de idiomas

BIBLIOGRAFÍA CONSULTADA

[1] Aho, Alfred V. *Compiladores, principios, técnicas y herramientas,* PEARSON EDUCACIÓN, Segunda Edición, México, 2008.

[2] Hopcroft, Jonh E. *Introduction to Automata Theory, Languajes, and Computation, Assison Wesley,* Segunda Edición, USA, 2001.