EXDACLET: HERRAMIENTA DE DATACLEANING BASADA EN AGENTES INTELIGENTES ORIENTADA A LA WEB

JAVIER ARMANDO LASSO ACHICANOY
JHON FREDDY QUINTAS RODRIGUEZ

UNIVERSIDAD DE NARIÑO
FACULTAD DE INGENIERIA
DEPARTAMENTO DE SISTEMAS
PASTO
2008

EXDACLET: HERRAMIENTA DE DATACLEANING BASADA EN AGENTES INTELIGENTES ORIENTADA A LA WEB

JAVIER ARMANDO LASSO ACHICANOY
JHON FREDDY QUINTAS RODRIGUEZ

Trabajo de Grado presentado como requisito parcial para optar el título de Ingenieros de Sistemas

Dr. RICARDO TIMARAN PEREIRA, Ph.D.
Director

UNIVERSIDAD DE NARIÑO
FACULTAD DE INGENIERIA
DEPARTAMENTO DE SISTEMAS
PASTO
2008

Nota de aceptación:

_____
_____
_____
_____
_____
_____

_____
Firma del presidente del jurado

_____
Firma del jurado

_____
Firma del jurado

San Juan de Pasto, 18 de Febrero de 2008.

"Las ideas y conclusiones aportadas en la tesis de grado son responsabilidad exclusiva de los autores"

Articulo 1. Acuerdo No. 234 del 11 de Octubre de 1966, emanado por el Honorable Consejo Directivo de la Universidad de Nariño.

Después de tanto tiempo de formación académica y personal en la Universidad, es muy importante reconocer el esfuerzo hecho por aprender, progresar y ser capaces de dibujar un futuro con fuertes bases en el presente para nuestras vidas. También reconocer el apoyo constante e incondicional de la Familia y los amigos. Por eso quiero expresar mis agradecimientos.

En primer lugar a mi Mamá y a mi Papá porque han sido mi mayor ejemplo de vida, por su firmeza, su apoyo incondicional ante cualquier circunstancia, porque son ellos quienes me han mostrado como es la vida y me han dado la libertad de escoger mis propios caminos, siempre guiándome y recordándome que cada día debo y puedo ser mejor, por mi bien y por el bien de todos aquellos que hacen parte de mi vida. A mi hermana, por ser mi principal ejemplo de superación y de perseverancia, porque gracias a ella he aprendido lo que significa enfrentar los miedos y luchar por lo que se quiere. Gracias a ella porque en todos estos años me ha hecho ver que todos los sueños son alcanzables con esmero, esfuerzo, dedicación y convicción de que uno es capaz.

A mi novia, por ser mi compañera, mi guía, mi calma, mi motivo para despertar con alegría y ganas de ser mejor, por ser la acompañante de mis sueños y de mi realidad y por su apoyo como colega y como persona.

A mi amigo y compañero de trabajo con quien aprendí valores como la tranquilidad, la serenidad y la calma. A nuestro asesor, el Doctor Ricardo Timaran Pereira, Ph.D, por mostrarnos que el conocimiento no tiene barreras y que siempre hay algo nuevo por hacer, por mejorar y por crear e investigar.

A mis amigos, del barrio y de la universidad por compartir cada día sus ocurrencias y su presencia, por mostrarme que la vida es alegre y que siempre hay un momento para hablar y compartir.

Javier Armando Lasso Achicanoy.

"Nunca consideres el estudio como una obligación, sino como una oportunidad para penetrar en el bello y maravilloso mundo del saber." Albert Einstein.

En el transcurso de mi vida he emprendido el largo camino de realizarme como persona, y en la búsqueda de mi realización he aprendido a descubrir el mundo guiado de la mano de mis padres, hermanos y amigos a quienes les debo no solo una palabra, no solo un abrazo sino un de todo corazón Gracias.

Gracias a Dios por su apoyo aunque invisible sé que incondicional, Gracias a mi madre y padre por confiar en mi, por los concejos que a diario me dan y porque todos los días me han guiado por el camino del bien; a mis hermanos por apoyarme en todo lo que han podido, Gracias a todas las personas que residen en mi corazón y que yo sé que resido en el de ellas, Gracias al trabajador amigo que encontré en quien comenzó siendo mi compañero de tesis, Gracias a todos y cada uno de los profesores que han ilustrado el camino del saber y en especial al Doctor Ricardo Timaran Ph.D, quien nos impulsó hacia la investigación y nos mostró que siempre se puede hacer algo más.

Gracias a todos y cada uno por que han influido notablemente en la forma de ver el mundo y porque me han enseñado que el mundo se recorre paso a paso y que caminar por el largo sendero de la vida nos hace cada vez más sabios.

Jhon Freddy Quintás

Vale más saber alguna cosa de todo, que saberlo todo de una sola cosa." Blaise Pascal.

CONTENIDO

LISTA DE FIGURAS

Pág.

LISTA DE TABLAS

Pág.

RESUMEN

En este trabajo se presenta el análisis, diseño e implementación de EXDACLET, una Herramienta de Datacleaning Basada en Agentes Inteligentes Orientada a la Web para ayudar en los procesos de Limpieza, Selección y Transformación de datos.

La arquitectura de EXDACLET consta de cinco módulos, el módulo de conexión encargado de mantener una comunicación constante entre la herramienta y el Sistema Gestor de Base de Datos (SGBD) el cual almacena los resultados de los procesos de Limpieza de Datos, el módulo de importación encargado de interpretar y cargar los archivos soportados por la herramienta, el módulo de Kernel dentro del cual se encuentran todos los filtros de limpieza, selección y transformación implementados, el módulo de exportación que permite almacenar los resultados en archivos y el módulo de interfaz gráfica que permite la interacción sencilla y amigable entre el usuario y la herramienta.

Los filtros de limpieza implementados son *Number Null Clean* que contiene varios filtros internos, *String Null Clean*, *Trim, Expert Rule Editor* y *Email Cleaner*. Los filtros de transformación son *Discretize, Normalize, Char Replace, Upper Case, Lower Case, Truncate String, Non-printable Character Search, Change Attribute Name, Change Attribute Type, Add Attribute, Delete Attribute, Attribute Clear, Binarize, Table Encoder* y *Table Decoder*. Los filtros de selección son *Max Length, Min Length, Jaro-Winkler Search, Metaphone Search, Double Metaphone Search, LD Search, Soundex Search, Duplicates Search* y *Table Union*.

En este trabajo se analiza y evalúa los resultados y el desempeño de *Duplicates Search, Soundex Search y LD Search* para filtros de selección, *Char Replace, Non-Printable Character Search, Binarize y Table Enconder/Decoder* para filtros de transformación y *Email Cleaner* para filtros de limpieza.

ABSTRACT

In this work is presented the analysis, design and implementation of EXDACLET, a Web Oriented Data Cleaning Tool Based on Intelligent Agents to assist in Data Cleaning, Data Selection and Data Transform procedures.

EXDACLET architecture is composed of five modules, the connection module keeps a constant communication between the tool and the Data Base Management System (DBMS) who stores the results of Data Cleaning procedures, the import module who interpret and upload the tool supported files, the Kernel module who has the Cleaning, Selection and Transform procedures implemented, the export module who allows to save the results in a file and the graphics interface module who allows the simple and friendly interaction between the user and the tool.

The cleaning filters implemented are *Number Null Clean* with other filters inside, *String Null Clean*, *Trim, Expert Rule Editor and Email Cleaner.* The transform filters are *Discretize, Normalize, Char Replace, Upper Case, Lower Case, Truncate String, Non-printable Character Search, Change Attribute Name, Change Attribute Type, Add Attribute, Delete Attribute, Attribute Clear, Binarize, Table Encoder* y *Table Decoder. And the selection filters are Max Length, Min Length, Jaro-Winkler Search, Metaphone Search, Double Metaphone Search, LD Search, Soundex Search, Duplicates Search* y *Table Union*.

In this work are analyzed and evaluated the results and performance of *Duplicates Search, Soundex Search y LD Search* for selection filters, *Char Replace, Non-Printable Character Search, Binarize and Table Encoder/Decoder* for transform filters, *and Email Cleaner* for cleaning filters.

1. INTRODUCCION

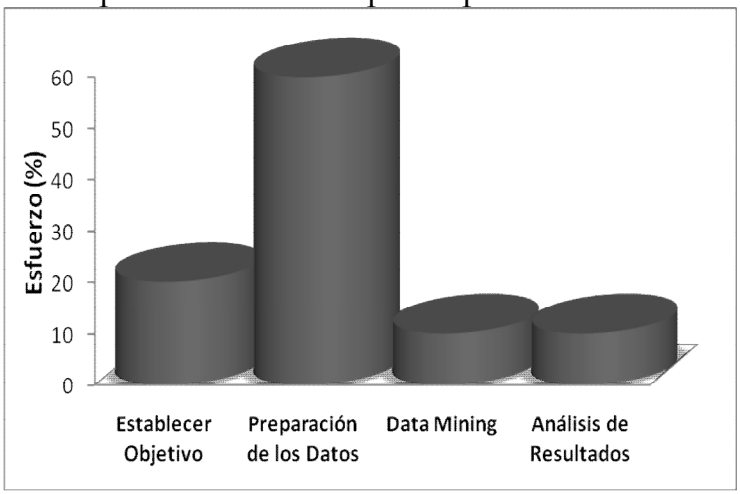
Los sistemas de soporte a la decisión, DSS (Decision Support Systems), son sistemas de información que combinan modelos y datos para intentar resolver problemas no estructurados utilizando una interfaz amigable para el usuario. Tales sistemas, como los de minería de datos o bodegas de datos, tienen que estar libres de errores y con los datos perfectamente limpios. A menudo, estos datos no son controlados cuidadosamente para calidad y no son comunes en diferentes fuentes de datos. Por lo tanto, la calidad de los datos es un compromiso obligado para problemas en el diseño y en el mal manejo de los datos. La falta de normalización y la pérdida de integridad en las restricciones, son algunos de los problemas de la calidad de datos basados en el diseño. Datos erróneos o perdidos (missing), duplicación de datos sin controlar y otros errores lógicos, son algunas formas del mal manejo de los datos. Como resultado, todas las vistas globales de los datos que se generen tendrán una alta probabilidad de error y al momento de ser interpretadas pueden mostrar información incoherente con la realidad llevando de esta manera a la toma de decisiones de forma inadecuada.

En este documento se presenta el trabajo de grado para optar por el título En Ingeniería de Sistemas, "EXDACLET: HERRAMIENTA DE DATACLEANING BASADA EN AGENTES INTELIGENTES ORIENTADA A LA WEB", desarrollado bajo software libre. Esta herramienta está compuesta por los módulos de carga de datos, limpieza, transformación y selección, en los cuales se implementaron diferentes algoritmos de limpieza, selección y transformación de datos. Esta herramienta es novedosa en su género, pues no existen otras para limpieza orientadas a la Web, y al ser de licencia GPL, permite a cualquier organización, especialmente las Pequeñas Y Medianas Empresas (PYMES), utilizar de manera libre esta herramienta, con el fin de mejorar la calidad de sus datos y por ende el soporte a la toma de decisiones.

1.1 PROBLEMA OBJETO DE ESTUDIO

En la actualidad, los procesos de Limpieza de Datos (Data Cleaning) son una tarea indispensable dentro del pre procesamiento de información para minería de datos, construcción de bodegas de datos (DataWareHouse) y en sí, para permitir que toda la información que estas contienen sea real, explicita y coherente, a demás, el proceso de Limpieza de Datos es la tarea más tediosa y que más recursos consume dentro del proceso de Descubrimiento de Conocimiento en Bases de Datos (DCBD) (ver Figura 1.1).

Figura 1.1 Esfuerzo requerido en cada etapa del proceso DCBD



Fuente: Técnicas de Análisis de Datos, Universidad Carlos III, Madrid [10].

Teniendo en cuenta esta situación y otras tales como la inexistencia de herramientas libres especializadas para la Limpieza de Datos, los pocos y básicos filtros implementados dentro de herramientas de Limpieza de Datos libres como Weka [1], Orange [2] o Alphaminer [3], el uso de técnicas estadísticas no basadas en agentes y el alto costo de las herramientas de software propietario como WinPure ListCleaner Pro [4], MatchIT Suite [5], ListWare 6.0 [6], MerlinMerge® SpeedPro [7], que además suelen aplicar el proceso a formatos muy estrictos, se ve la necesidad de desarrollar esta herramienta.

MatchIT Suite [5] y MerlinMerge® SpeedPro [7] exigen a parte del pago por realizar el proceso del Limpieza de Datos la entrega de la información a la cual se le quiere realizar este tratamiento, lo cual implica exponer la información que

puede ser confidencial y más aún, no se ofrecen todos los servicios en un solo paquete, sino por módulos, incrementando así el costo.

WinPure ListCleaner Pro [4] y ListWare 6.0 [6] permiten ser adquiridas, pero se limitan a una baja cantidad de registros a manipular.

ListWare 6.0 [6] diseñada para funcionar bajo una plataforma creada por el mismo desarrollador Melissa Data [8]

El desarrollo de una herramienta especializada bajo software libre para Limpieza de Datos, permitirá que muchas pequeñas y medianas empresas puedan aplicar procesos de Limpieza de Datos a sus bases de datos.

1.2 OBJETIVOS

1.2.1 Objetivo General Permitir el acceso a tecnologías de minería de datos a pequeñas y medianas empresas a través de una herramienta orientada a la Web que facilite los procesos de Limpieza de Datos (Data Cleaning).

1.2.2 Objetivos Específicos

- Analizar diferentes tipos de herramientas para Minería de Datos y Limpieza de Datos.
- Identificar todos los requerimientos de la herramienta EXDACLET.
- Utilizar el lenguaje de modelamiento unificado (UML) para realizar el diseño de la herramienta.
- Desarrollar la herramienta utilizando plataformas de desarrollo libres.
- Probar la herramienta utilizando repositorios reales y simulados de diferentes fuentes (Internet).

1.3 ORGANIZACIÓN DEL DOCUMENTO

Este documento está organizado en capítulos. En el capítulo dos se presenta toda la información concerniente al proceso KDD (Knowledge Discovery in Databases, Descubrimiento de Conocimiento en Bases de Datos), en el capitulo tres se hace referencia al proceso de Limpieza de Datos (Data Cleaning). En el capítulo cuatro se describen cada uno de los algoritmos implementados en EXDACLET para los procesos de precarga, limpieza, transformación y selección. En el capítulo cinco se describen las herramientas para Limpieza de Datos existentes en el mercado y las ventajas y desventajas de las mismas. En el capitulo seis se presentan las tecnologías utilizadas para el desarrollo de la herramienta. En el capítulo siete se desarrolla todo lo concerniente al análisis orientado a objetos (UML) que se realizó para construir la herramienta. En el capítulo ocho se presenta la implementación del proyecto, en el capítulo nueve se presentan las pruebas realizadas y los resultados, y por último en el capítulo diez se presentan las conclusiones de este trabajo.

2. EL PROCESO DE DESCUBRIMIENTO DE CONOCIMIENTOS EN BASE DE DATOS - DCBD

2.1 DEFINICION

El termino DCBD (Descubrimiento de Conocimiento en Bases de Datos), KDD por sus siglas en ingles (Knowledge Discovery in Databases), apareció por primera vez a finales de los años 80 para enfatizar que el conocimiento es el producto de un descubrimiento guiado por los datos, y ha servido como punto de unión para las diferentes áreas de investigación que se dedicaban a estudiar el proceso de análisis de datos y extracción de conocimiento a partir de los datos desde distintos puntos de vista, tales como el campo de las bases de datos, la estadística, las matemáticas, la lógica o la inteligencia artificial.

El proceso de *KDD* es el proceso que utiliza métodos de minería de datos (algoritmos) para extraer (identificar) patrones que tras ser evaluados e interpretados, de acuerdo a las especificaciones de medidas y umbrales, usando una base de datos con alguna selección, pre-procesamiento, muestreo y transformación, se obtiene lo que se piensa es conocimiento [26].

El proceso de *KDD* es interactivo, iterativo e involucra numerosos pasos con la intervención del usuario en la toma de muchas decisiones y se resumen en las siguientes etapas:

- Selección.
- Pre-procesamiento / Limpieza de Datos.
- Transformación / Reducción.
- Minería de Datos.
- Interpretación / evaluación.

2.2 ETAPAS DEL PROCESO DE KDD

En la figura 2.1 se muestra todas las etapas del proceso de KDD.

Figura 2.1 Etapas del proceso de KDD

2.2.1 Selección de los objetivos En esta etapa hay que estudiar el problema y decidir cuál es la meta del proyecto. Esta fase es similar a la de cualquier otro proyecto, y suele requerir la colaboración de un experto. Si se hace bien el planteamiento del problema, se descubren fácilmente las fuentes de datos y los algoritmos que se aplicarán. Un mal planteamiento del problema puede llevar a resultados erróneos.

2.2.2 Preparación de los datos Esta etapa del proceso de KDD es la que mayor esfuerzo requiere, se pueden distinguir dos sub-fases fundamentales.

- **Selección de los datos:** Se identifican las fuentes de datos internas o externas y se selecciona el subconjunto de datos necesarios para la aplicación de Minería de Datos, ya sean tablas de una base de datos o archivos de texto.
- **Pre-proceso de los datos:** Una vez identificados los datos a utilizar hay que estudiarlos para, por un lado entender el significado de los atributos y, por otro lado, para detectar errores de integración, como puede ser el hecho de que haya datos repetidos con distinto nombre o datos que significan lo mismo en diferente formato. Estos problemas surgen porque los datos pueden provenir de fuentes diferentes, y no todas almacenan la misma información de la misma manera. Con el pre-proceso de los datos lo que se consigue es tener un conjunto de datos adecuado para el correcto funcionamiento de las fases posteriores del proceso de KDD.

2.2.3 Limpieza y transformación de los datos En esta etapa se aplican todos los algoritmos de limpieza y transformación encargados de alcanzar y mantener la integridad de los datos de forma que estos resulten concisos y coherentes para posteriores procesos que se apliquen sobre ellos, ya sea de Minería de Datos o de otro propósito.

2.2.4 Minería de datos y análisis de resultados Son las etapas en las cuales se aplican los algoritmos de Minería de Datos y se generalizan los resultados o reglas generadas tras ese análisis.

Un algoritmo de Minería de Datos consiste en la aplicación de algoritmos de descubrimiento y de análisis de datos que bajo unas ciertas limitaciones de eficiencia computacional, produce una serie de patrones de los datos sobre los que actuó. Es importante resaltar que el espacio de posibles patrones es a menudo infinito, y que la enumeración de los mismos incluye alguna forma de búsqueda en ese espacio. Las restricciones computacionales en la práctica imponen límites severos en el sub-espacio de búsqueda a ser explorado por un cierto algoritmo de Minería de Datos.

3. EL PROCESO DE LIMPIEZA DE DATOS (DATA CLEANING)

3.1 DEFINICIÓN

La Limpieza de Datos (*Data Cleaning* o *Data Scrubbing*) es un término relacionado al proceso de resolver los problemas de los datos [25] y es el encargado de mejorar la calidad y la coherencia de los datos a través de procesos que modifiquen su forma o su contenido.

Estos problemas son denominados inconsistencias y pueden ser causadas por diferentes definiciones y formatos de datos en diferentes fuentes, por errores de digitación o por errores en la transmisión y/o almacenamiento de los datos [28].

3.2 HETEROGENEIDAD DE LOS DATOS

Se distinguen dos tipos de heterogeneidad en los datos: **Estructural** y **Semántica**. La heterogeneidad estructural ocurre cuando los atributos se definen de distinta forma en distintas bases de datos. La heterogeneidad semántica ocurre cuando atributos con estructuras similares toman diferentes semánticas y valores en diferentes bases de datos. Esta heterogeneidad o inconsistencia de los datos da como resultado que las organizaciones sean incapaces de manejar bien sus negocios. Este problema ha sido conocido como el **problema de cruces de datos**, cuyo objetivo se centra en identificar registros de la misma o de diferentes fuentes de datos que se refieran a la misma entidad del mundo real, incluso si los registros no se cruzan completamente.

Los problemas anteriormente mencionados se centran en la exactitud, puesto que se pueden presentar errores introducidos accidentalmente o intencionalmente a las bases de datos que comprometan dicho término al punto de hacer colapsar los sistemas de análisis de datos. Hay que tener en cuenta que una base de datos llena de datos erróneos o llena de inexactitudes puede implicar que algoritmos de minería de datos que sean aplicados sobre ellas nunca converjan en un punto para la generación de reglas en el soporte para toma de decisiones, puesto que sus resultados serían poco coherentes y poco confiables e inclusive podrían hacer que dichos algoritmos nunca terminen su procesamiento.

Teniendo en cuenta que el proceso de Limpieza de Datos es uno de los más costosos dentro de todo el proceso de Descubrimiento de Conocimiento en Bases de Datos (DCBD), la intervención del analista es indispensable durante esta etapa [9], esto significa que ningún proceso de Limpieza de Datos se puede desarrollar por si solo a pesar de que se apliquen un sin fin de metodologías como *algoritmos genéticos, lógica fuzzy, redes neuronales* u otras que impliquen un auto aprendizaje de máquina, debido a que todos los ámbitos en los cuales se desarrollan las bases de datos son generalmente excluyentes y no manejan un entorno común. Cada empresa diseña y desarrolla sus bases de datos de maneras diferentes y por lo tanto, su manipulación depende mucho de los usuarios. De esta manera, todos los procesos de Limpieza de Datos en los cuales intervenga el usuario deben y tienen que ser lo suficientemente intuitivos, sencillos de aplicar y no ser demasiado complejos, pues si lo son, conllevan a un alto costo de procesamiento [9].

Durante los últimos años se han desarrollado una gran variedad de algoritmos para Limpieza de Datos, que se encargan de buscar la forma más óptima de solucionar problemas sobre campos tanto numéricos como de texto. También existen otros algoritmos de transformación que tratan de tener en cuenta todas las posibles variantes de los errores que pueden presentar este tipo de campos con el fin de aplicar sobre ellos procesos de minería de datos [25].

Durante el proceso de diseño de todo tipo de base de datos se busca mantener los datos en su interior lo más íntegros, completos, y consistentes posibles, pero para aplicar técnicas de minería de datos es necesario realizar una buena limpieza y transformación de los datos en formatos que sean asequibles a la aplicación de estas técnicas.

En la mayoría de bases de datos existe mucha información que es incorrecta respecto al dominio de la realidad que se desea cubrir, y un número menor, pero a veces también relevante, de datos inconsistentes. Estos problemas se acentúan durante los procesos de integración de diferentes fuentes de datos. No obstante, mientras los datos erróneos crecen de manera lineal respecto al tamaño de los datos recopilados, los datos inconsistentes se multiplican [24].

Al existir una gran variedad de posibilidades sobre las cuales se puede iniciar una labor de limpieza y transformación de datos no implica que todas o algunas de ellas se deban aplicar de una manera secuencial, puesto que no todas las bases de

datos se componen de la misma manera y por lo tanto, tampoco contienen los mismos datos. Sumado a esto, entra en juego y en gran manera, que a pesar de que entre diferentes fuentes existan las mismas definiciones de cómo son los datos, también sea completamente diferente la forma mediante la cual estos son representados. Por esta razón, es muy importante conocer el dominio de donde provienen los datos.

3.3 INTEGRACIÓN DE DATOS

Muchas empresas en el momento de combinar o unir los datos de diferentes fuentes, siempre se suelen encontrar con tres tipos de problemas a saber:

- **Problemas de formato**: Cuando existen valores que se escriben de diferentes formas pero que representan a un mismo objeto o sujeto en la realidad, por ejemplo cuando en una fuente en el atributo Sexo aparece "F" y en otra fuente en el mismo atributo aparece "Femenino".
- **Problemas de definición**: Cuando atributos de diferentes fuentes se asocian directamente pero son identificados con diferentes nombres, por ejemplo, en una fuente aparece el nombre del atributo como "Genero", mientras que en otra aparece como "Sexo".
- **Problemas de tipo estructural**: Aparecen cuando dos atributos de distintas fuentes que serán relacionados, tienen diferentes estructuras de datos, por ejemplo, si en una fuente un atributo es de tipo numérico y este es asociado con otro atributo de otra fuente el cual es de tipo cadena.

Una vez esas fuentes de datos han sido combinadas se presenta el problema con la identificación de objetos, es decir, que datos sobre el mismo objeto se unifiquen y datos de diferentes objetos permanezcan separados. Este problema se conoce como el problema del esclarecimiento de la identidad [24] y puede presentar dos tipos de errores:

- **Dos o más objetos diferentes se unifican**: Los datos resultantes mezclarán patrones de diferentes individuos y serán un problema para extraer conocimiento. Esto será más grave cuanto más diferentes sean los dos objetos unificados.

- **Dos o más fuentes de objetos iguales se dejan separadas**: Los patrones del mismo individuo aparecerán repartidos entre varios individuos parciales. Este problema genera menos "ruido" que el anterior, aunque es especialmente problemático cuando se usan valores agregados (el total de compras será mucho menor si se considera un individuo real como dos individuos en la base de datos, por ejemplo).

Cuando se integran dos fuentes diferentes de datos de distintos objetos, suelen aparecer datos faltantes o datos inconsistentes, el primero hace referencia a que un dato es registrado en una fuente pero no en otra, mientras que el segundo hace referencia a que el dato es diferente entre las dos fuentes [27].

En este punto, aparecen campos redundantes total o parcialmente: "edad" y "fecha_de_nacimiento", "estado_civil" y "casado". En muchos casos, estos datos que se pueden considerar como inconsistentes, se convierten en faltantes; por ejemplo, si el mismo cliente tiene estados civiles diferentes en dos fuentes, es preferible declarar el valor como faltante que elegir al azar uno de los dos.

3.4 VALORES FALTANTES

La detección y posterior tratamiento de valores faltantes, perdidos o ausentes (*missing values*) es otra de las tareas fundamentales en la limpieza y transformación de datos. Estos valores, deben ser reemplazados por razones, tales como:

- Que los algoritmos de minería no puedan tratar bien este tipo de datos, o que sean ignorados y por lo tanto presentar un sesgo.
- Que los algoritmos de minería tengan algún método de sustitución de campos faltantes que no sea el correcto dado que no conocen el contexto al cual se asocia el atributo faltante.

La detección de campos faltantes puede resultar sencilla, basta con mirar una tabla de resumen de atributos o una gráfica estadística de los mismos y de esa manera poderlos identificar, pero a veces, esos campos faltantes no suelen ser representados como nulos. Por ejemplo, campos sin formato, es decir, campos en los cuales un valor nulo puede ser representado con un -1 para códigos postales o

"no tiene" para campos como teléfono o direcciones. Algunas veces, hasta son las mismas restricciones de integridad las que causan el problema [24] [27].

Pero más allá de que estos campos faltantes tengan que ser sustituidos por algún otro valor para los procesos de minería o de extracción de conocimiento, es importante saber el por qué esos valores son faltantes. Algunas veces esos valores expresan características relevantes, por ejemplo, la falta de teléfono puede significar que la persona no desea ser molestada o que simplemente no tiene una línea telefónica. También, muchos valores faltantes pueden existir en la realidad, como por ejemplo no tener un registro de accidentes medio de los últimos años cuando el cliente jamás ha tenido un accidente. Por estas razones, es muy necesario que el analista determine las acciones que se deben tomar sobre este tipo de valores en las bases de datos, entre ellas están omitir, reemplazar o eliminar.

3.5 VALORES ERRÓNEOS

A parte de los valores faltantes, también aparecen aquellos valores erróneos o anómalos, es decir que, representan información completamente incongruente con respecto a la realidad. Por ejemplo, casos en los que en un atributo llamado "Citología", para un registro en el cual la persona es de sexo masculino tiene como valor "si" o "no", en la realidad este valor debería ser tomado o registrado como "no aplica", puesto que decir "no" significaría que aún no se le ha realizado dicho procedimiento médico. Para este tipo de errores que se presentan en las bases de datos, se deben aplicar procesos de transformación de datos guiados por el analista, dado que muy difícilmente un sistema es capaz de interpretar dichos valores y asociarlos a la realidad, además que esto, puede causar serias alteraciones a procesos de extracción de conocimiento o minería de datos.

En general, existen muchas otras situaciones, algunas de ellas relativamente comunes en procesos de migración y fusión de bases de datos, otras que ya vienen dadas desde la misma construcción de las bases de datos y otras que son específicas del dominio y que por lo tanto necesitan soluciones muy específicas.

4. FILTROS Y ALGORITMOS IMPLEMENTADOS EN EXDACLET

El proceso de limpieza y transformación de datos tiene como pilar fundamental la aplicación de distintas técnicas o filtros para realizar un refinamiento de los datos corruptos, vacíos, ruidosos, inconsistentes, duplicados, alterados, homónimos, etc., con el fin de que el analista obtenga un conjunto de datos lo suficientemente claro y preciso para que se puedan realizar análisis o procesos de minería de datos.

EXDACLET contiene filtros que se aplican desde la importación de un archivo, verificando la integridad de los datos de este, por esto es posible importar archivos de diferentes formatos, estos son estandarizados y posteriormente son cargados en el Sistema Gestor de Base de Datos.

Además, EXDACLET tiene implementados filtros para limpieza de datos que utilizan procedimientos estadísticos, de búsqueda y de verificación de estructura en base a reglas; filtros de transformación que se encargan de tratar los diferentes valores en una tabla de datos, estandarizándolos a tamaños específicos o a formas de escritura estándar dependiendo de su estructura a través de algoritmos de estandarización y algoritmos de codificación; y filtros de selección que utilizan programación dinámica en base a algoritmos fonéticos y de similitud.

Los algoritmos fonéticos son algoritmos que indexan las palabras de acuerdo a su pronunciación y contexto pero no a su significado, mientras que los algoritmos de similitud se basan en la métrica de las palabras, es decir, hacen referencia a las distancias de edición (insertar, modificar, transponer o eliminar) por las cuales se logra que una palabra sea idéntica a otra.

4.1 ALGORITMOS DE PRECARGA

El proceso precarga de archivos en EXDACLET puede llevarse a cabo utilizando los siguientes algoritmos.

- **CSV Import.**

- **XLS Import.**
- **ARFF Import.**
- **SQL Import.**

4.1.1 CSV Import Es un módulo que permite la carga de archivos *CSV* (Comma-Separated Values), el cual es un formato de texto plano que almacena datos en forma de tabla, es un formato común en todas las plataformas gracias a su simplicidad. *CSV* es una implementación de un archivo de texto delimitado, que usa la coma como separador de valores o en otras ocasiones el punto y coma, las dobles comillas como delimitador de cadenas y los saltos de línea como delimitador de filas. En las ciencias de la computación este formato de archivo es denominado Texto Plano porque solo una tabla puede ser almacenada en un archivo y porque no sigue una estructura definida [13].

Durante el proceso de carga de archivos *.csv* se realiza un proceso (filtro) de verificación del tipo de atributos que contiene el archivo y los respectivos datos asociados a ellos, con el fin de garantizar la integridad de los datos.

Un ejemplo en el cual se ejecuta este filtro, es cuando se intenta cargar un archivo en el que un atributo está en formato fecha y es definido por el analista como numérico, ante esta situación el filtro cambia el tipo de dato al formato más estándar existente (varchar(255)). Un ejemplo de un archivo en formato *CSV* para representar la tabla de datos se muestra en la tabla 4.1.

Tabla 4.1 Tabla de datos de una archivo *CSV*

OBRA	AUTOR	EDITORIAL
Robots e Imperio	Isaac Asimov	Plaza y Janés
Invernáculo	Brian W. Aldiss	Minotauro
Absolute OpenBSD 2d Edition	Michael W. Lucas	No Starch Press
Mercaderes del Espacio	Frederik Pohl, M. Kornbluth	Minotauro

El archivo en formato *CSV* es:

OBRA, AUTOR, EDITORIAL
"Robots e Imperio", "Isaac Asimov", "Plaza y Janés"
"Invernáculo", "Brian W. Aldiss", "Minotauro"
"Absolute OpenBSD 2d Edition", "Michael W. Lucas", "No Starch Press"
"Mercaderes del Espacio", "Frederik Pohl, M. Kornbluth","Minotauro"

4.1.2 XLS Import Es un módulo que permite la carga de archivos XLS. XLS es un formato de archivo binario (BIFF5) propiedad Microsoft ™, diseñado para Microsoft Office Excel, este formato es uno de los más difundidos a nivel mundial gracias a su aplicabilidad. Este tipo de formato se carga en una hoja de cálculo para ser interpretado y manipulado.

Una hoja de cálculo es un programa que permite manipular datos numéricos y alfanuméricos en forma de tablas, aplicar fórmulas y funciones para realizar cálculos complejos y presentar los datos mediante gráficas.

4.1.3 ARFF Import (Atributte - Relation File Format) Es un módulo que permite la carga de archivos en formato ARFF.

El formato ARFF consiste en una archivo sencillo de texto que funciona a partir de etiquetas, similar a XML, y que debe cumplir con dos secciones: una cabecera y un conjunto de datos. La cabecera contiene las etiquetas del nombre de la relación (@relation) y los atributos (@attribute), que describen los tipos y el orden de los datos. La sección datos (@data) en un archivo ARFF contiene todos los datos del conjunto que se quiere evaluar separados por comas y en el mismo orden en el que aparecen en la sección de atributos [14].

Este formato está compuesto por una estructura claramente diferenciada en tres partes:

- **Cabecera.** Se define el nombre de la relación. Su formato es el siguiente:

@relation <nombre-de-la-relación>

Donde <nombre-de-la-relación> es de tipo String. Si dicho nombre contiene algún espacio será necesario expresarlo entre comillas.

- **Declaraciones de atributos.** En esta sección se declaran los atributos que compondrán el archivo junto a su tipo. La sintaxis es la siguiente:

@attribute <nombre-del-atributo> <tipo>

Donde <nombre-del-atributo> es de tipo String teniendo las mismas restricciones que el caso anterior. Weka acepta diversas definiciones de tipos, estas son:

a) NUMERIC: Expresa números reales.
b) INTEGER: Expresa números enteros.
c) DATE: Expresa fechas, para ello este tipo debe ir precedido de una etiqueta de formato entrecomillada. La etiqueta de formato está compuesta por caracteres separadores (guiones y/o espacios) y unidades de tiempo:

dd Día.
MM Mes.
yyyy Año.
HH Horas.
mm Minutos.
ss Segundos.

d) STRING: Expresa cadenas de texto, con las restricciones del tipo String comentadas anteriormente.
e) ENUMERADO: El identificador de este tipo consiste en expresar entre llaves y separados por comas los posibles valores (caracteres o cadenas de caracteres) que puede tomar el atributo. Por ejemplo, si se tiene un atributo que indica el tiempo podría definirse:

@attribute tiempo {soleado,lluvioso,nublado}

- **Sección de datos.** Se declaran los datos que componen la relación separando entre comas los atributos y con saltos de línea las relaciones.

@data
4,3.2

En el caso de que algún dato sea desconocido se expresará con un símbolo de cierre de interrogación ("?").

Es posible añadir comentarios con el símbolo "%", que indicará que desde ese símbolo hasta el final de la línea es todo un comentario. Los comentarios pueden situarse en cualquier lugar del archivo.

Un ejemplo de un archivo en formato ARFF es el siguiente:

@RELATION iris

@ATTRIBUTE sepallength NUMERIC
@ATTRIBUTE sepalwidth NUMERIC
@ATTRIBUTE petallength NUMERIC
@ATTRIBUTE petalwidth NUMERIC
@ATTRIBUTE class {Iris-setosa, Iris-versicolor, Iris-virginica}

@DATA
5.1, 3.5, 1.4, 0.2, Iris-setosa
4.9, 3.0, 1.4, 0.2, Iris-setosa
4.7, 3.2, 1.3, 0.2, Iris-versicolor
4.6, 3.1, 1.5, 0.2, Iris-setosa
5.0, 3.6, 1.4, 0.2, Iris-virginica

4.1.4 SQL Import El Lenguaje de Consulta Estructurado (Structured Query Language) es un lenguaje declarativo de acceso a bases de datos relacionales que

permite especificar diversos tipos de operaciones sobre las mismas. Una de sus características es el manejo del álgebra y el cálculo relacional permitiendo ejecutar consultas con el fin de recuperar información de interés de una base de datos, de una forma sencilla. Es un lenguaje de cuarta generación (4GL).

El SQL es un lenguaje que explota la flexibilidad y potencia de los sistemas relacionales permitiendo gran variedad de operaciones sobre los mismos. Es un lenguaje declarativo de alto nivel o de no procedimiento, que gracias a su fuerte base teórica y su orientación al manejo de conjuntos de registros, y no a registros individuales, permite una alta productividad en codificación. De esta forma una sola sentencia puede equivaler a uno o más programas que utilicen un lenguaje de bajo nivel orientado a registros.

Como se dijo anteriormente, y como suele ser común en los lenguajes de acceso a bases de datos de alto nivel, el SQL es un lenguaje declarativo, es decir, que especifica qué es lo que se quiere y no cómo conseguirlo, por lo que una sentencia no establece explícitamente un orden de ejecución. El orden de ejecución interno de una sentencia puede afectar gravemente a la eficiencia del SGBD (Sistema Gestor de Bases de Datos), por lo que se hace necesario que éste lleve a cabo una optimización antes de la ejecución de la misma. Muchas veces, el uso de índices acelera una instrucción de consulta, pero ralentiza la actualización de los datos, dependiendo del uso de la aplicación, se priorizará el acceso indexado o una rápida actualización de la información. La optimización difiere sensiblemente en cada motor de base de datos y depende de muchos factores.

Teniendo en cuenta que las sentencias SQL pueden contener acciones de creación, modificación y eliminación de bases de datos o de tablas y que EXDACLET es una herramienta WEB que utiliza un SGBD como pilar fundamental para su funcionamiento, todos los archivos en formato SQL que sean importados a la herramienta solamente estarán delimitados por sentencias de creación de tablas e inserción de registros de las mismas puesto que esto puede incurrir en graves problemas de seguridad, funcionamiento y rendimiento de la herramienta.

Un ejemplo de sentencia SQL inmersa en un archivo de extensión .SQL válido es el siguiente:

```
CREATE TABLE TABLA_NOMBRE (
 ci integer not null,
 nombre   VARCHAR (50),
 fecha_nac   DATE        NOT NULL,
 PRIMARY KEY (ci)
)
INSERT INTO TABLA_NOMBRE VALUES (1,'JACK','12-12-1957');
INSERT INTO TABLA_NOMBRE VALUES (2,'HANZ','22-08-1987');
INSERT INTO TABLA_NOMBRE VALUES (3,'MICHEL','11-03-1945');
INSERT INTO TABLA_NOMBRE VALUES (4,'LYNN','05-11-1989');
```

4.2 ALGORITMOS DE LIMPIEZA

4.2.1 Number Null Clean Contiene filtros específicos los cuales se encargan de llenar los datos nulos con valores proporcionados por un análisis estadístico de los datos de un determinado atributo, siendo estos valores información lo suficientemente relevante, teniendo en cuenta el tipo de filtro que se especifique y los datos contenidos por el atributo.

*** Average (Promedio)** La media aritmética o promedio, de una cantidad finita de números, es igual a la suma de todos ellos dividida entre el número de sumandos [15]. Expresada de forma más intuitiva, la media (aritmética) es la cantidad total de la variable, distribuida en partes iguales entre cada observación.

Este filtro se encarga de llenar todos los datos nulos de un determinado atributo con el valor promedio de todos los valores que contiene ese atributo. Por ejemplo, si en una habitación hay tres personas, la media de dinero que tienen en sus bolsillos sería el resultado de tomar todo el dinero de los tres y dividirlo en partes iguales entre cada uno de ellos. Es decir, la media es una forma de resumir la información de una distribución (dinero en el bolsillo) suponiendo que cada observación (persona) tendría la misma cantidad de la variable.

También la media aritmética puede ser denominada como centro de gravedad de una distribución, el cual no es necesariamente la mitad. Así, dados los números a1,a2, ... , an, la media aritmética será igual a:

$$\bar{x} = \frac{\sum_{i=1}^{n} a_i}{n} = \frac{a_1 + \cdots + a_n}{n}$$

Por ejemplo, la media aritmética de 8, 5 y -1 es igual a:

$$\bar{x} = \frac{8 + 5 + (-1)}{3} = 4$$

* **Max (Máximo)** El valor máximo hace referencia al valor de mayor magnitud en una determinada muestra poblacional.

Este filtro es útil para llenar todos los campos nulos con el mayor valor que se encuentre dentro de toda la muestra poblacional del atributo.

Por ejemplo, MAX {3, 1, 5, 7, 12, 8, 4} = 12

* **Min (Mínimo)** El valor mínimo hace referencia al valor de menor magnitud en una determinada muestra poblacional.

Este filtro es útil para llenar todos los campos nulos con el menor valor que se encuentre dentro de toda la muestra poblacional del atributo.

Por ejemplo, MIN {3, 1, 5, 7, 12, 8, 4} = 1

* **Variance (Varianza)** La varianza representa la media aritmética de las desviaciones de la media elevadas al cuadrado. Si se tiene en cuenta la colección completa de datos (la población en su totalidad) se obtiene la varianza poblacional; y si por el contrario, se presta atención sólo a una muestra de la población, se obtiene en su lugar la varianza muestral [15].

Es un filtro el cual se aplica para llenar los campos nulos con un promedio de la distancia que se encuentre entre los distintos datos presentes en una muestra, con el fin de que esos nuevos datos se encuentren acorde con todos los valores evaluados y no sean omitidos para un posible análisis.

Expresión de la varianza poblacional:

$$S_X^2 = \frac{\sum_{i=1}^{n} \left(X_i - \overline{X}\right)^2}{n} = \frac{\sum X_i^2}{n} - \overline{X}^2$$

*** Sum (Suma)** Hace referencia a la suma de todos los valores de una muestra poblacional.

Es un filtro útil, para llenar todos los datos nulos con el resultado de la suma de los diferentes valores existentes en el atributo.

*** Random_Mid (Media Aleatoria)** Es un valor aleatorio entre la media aritmética mas la desviación estándar y la media aritmética menos la desviación estándar.

Es un filtro muy útil debido a que proporciona valores aleatorios centralizados teniendo en cuenta el tamaño de la muestra, logrando así que todo dato nulo se transforme en un dato útil y relevante acorde al contexto del promedio de los datos.

Por ejemplo, RANDOM_MID {3, 1, 5, 7, 12, 8, 4}

PROM = 5,714285714
DESV_EST = 3,638419332

ENTONCES →
RANDOM_MID {3, 1, 5, 7, 12, 8, 4} = RANDOM ([PROM – DESV_EST, PROM + DESV_EST]) = 6.5; 6.9; 4.25; etc;

*** Random_Full (Aleatorio)** Es un valor aleatorio de toda la muestra poblacional, útil para mantener todos los valores dentro del rango del mayor y menor valor de toda la muestra.

Por ejemplo, RANDOM_FULL{3, 1, 5, 7, 12, 8, 4}

ENTONCES →
RANDOM_FULL {3, 1, 5, 7, 12, 8, 4} = RANDOM ({3, 1, 5, 7, 12, 8, 4}) = 1; 3; 12; 9; 6 etc.

*** Mode (Moda)** La moda es el valor que cuenta con una mayor frecuencia en una distribución de datos.

$$Md = x_i, \text{Si } n_i = \max\left\{f_j, j \in \{1, 2, ..., k\}\right\}$$

Se habla de una distribución bimodal de los datos, cuando se encuentran dos modas, es decir, dos datos que tengan la misma frecuencia absoluta máxima [15].

Es un filtro que se encarga de llenar los datos nulos con los valores de mayor frecuencia dentro de la muestra poblacional, siendo así, que estos nuevos valores ayudarán a mantener dicha muestra lo suficientemente centralizada.

Por ejemplo, en una muestra poblacional en la que se encuentran datos referentes a los automóviles que se compraron el último año se tienen los siguientes datos:

Renault = 120, Chevrolet = 100, Mazda = 60, BMW = 1, Datos Vacios = 12.

Entonces los 12 datos vacios serán llenados con Renault ya que este es el valor más frecuente y no con Mazda o BMW, manteniendo de esta forma los valores centralizados.

*** Zero (Cero)** Reemplaza todos los datos nulos por el valor cero (0). Es un filtro muy útil para muestras en las cuales se necesita la existencia de un valor que no sea el NULL.

*** Class_Rule (Marca De Clase)** La marca de clase es el valor medio entre el valor de menor magnitud y el valor de mayor magnitud [15].

Por ejemplo CLASS_RULE {3, 1, 5, 7, 12, 8, 4} = (1 + 12) / 2 = 6.5

Es un filtro muy útil debido a que proporciona un dato centralizado acorde a los datos existentes en el atributo y relativo al tamaño de los mismos. Este filtro se diferencia del promedio (average) en que el valor resultante no se obtiene teniendo en cuenta todos los datos sino el mayor y el menor valor existente.

4.2.2 Expert Rule Editor En este filtro el analista puede introducir reglas para modificar el contenido de un campo teniendo en cuenta el contenido de los campos que considere necesarios.

Para hacer uso de este filtro es necesario tener conocimientos básicos sobre la sentencia de control 'if' en su forma estándar y tener en cuenta ciertas variaciones que proporcionaran al analista un mayor control sobre los datos.

El analista encuentra en este filtro, un editor de texto sencillo, en el que puede escribir su sentencia; esta sentencia es analizada previamente para reconocer posibles errores de sintaxis.

La forma básica de esta sentencia es la siguiente:

```
if (Condición) {
Sentencia_1;
Sentencia_2;
}
```

En donde, ***Condición*** es una expresión lógica, que debe cumplirse a cabalidad.
Para crear una condición es posible utilizar los operadores que se muestra en la tabla 4.2.

Tabla 4.2 Operadores utilizados para crear condiciones

OPERADOR	DESCRIPCION
AND, &&	Operador lógico AND
BINARY	Convierte cadenas a cadenas binarias
/	Operador división
=	Operador igual
>=	Operador mayor o igual que
>	Operador mayor que
IS NULL	Compara con el valor NULL
IS	Compara un valor con un booleano
<=	Operador menor o igual que
<	Operador menor que
LIKE	Cadena paterna
NOT LIKE	Negación de LIKE
-	Operador menos
!=, <>	Operador diferente
REGEXP	Cadena paterna usando expresiones regulares
NOT REGEXP	Negación de REGEXP
NOT, !	Negación lógica
%	Operador módulo
||, OR	Operador lógico OR
+	Operador adición
*	Operador multiplicación
XOR	Operador lógico XOR

Para realizar comparación directa entre el contenido de un campo con algún valor, solo se escribe el valor, por ejemplo: campo = 3; pero si se quiere comparar el contenido de un campo con una cadena de texto o un caracter se debe utilizar como delimitador de la cadena la coma sencilla (‘), por ejemplo: campo = ‘nombre’; de no utilizar este delimitador de cadenas, se incurre en un error sintáctico.

En la condición son permitidos única y exclusivamente como signos de agrupación los paréntesis ( ), se puede poner uno o más grupos de ellos, pero teniendo en

cuenta que los grupos formados tengan una estructura lógica, el uso inadecuado de estos signos de agrupación puede conducir a un error sintáctico.

La ***Sentencia*** es una expresión, que contiene el campo que se va a modificar seguido de un valor literal o un valor proporcionado por una función.

La estructura de la sentencia está dada por:

Campo = Valor

Campo: Hace referencia al atributo que va a ser modificado.
Valor: Hace referencia a un valor ó al resultado de una función, en la tabla 4.3 se muestran las funciones matemáticas que se pueden utilizar en una sentencia y en la tabla 4.4 se muestran las funciones que se pueden utilizar sobre cadenas.

Tabla 4.3 Funciones matemáticas

FUNCION	DESCRIPCION
ACOS(X)	Retorna el arco coseno de X, esto es, el valor cuyo coseno es X. Retorna NULL si X no está en el rango -1 a 1
ASIN(X)	Retorna el arco seno de X, esto es, el valor cuyo seno es X. Retorna NULL si X no está en el rango de -1 a 1
ATAN(X)	Retorna el arco tangente de X, esto es, el valor cuya tangente es X.
ATAN2(Y,X)	Retorna el arco tangente de las variables X y Y. Es similar a calcular la arco tangente de Y / X, excepto que los signos de ambos argumentos se usan para determinar el cuadrante del resultado.
CEIL(X)	Retorna el entero más pequeño no menor a X.
CEILING(X)	Retorna el entero más pequeño no menor a X.
COS(X)	Retorna el coseno de X, donde X se da en radianes.
COT(X)	Retorna la cotangente de X.
DEGREES(X)	Retorna el argumento X, convertido de radianes a grados.
EXP(X)	Retorna el valor de e (la base del logaritmo natural) a la potencia de X.
FLOOR(X)	Retorna el valor entero más grande pero no mayor a X.
LN(X)	Retorna el logaritmo natural de X.

FUNCION	DESCRIPCION
LOG(X)	Si se llama con un parámetro, esta función retorna el logaritmo natural de X.
LOG10(X)	Retorna el logaritmo en base 10 de X.
LOG2(X)	Retorna el logaritmo en base 2 de X.
MOD(N,M)	Operación de módulo. Retorna el resto de N dividido por M.
PI()	Retorna el valor de π (PI). El número de decimales que se muestra por defecto es siete.
POW(X,Y)	Retorna el valor de X a la potencia de Y.
POWER(X,Y)	Retorna el valor de X a la potencia de Y.
RADIANS(X)	Retorna el argumento X, convertido de grados a radianes. (Tenga en cuenta que π radianes son 180 grados)
RAND(X)	Retorna un valor aleatorio en coma flotante del rango de 0 a 1. Si se especifica un argumento entero N, se usa como semilla, que produce una secuencia repetible.
ROUND(X,Y)	Retorna el argumento X, redondeado al entero más cercano. Con dos argumentos, retorna X redondeado a Y decimales. Y puede ser negativo para redondear Y dígitos a la izquierda del punto decimal del valor X.
SIGN(X)	Retorna el signo del argumento como -1, 0, o 1, en función de si X es negativo, cero o positivo.
SIN(X)	Retorna el seno de X, donde X se da en radianes.
SQRT(X)	Retorna la raíz cuadrada de un número no negativo X.
TAN(X)	Retorna la tangente de X, donde X se da en radianes.
TRUNCATE(X,Y)	Retorna el número X, truncado a Y decimales. Si Y es 0, el resultado no tiene punto decimal o parte fraccional. Y puede ser negativo para truncar (hacer cero) Y dígitos a la izquierda del punto decimal del valor X.

Tabla 4.4 Funciones sobre cadenas

FUNCION	DESCRIPCION
CONCAT(STR1,STR2,..,STRn)	Devuelve la cadena resultante de concatenar los argumentos. Devuelve NULL si alguno de los argumentos es NULL. Puede haber más de 2 argumentos.
CONCAT_WS(SEP,STR1,..,STRn)	Forma especial de CONCAT(). El primer argumento es el separador para el resto de los argumentos, se añade entre las cadenas a concatenar: Puede ser una cadena. Si el separador es NULL, el resultado es NULL.

FUNCION	DESCRIPCION
LCASE(STR)	Devuelve la cadena STR con todos los caracteres cambiados a minúsculas.
LEFT(STR,X)	Devuelve los X caracteres de la izquierda de STR.
LTRIM(STR)	Devuelve la cadena STR con los caracteres de espacios iniciales eliminados.
REPEAT(STR,X);	Devuelve una cadena que consiste en la cadena str repetida count veces. Si count <= 0, devuelve una cadena vacía.
REPLACE(STR,OLD,NEW);	Devuelve la cadena STR con todas las apariciones de la cadena OLD sustituidas por la cadena NEW.
REVERSE(STR)	Devuelve la cadena STR con el orden de los caracteres invertido.
RIGHT(STR,X)	Devuelve los X caracteres de la derecha de STR.
RTRIM(STR)	Devuelve la cadena str con los caracteres de espacios finales eliminados
SUBSTRING(STR,X,Y)	Devuelve la cadena STR entre los caracteres X hasta el caracter Y, el caracter Y es opcional
TRIM(STR)	Devuelve la cadena STR con los caracteres de espacios tanto iniciales como finales eliminados
UCASE(STR)	Devuelve la cadena STR con todos los caracteres convertidos a mayúsculas.

Una de las principales actividades que se pueden realizar dentro de la herramienta es la denominada **attribute clustering**, que permite disminuir la dimensión de una tabla combinando dos o más atributos.

Es una técnica que se aplica solamente cuando se tiene un conocimiento a priori de los datos y de su estructura. Además, se debe garantizar que los atributos a agrupar se encuentren libres de errores, por lo tanto, esto implica que previo a la realización del clustering o agrupamiento, se debe realizar una limpieza de los datos que contienen los atributos a agrupar. Por ejemplo, si se tiene el conocimiento a priori de que el campo **"sexo"** contiene los valores **"masculino"** y **"femenino"**, y el campo **"Estado Civil"** contiene los valores "**soltero", "casado"** y

"**divorciado"**. Teniendo en cuenta de que en los dos atributos se encuentran única y exclusivamente dichos valores, entonces estos registros de la base de datos pueden ser fácilmente agrupados en 6 grupos así:

1. Todos los registros de los masculino – soltero.
2. Todos los registros de los masculino – casado.
3. Todos los registros de los masculino – divorciado.
4. Todos los registros de los femenino – soltero.
5. Todos los registros de los femenino – casado.
6. Todos los registros de los femenino – divorciado.

Si se sabe que los valores correspondientes a esos campos son correctos, entonces no se presentará ningún problema de que aparezcan relaciones erróneas. De esta forma, se reducirá la cantidad de atributos de la base de datos logrando así la eliminación de redundancias y que los procesos de minería posteriores a la etapa de Limpieza de Datos resulten mucho más eficientes [24].

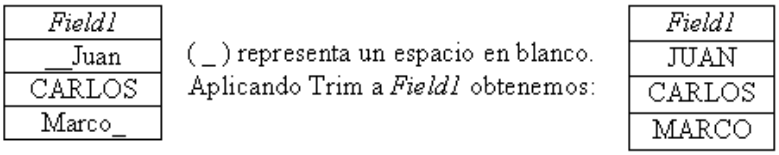
Los pasos a seguir para aplicar **attribute clustering** de acuerdo con el ejemplo anteriormente citado, son los siguientes:

- Crear un nuevo atributo mediante el filtro **ADD ATTRIBUTE** del tipo que resulte conveniente (group double)**.**
- Estandarizar los datos del campo **sexo** y el campo **estado civil** a través del filtro **EXPERT RULE EDITOR** de la siguiente manera:
    - Si sexo = masculino => sexo = m;
    - Si sexo = femenino => sexo = f;
    - Se estado_civil = '' => estado civil = soltero.
- Una vez estandarizados los datos utilizando nuevamente el filtro **EXPERT RULE EDITOR** se procede a especificar las nuevas reglas mediante las cuales serán generados los nuevos valores para el campo creado en el primer paso.
    - Si sexo = m y estado_civil = soltero => group = 1
    - Si sexo = m y estado_civil = casado => group = 2
    - Si sexo = m y estado_civil = divorciado => group = 3
    - Si sexo = f y estado_civil = soltero => group = 4
    - Si sexo = f y estado_civil = casado => group = 5
    - Si sexo = f y estado_civil = divorciado => group = 6.

- De esta manera, se garantiza que el nuevo atributo cuente con valores simples para que sobre él puedan ser aplicados procesos de minería de datos.
- Si el caso lo amerita, se eliminan los atributos que fueron tomados como base para este proceso de transformación.

4.2.3 Trim Este filtro de transformación, permite eliminar los espacios en blanco existentes en una cadena de texto, tanto en el inicio como en el final de la misma. Un ejemplo de este filtro se puede detallar en la figura 4.1.

Figura 4.1 Ejemplo filtro Trim



Este filtro es muy útil para la estandarización de datos, además optimiza otros algoritmos, obteniendo mejores resultados al momento de ser aplicados.

4.2.4 String Null Clean Es un filtro para limpieza de atributos de tipo cadena, en donde todos los valores vacios que se encuentren en un determinado atributo son reemplazados por el valor más común que se encuentre dentro del mismo atributo en general o teniendo en cuenta ciertas especificaciones que deban cumplir los otros atributos que componen la tabla de datos. Tales especificaciones son definidas por el usuario teniendo en cuenta que es él quien tiene conocimiento explicito de los datos que componen la tabla.

Este es un filtro muy útil a la hora de mantener la tabla de datos lo más consistente posible, basándose en los valores más comunes que esta pueda presentar.

4.2.5 Email Cleaner Es un filtro específico para cadenas de caracteres, en las cuales se busca que estas coincidan con la estructura de un correo electrónico (e-mail) básico; el filtro tiene la capacidad de avalar una dirección de correo electrónico o definitivamente descartarla como una dirección valida.

La fortaleza del filtro Email Cleaner radica en que es capaz de rescatar mediante una sugerencia, direcciones electrónicas erróneas siempre y cuando estas mantengan un nivel de error en su sintaxis tolerable; ese nivel es definido por la estructura de la dirección y por los caracteres que se utilizan en la misma. Un ejemplo de funcionamiento de este filtro se muestra en la tabla 4.5.

Tabla 4.5 Ejemplo filtro Email Cleaner

EMAIL	SUGERENCIA
.juan_perez#hotmail.com.	juan_perez@hotmail.com
*juan_perez/h.otmail.com	

4.3 EL PROCESO DE TRANSFORMACION

4.3.1 Discretize Es un filtro que permite mostrar ciertos valores continuos de forma discreta, agrupando valores y asignándole un nombre. Ejemplo.

Valor → {3, 1, 2}
Discretizando en 2 intervalos obtenemos: [1 - 1.5] y (1.5 – 3]
Discretizando valor → {(1.5 - 3] , [1-1.5] , (1.5-3]}

4.3.2 Normalize Este filtro transforma una variable aleatoria que tiene alguna distribución en una nueva variable aleatoria con distribución normal o aproximadamente normal, estos valores pueden ser representados en la campana de Gauss y sirven además para comparar diferentes valores que en la práctica no pueden ser comparados. Un ejemplo del proceso de normalización a una columna con valores numéricos se muestra en la tabla 4.6.

Tabla 4.6 Ejemplo filtro normalize

Valor inicial	Valor normalizado
1	-0.866
7	0.866
8	1.1547
1	-0.866
2	-0.5774

Valor inicial	Valor normalizado
2	-0.5774
2	-0.5774
11	2.0207
2	-0.5774

4.3.3 CharReplace Es un filtro específico para cadenas de caracteres, el cual se encarga de transformar un sólo caracter o una cadena completa por un caracter o cadena especificado por el analista, esto con el fin de convertir los datos en un conjunto estandarizado el cual ayudará a mejorar la eficiencia y la calidad de resultados que se obtengan al aplicar otra serie de filtros como la búsqueda de duplicados (Duplicates Search), la búsqueda de homónimos (LD Search), etc.

Además permite eliminar o transformar las incompatibilidades de los diferentes sistemas de escritura existentes en el entorno informático. Acoplando todos los datos a un sistema de caracteres que el analista considere de su interés.

4.3.4 Upper Case Este filtro de transformación, permite a una cadena de texto independientemente a que su contenido esté escrito en letras mayúsculas o minúsculas cambiarlo a letras mayúsculas de acuerdo al mapa de caracteres. Es un filtro muy útil a la hora de estandarizar cadenas de texto. Un ejemplo se muestra en la figura 4.2.

Figura 4.2 Ejemplo filtro Upper Case

Field1
Juan
CARLOS
marco

Aplicando Upper Case a *Field1* obtenemos:

Field1
JUAN
CARLOS
MARCO

4.3.5 Lower Case Este filtro de transformación, permite a una cadena de texto independientemente de que su contenido este escrito en letras mayúsculas o minúsculas, cambiarlo a letras minúsculas de acuerdo al mapa de caracteres. Es un filtro muy útil a la hora de estandarizar cadenas de texto Un ejemplo se muestra en la figura 4.3.

Figura 4.3 Ejemplo filtro Lower Case

Field1
Juan
CARLOS
marco

Aplicando Lower Case a *Field1* obtenemos:

Field1
juan
carlos
marco

4.3.6 String Truncate Este filtro de transformación, permite cortar una cadena; por izquierda, se debe especificar la cantidad de caracteres iniciales a eliminar y por derecha se debe especificar la cantidad de caracteres iniciales que no son eliminados de la cadena. Un ejemplo de este filtro se muestra en las figuras 4.4 y 4.5

Figura 4.4 Ejemplo filtro String Truncate (Truncate = 2)

Field1
Juan
Carlos
Marco

Aplicando Truncate = 2,
Por izquierda, obtendremos:

Field1
an
rlos
rco

Figura 4.5 Ejemplo filtro String Truncate (Truncate = 4)

Field1
Juan
Carlos
Marco

Aplicando Truncate = 4,
Por derecha, obtendremos:

Field1
Juan
Carl
Marc

4.3.7 Non-Printable Character Search Es un filtro que permite eliminar o reemplazar cualquier tipo de caracteres extraños por un espacio en blanco o por un caracter en especial. Todos estos caracteres son buscados dentro de la base de datos actual y si se encuentra más de uno de dichos caracteres estos serán listados con el fin de ser reemplazados uno a uno o en bloque de acuerdo a las necesidades o parámetros que considere adecuados el analista.

4.3.8 Change Attribute Type Este es un filtro capaz de transformar el tipo de un atributo, es decir, se encarga de convertir valores por ejemplo, de tipo cadena a tipo numérico siempre y cuando los datos contenidos en ese atributo cumplan con el nuevo formato que se les está especificando. Si no cumplen el nuevo formato pasarán a un nuevo filtro que necesitará de la interacción del analista para verificar

todos aquellos datos que no cumplen con el tipo que se haya especificado, este filtro consiste en la edición y/o eliminación de dichos datos.

4.3.9 Change Attribute Name Este es un filtro el cual permite el cambio del nombre de un atributo, todo esto con el fin de lograr que los nombres que identifican a cada atributo de la tabla de datos no resulte ser un cadena extraña compuesta por una serie de caracteres que pueden causar confusión a la hora de ser identificados. Junto con lo anterior resulta ser un filtro útil debido a que facilita la interpretación e identificación de los valores que componen a ese atributo.

4.3.10 Attribute Clear Este es un filtro el cual se encarga de limpiar todos los valores presentes en un atributo, puede ser de utilidad en situaciones en las que hay demasiados valores vacios dentro del atributo y aquellos valores que contiene datos están muy dispersos.

4.3.11 Add Attribute Es un filtro utilizado con el fin de definir nuevos atributos con un tipo de datos determinado, estos atributos pueden ser tratados mediante otros filtros según lo considere necesario el analista.

4.3.12 Delete Attributes Elimina uno o varios atributos dependiendo de su consistencia.

Es una técnica que se utiliza para permitir que los procesos de Limpieza de Datos y Minería de Datos, sean óptimos y trabajen sobre datos consistentes. Además de que no consuman muchos recursos de operación sobre conjuntos de campos pertenecientes a un atributo que resulta ser irrelevante para los procesos anteriormente mencionados. Por ejemplo, no resulta conveniente realizar un proceso ni de limpieza, ni de transformación, ni de minería, sobre un atributo en el que más del 50% de sus campos presenten datos nulos o completamente diferentes, puesto que esto incurre en un gasto excesivo de procesos que seguramente no resultarán lo suficientemente explícitos para que sobre ellos sea aplicado un buen proceso de Minería de Datos y por lo tanto el conocimiento que ellos generen sea muy escaso.

4.3.13 Binarize El objetivo fundamental de este filtro es reducir el tamaño de la tabla de datos que el usuario está utilizando y sobre la cual desea aplicar procesos de minería de datos, todo esto se logra mediante la identificación de los distintos valores que componen dicha tabla y la posterior generación de atributos relacionados a cada uno de esos valores en donde se establece la aparición o no de dichos valores en las diferentes transacciones que componen a la tabla de datos, de esta manera se logra reducir atributos que contienen valores con cadenas de texto o valores muy grandes en su interior (haciendo referencia a tamaño en bytes) a valores como lo son el uno (1) y el cero (0) que hacen referencia a la aparición de un determinado valor en un determinado atributo y que no incurren en un gasto de espacio.

4.3.14 Table Encoder – Decoder Este filtro de transformación permite asignar un código a cada uno de los diferentes valores que se encuentren dentro de los atributos seleccionados por el usuario; cuando se codifica un grupo de atributos, es necesario crear un diccionario de datos (DD) donde se registra el valor original de los atributos. Este diccionario de datos tiene la siguiente estructura: Code-Attribute-Value.

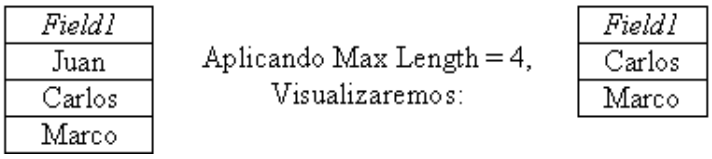
En el campo Code se almacena el código asignado a un Valor dentro de un atributo, en el campo Attribute se almacena el nombre del atributo en el cual es encuentra el código anterior y en el campo Value se encuentra el valor original a codificar.

El complemento a este filtro es el decodificador, este filtro actúa en base a un diccionario de datos previamente generado mediante el codificador de este filtro; su misión es retornar los valores codificados a los valores originales.

4.4. FILTROS DE SELECCIÓN

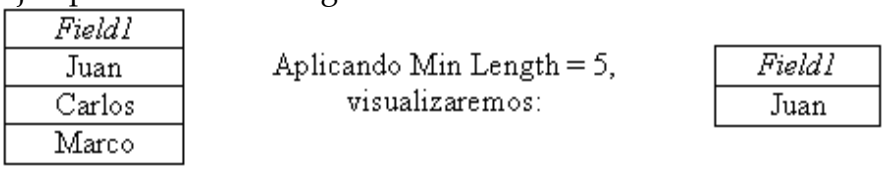
4.4.1 Max Length Este filtro permite visualizar los registros de un campo que superan un tamaño máximo específico, con el fin de editarlos o borrarlos. Un ejemplo se muestra en la figura 4.6.

Figura 4.6 Ejemplo filtro Max Length



4.4.2 Min Length Este filtro permite visualizar los registros de un campo que no superan un tamaño mínimo específico, con el fin de editarlos o borrarlos. Un ejemplo se muestra en la figura 4.7.

Figura 4.7 Ejemplo filtro Min Length



Los filtros Max Length y Min Length son útiles, ya que en algunos casos se confía en que los datos tienen un tamaño de cadena adecuado, con estos filtros es posible saber qué registros están por fuera del tamaño especificado.

4.4.3 LD Search (Levenshtein Distance - Search) La distancia de Levenshtein (LD) es la medida de similitud entre dos cadenas, la cual se refiere a una cadena de origen (s) y una cadena destino (t). La distancia entre las dos cadenas es el número de operaciones necesarias para transformar una cadena en otra, en donde una operación es la inserción, eliminación o sustitución de un caracter [16]. Por ejemplo:

Si s es "ardila" y t es "ardila", entonces LD(s,t) = 0, porque no se necesitan transformaciones, las cadenas ya son iguales.

Si s es "ardila" y t es "ardiya", entonces LD(s,t) = 1, porque una substitución (cambiar "l por y") es suficiente para transformar s en t.

La distancia de Levenshtein aumenta, cuanto más distintas sean las cadenas a comparar.

Este es un filtro de similitud que resulta útil cuando no se tiene en cuenta el sonido de las palabras, es decir, se pueden realizar transposiciones entre el primer y el ultimo caracter de la primera cadena para que coincida con la segunda y su distancia de levenshtein estará dada por 1, omitiendo completamente el posible sonido que la palabra pueda tener.

La ejecución de la distancia de Levenshtein involucra el uso de una matriz (n + 1) x (m + 1), donde n y m son las longitudes de las dos cadenas. El pseudocódigo del algoritmo se muestra en la figura 4.8.

Figura 4.8 Pseudocódigo algoritmo de la Distancia de Levenshtein

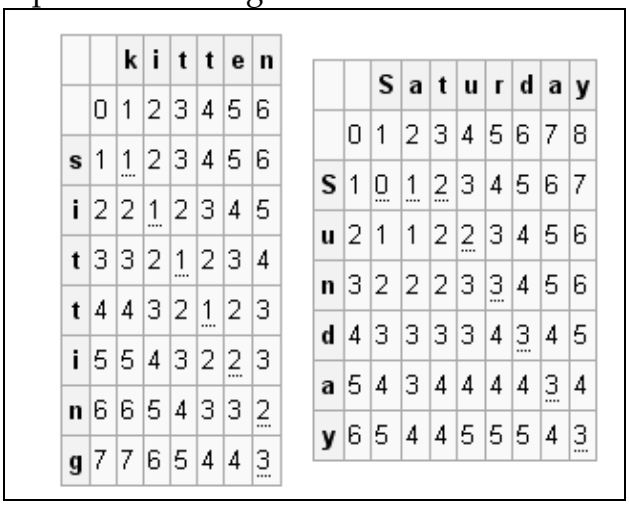
```
int LevenshteinDistance(char s[1..m], char t[1..n])
  // d is a table with m+1 rows and n+1 columns
  declare int d[0..m, 0..n]
  for i from 0 to m
     d[i, 0] := i
  for j from 0 to n
     d[0, j] := j
  for i from 1 to m
     for j from 1 to n
        if s[i] = t[j] then cost := 0
                       else cost := 1
        d[i, j] := minimum(
                           d[i-1, j] + 1,    // deletion
                           d[i, j-1] + 1,    // insertion
                           d[i-1, j-1] + cost   // substitution
                          )
  return d[m, n]
```

Gráficamente, los resultados generados por este algoritmo se pueden representar de la siguiente manera:

La distancia levenshtein entre kitten y sitting es 3
La distancia levenshtein entre Saturday y Sunday es 3

De acuerdo a la figura 4.9 los valores subrayados son la cantidad de operaciones a ejecutarse.

Figura 4.9 Tabla de representación algoritmo Levenshtein



		k	i	t	t	e	n
	0	1	2	3	4	5	6
s	1	1	2	3	4	5	6
i	2	2	1	2	3	4	5
t	3	3	2	1	2	3	4
t	4	4	3	2	1	2	3
i	5	5	4	3	2	2	3
n	6	6	5	4	3	3	2
g	7	7	6	5	4	4	3

		S	a	t	u	r	d	a	y
	0	1	2	3	4	5	6	7	8
S	1	0	1	2	3	4	5	6	7
u	2	1	1	2	2	3	4	5	6
n	3	2	2	2	3	3	4	5	6
d	4	3	3	3	3	4	3	4	5
a	5	4	3	4	4	4	4	3	4
y	6	5	4	4	5	5	5	4	3

Para determinar la similitud entre las dos cadenas es necesario establecer un mínimo valor del resultado de la aplicación del algoritmo, teniendo en cuenta pruebas realizadas con datos reales se determinó que este valor debe ser menor o igual a 2, es decir, menor o igual a 2 operaciones ya sean de eliminación, modificación o transposición.

4.4.4 Double Metaphone Search El algoritmo de búsqueda double metaphone es un algoritmo fonético escrito por Lawrence Philips, el cual retorna dos códigos para una cadena. Esto ayuda a algunos casos de ambigüedad así como para las variantes de múltiples apellidos con un linaje común. Por ejemplo, al codificar el nombre "Smith", devuelve un código primario SM0 y un secundario XMT, mientras que "Schmidt" devuelve un código primario XMT y un secundario SMT. Los dos tienen en común el código XMT [17].

Double Metaphone intenta responder a inconsistencias en palabras en Ingles, Eslavo, Germánico, Céltico, Griego, Francés, Italiano, Español y otros. Por esto utiliza un conjunto de reglas mucho más complejas (ejemplo: aproximadamente 100 diferentes contextos para utilizar la letra C.), convirtiéndolo en un algoritmo experto.

4.4.5 Jaro-Winkler Search Es un filtro basado en el esquema propuesto por Monge & Elkan [18][19] de edición de distancias entre cadenas para verificar su similitud, este esquema mapea un par de cadenas S y T a un número real R, en el que entre menor sea el valor de R, mayor será la similitud entre las cadenas S y T. el valor de R hace referencia al costo de la mejor secuencia de operaciones de edición que convierten a S en T, estas operaciones son típicamente la inserción, eliminación y sustitución de caracteres. A cada una de estas operaciones se les asigna un costo el cual sumado resultará en el valor mapeado entre las dos cadenas.

La distancia Jaro (dj) basada en dos cadenas s1 y s2 es:

$$d_j = \frac{m}{3a} + \frac{m}{3b} + \frac{m-t}{3m}$$

Donde:

m es el número de caracteres que coinciden en las dos cadenas.
a y b es la longitud de las cadenas s1 y s2.
t es el número de transposiciones que se requieren para convertir s1 en s2.

Dos caracteres de las cadenas s1 y s2 se consideran coincidentes solamente si no se encuentran más lejanos que el máximo valor de la longitud entre las dos cadenas dividido entre 2 y restado 1.

$$\frac{\max(a,b)}{2} - 1.$$

El número de transposiciones viene dado por el número de caracteres coincidentes diferentes dividido entre dos [20][21][22][23].

4.4.6 Soundex Search Es un algoritmo fonético para la indexación de nombres por su sonido cuando son pronunciados en Ingles.

El objetivo básico es que nombres con la misma pronunciación sean codificados a una cadena de tal forma que sean cruzados a pesar de sus diferencias ortográficas.

El código soundex para un nombre consiste de una letra seguida de 3 dígitos. La letra es la primera letra del nombre y los números son la codificación de las consonantes restantes. Las consonantes que tienen un sonido similar comparten el mismo número, por ejemplo las consonantes labiales B, F, P y V son codificadas como 1. Las vocales pueden afectar la codificación pero nunca son codificadas a menos de que aparezcan al inicio del nombre. El algoritmo funciona de la siguiente manera:

- Retener la primera letra de la cadena.
- Remover todas las ocurrencias de las siguientes letras, menos si están al inicio de la cadena: a, e, i, o, u, h, y, w.
- Asignar los números a las letras sobrantes, después de la primera así:
  - b, f, p, v = 1.
  - c, g, j, k, q, s, x, z = 2.
  - d, t = 3.
  - l = 4.
  - m, n = 5.
  - r = 6.
- Si dos o más letras con el mismo número se encuentran adyacentes en el nombre original (antes del paso 1), o adyacente exceptuando cualquier intervención de la h y la w, entonces omita todas menos la primera.
- Devuelva los primeros 4 caracteres de la cadena, si la cadena es menor de 4 entonces rellénela con ceros (0) hasta que complete 4.

Soundex Search es uno de los filtros más eficientes en lo que se refiere a la búsqueda de posibles duplicados pero presenta una limitación dada porque el código generado tiene como primer caracter la primera letra de la cadena y no siempre las cadenas que pueden resultar similares tienen la misma primera letra.

4.4.7 Duplicates Search Este es un filtro de similitud en el cual son analizados todos los atributos que el analista considere convenientes y en los cuales la distancia de edición entre 2 cadenas es igual a cero (0).

Es uno de los algoritmos o filtros más prácticos y eficientes cuando lo que se busca en la base de datos es reducir el número de registros existentes teniendo en cuenta que algunos de ellos se encuentran duplicados bajo ciertos atributos, hay que tener en cuenta que el buscar duplicados en base a un atributo es completamente diferente a buscarlos en base a dos o más de ellos, un ejemplo en el que se evidencia esta situación y el cual es muy común dentro del manejo de bases de datos de usuarios en entornos comerciales o en entornos gubernamentales es la búsqueda en base a atributos como ***documento de identidad, primer apellido, segundo apellido, primer nombre y segundo nombre***.

Los duplicados que genere una búsqueda por ***documento de identidad*** no necesariamente son los mismos que si se buscara por ***primer apellido, primer nombre***, y la misma situación se presentará para cualquier combinación que se realice sobre estos atributos.

5. ANTECEDENTES

En el mercado existen algunas herramientas diseñadas para la limpieza y transformación de datos. A continuación se menciona en detalle algunas de las herramientas más conocidas y utilizadas por su aplicabilidad y rendimiento.

5.1 WEKA

Es una extensa colección de algoritmos de máquinas de conocimiento desarrollados en la universidad de Waikato (Nueva Zelanda), esta herramienta implementada en el lenguaje de desarrollo JAVA, se ha convertido en una herramienta muy útil para el descubrimiento de conocimiento en bases de datos, dada su aplicabilidad se ha hecho necesario la implementación de algoritmos para limpieza y transformación de bases de datos, para obtener más y mejores resultados a la hora de aplicar los distintos algoritmos de Minería de Datos [1].

La licencia de Weka es GPL (GNU Public License), lo que significa que este programa es de libre distribución y difusión. Además, ya que Weka está desarrollado en Java, es independiente de la arquitectura y funciona en cualquier plataforma sobre la que haya una máquina virtual Java disponible.

Weka básicamente manipula archivos de formato ARFF, pero esto no implica que sea el único formato que acepte, puesto que también trabaja con archivos de formato CSV, C45 y Binary Serialized Instances.

Los algoritmos más utilizados para limpieza y transformación de datos por WEKA son los siguientes:

Filtros Aplicados A Atributos

Add: Permite adicionar un nuevo atributo. Como parámetros se le debe proporcionar la posición que va a ocupar este nuevo atributo (comenzando desde

el 1), el nombre del atributo y los posibles valores de ese atributo separados por comas. Si no se especifican, se sobreentiende que el atributo es numérico.

AddExpression: Permite agregar un nuevo atributo cuyo contenido es el resultado de una función que puede involucrar o no el contenido de los demás atributos. En el momento de especificar la fórmula para el nuevo atributo se puede hacer referencia a los atributos utilizando el caracter "a" seguido del número del atributo (comenzando por 1), un ejemplo de fórmula utilizado en este filtro puede ser:

(a3^3.4)*a1 + sqrt (floor(tan(a4)))

Los operadores y funciones que soporta son +, −, *, /, ^, log, abs, cos, exp, sqrt, floor (función techo), ceil (función suelo), rint (redondeo a entero), tan, sin, (,). Otro argumento para este filtro es el nombre del nuevo atributo.

AddNoise: Este filtro permite añadir ruido a un determinado atributo que debe ser nominal. Como argumento es necesario especificar el porcentaje de ruido deseado, la semilla para generarlo y si se quiere que al introducir el ruido cuente o no con los atributos que faltan. Un ejemplo del accionar de este filtro se muestra en la tabla 5.1.

Tabla 5.1 Filtro AddNoise de Weka

Sexo	Noise(50%) Sexo
M	F
F	M
M	M
F	F

ClusterMembership: Este filtro actúa dado un conjunto de atributos y un atributo que define la clase de los mismos, devuelve la probabilidad de cada uno de los atributos de estar clasificados en una clase u otra. Tiene por parámetro ignoredAttributeIndices que es el rango de atributos que se desea excluir para aplicar este filtro. Dicho intervalo se puede expresar por cada uno de los índices, los atributos separados por comas o definiendo rangos con el símbolo guión ("-").

Es posible denotar al primer y último atributo con los identificadores first y last (en este caso la numeración de los atributos comienza en 1, por lo que first corresponde al atributo número 1).

Copy: Realiza una copia de un conjunto de atributos en los datos. Este filtro es útil en conjunción con otros, ya que hay ciertos filtros (la mayoría) que destruyen los datos originales. Como argumentos toma un conjunto de atributos expresados de la misma forma que el filtro anterior. También tiene una opción que es invertSelection que invierte la selección realizada (útil para copiar, por ejemplo, todos los atributos menos uno).

Discretize: Discretiza un conjunto de valores numéricos en rangos de datos. Como parámetros toma los índices de los atributos discretizar (attribute índices) y el número de particiones en que queremos que divida los datos (bins). Si se quiere que las particiones las realice por la frecuencia de los datos y no por el tamaño de estas, existe la opción useEqual-Frecuency. Si esta última opción está activa, se puede variar el peso de las instancias para la definición de los intervalos con la opción DesiredWeightOfInstancesPerInterval. Si al contrario, se tiene en cuenta el número de instancias para la creación de intervalos se puede usar findNumBins que optimiza el procedimiento de confección de los mismos. Otras opciones son makeBinary que convierte los atributos en binario e invertSelection que invierte el rango de los atributos elegidos.

FistOrder: Este filtro realiza una transformación de los datos obteniendo la diferencia de pares consecutivos de datos, suponiendo un dato inicial adicional de valor 0 para conseguir que la cardinalidad del grupo de datos resultante sea la misma que la de los datos origen. Por ejemplo, si los datos son {1 5 4 6}, el resultado al aplicar este filtro será {1 4 -1 2}. Este filtro toma un único parámetro que es el conjunto de atributos con el que obtiene esta transformación.

MakeIndicator: Crea un nuevo conjunto de datos reemplazando un atributo nominal por uno booleano (Asignará "1" si en una instancia se encuentra el atributo nominal seleccionado y "0" en caso contrario). Como atributos este filtro toma el índice del atributo nominal que actuará como indicador, si se desea que la salida del filtro sea numérica o nominal y los índices los atributos sobre los que se quiere aplicar el filtro.

MergeTwoValues: Fusiona dos atributos nominales en uno solo. Toma como argumentos la posición del argumento resultado y la de los argumentos fuente.

Normalize: Este filtro permite normalizar o estandarizar todos los datos numéricos de manera que el rango de los datos pase a ser [0,1]. Para normalizar un vector se utiliza la fórmula:

$$X(i) = \frac{x(i)}{\sqrt{\sum_{i=1}^{n} x(i)^2}}$$

Obfuscate: Este filtro permite ofuscar todas las cadenas de texto de los datos. Este filtro es muy útil si se desea compartir una base de datos pero no se quiere compartir información privada. Un ejemplo de este filtro se encuentra en la tabla 5.2.

Tabla 5.2 Filtro Obfuscate de Weka

APELLIDO	Obfuscate APELLIDO
DELGADO	V1
ALVAREZ	V2
RODRIGUEZ	V3
GONZALES	V4
ALVAREZ	V2
RODRIGUEZ	V3

PKIDiscretize: Discretiza atributos numéricos (igual que Discretize), pero el número de intervalos es igual a la raíz cuadrada del número de valores definidos. Como argumentos tiene AttributeIndices para especificar los atributos que van a ser filtrados, DesiredWeightOfInstancesPerInterval para establecer valores deseados en la conformación de los intervalos de igual frecuencia, InvertSelection que invierte la selección de los atributos a ser discretizados y MakeBinary que convierte el resultado de los atributos a resultados binarios.

RandomProjection: Este filtro permite reducir la dimensionalidad de los datos (útil cuando el conjunto de datos es muy grande) proyectándola en un sub-espacio

de menor dimensionalidad, utilizando para ello una matriz aleatoria. A pesar de reducir la dimensionalidad de los datos resultantes se procura conservar la estructura y propiedades fundamentales de los mismos. El funcionamiento se basa en el siguiente producto de matrices:

$$X(i \times n) * R(n \times m) = Xrp(i \times m)$$

Siendo X la matriz de datos original de dimensiones i (número de instancias) × n (número de atributos), R la matriz aleatoria de dimensión n (número de atributos) × m (número de atributos reducidos) y Xrp la matriz resultante siendo de dimensión i×m. Como parámetros toma el número de parámetros en los que se quiere aplicar este filtro (numberOfAttributes) y el tipo de distribución de la matriz aleatoria que puede ser:

- Sparse 1: $-\sqrt{3}$ con probabilidad $\frac{1}{6}$, 0 con probabilidad $\frac{2}{3}$ y $\sqrt{3}$ con probabilidad $\frac{1}{6}$.
- Sparse 3: -1 con probabilidad $\frac{1}{2}$ y 1 con probabilidad $\frac{1}{2}$.
- Gaussian: Utiliza una distribución gaussiana.

Además de estos parámetros, también se puede utilizar el número de atributos resultantes después de la transformación expresado en porcentaje del número de atributos totales (percent), la semilla usada para la generación de números aleatorios (seed).

Remove: Este filtro permite borrar uno o más atributos pertenecientes al conjunto de datos; como parámetro es necesario especificar el nombre del o de los atributos a borrar.

RemoveType: Este filtro nos permite borrar grupos de atributos que pertenezcan a determinado tipo de dato; como parámetro es necesario especificar el tipo de dato al cual pertenece el o los atributos a borrar.

RemoveUseless: Este filtro permite eliminar atributos que oscilan menos que un nivel de variación. Es útil para eliminar atributos constantes o con un rango muy

pequeño. Como parámetro se especifica el máximo porcentaje de variación permitido, si este valor obtenido es mayor que la variación obtenida la muestra es eliminada.

ReplaceMissingValues: Este filtro reemplaza todos los valores no definidos por la moda en el caso de que sea un atributo nominal ó la media aritmética si es un atributo numérico; este filtro no necesita parámetro alguno puesto que establece los cambios a realizar teniendo en cuenta el tipo del atributo

Filtros Aplicados A Instancias

Randomize: Este filtro modifica el orden de todas las instancias, sin modificar su contenido; como parámetro se especifica el número semilla para la generación de los números aleatorios que determinan el orden de las instancias.

RemovePercentage: Suprime un porcentaje de muestras.

WEKA es una muy buena herramienta para minería de datos y presenta una gran variedad de filtros de pre-procesamiento y limpieza, pero al ser una herramienta tan técnica y completamente enfocada al descubrimiento de conocimiento en base de datos no resulta de muy fácil manipulación. Además por ser una herramienta de tipo cliente no orientada a la Web, requiere que tal cliente sea una máquina con muchos recursos disponibles de sistema y de memoria.

5.2 WINPURE LIST CLEANER

Es una herramienta que limpia, corrige y estandariza listas o tablas de Excel, Access, archivos de texto y otras bases de datos. Maneja varios tipos de listados incluyendo detalles de direcciones de contactos, listados de direcciones de correo electrónico, categorías de productos, nombres, etc. [4]. Es un producto que contiene 8 módulos en uno:

- Tabla de datos.
- Estadísticas.

- Conversor de cadenas.
- Limpiador de texto.
- Limpiador de columnas.
- Limpiador de correos electrónicos.
- Eliminación de duplicados.
- Mezclar tablas.

Utiliza interfaces muy sencillas e intuitivas, las cuales fueron diseñadas para ahorrar tiempo y para ayudar a obtener mejores beneficios de una base de datos. Cada una de las opciones de la herramienta contiene un botón de ayuda que explica la utilidad del mismo y los beneficios que se pueden obtener de él.

Esta herramienta permite importar hasta dos tablas de datos, para las tablas de datos que se encuentren en formato CSV (Separado por comas), se mostrará una pantalla en la cual se especificará la estructura de las mismas (Delimitador de campos, Delimitador de cadenas, formatos de número, formatos de fecha, Si la primera fila contiene o no los nombres de los atributos, etc.).

Statistics (Estadísticas), permite generar dos tipos de gráficos (Barras, Líneas), teniendo en cuenta los campos vacíos o los campos con datos. Es posible generar las gráficas de todos los atributos de la tabla o de alguno(s) en especial. También presenta un resumen de cada uno de los atributos dependiendo de los tipos de gráfica.

Case Converter (Conversor de cadenas), permite convertir las cadenas de texto de un determinado atributo a Mayúsculas, Minúsculas o por defecto, teniendo en cuenta cierto tipo de pre-fijos y excepciones que hacen referencia a pedazos de cadena que no se deben modificar o que deben seguir un formato determinado.

Text Cleaner (Limpiador de Texto) presenta las siguientes características o filtros:

- Remover espacios en blanco al inicio o durante la cadena.
- Remover espacios dobles y la repetición de dos caracteres que no sean letras o dígitos como $$, //, etc. y reducirlos a un solo caracter.

- Remover caracteres no imprimibles (no identificados) y reemplazarlos por algún caracter en especial.
- Limpieza de atributos que sólo deben tener caracteres
- Convertir ceros (0) en o (O), y unos (1) en l (L).
- Remover caracteres que no sean letras o dígitos como /, $ etc. Excepto espacios, comas, puntos, apostrofes o rayas y reemplazarlos por algún caracter en especial.
- Remover todos los dígitos.
- Limpieza de atribuitos que solo deben tener dígitos
- Convertir o (O) en ceros (0), y l (L) en unos (1)..
- Remover todos los caracteres que no sean dígitos excepto espacios, comas, puntos, apostrofes o rayas.
- Remover todos los espacios, comas, puntos, apostrofes y rayas.
- Remover todos los espacios.

Cada uno de estos filtros se aplicará de acuerdo con las necesidades y con la estructura que deba tener el atributo al cual se le apliquen estos filtros.

Column Cleaner (Limpiador de Columnas) permite reemplazar uno o varios valores de uno o varios campos respectivamente, por un valor en especial o por un valor que se encuentre en la base de datos.

Email Cleaner (Limpiador de Correos Electrónicos) permite eliminar o modificar correos electrónicos erróneos y presenta las posibles sugerencias para que dichos correos queden correctos, contiene listados de dominios y sub-dominios válidos y se basa en la expresión regular que identifica a un correo para determinar cuáles son o no correctos. Contiene un filtro para determinar la cantidad de correos que pertenecen a un dominio, sub-dominio, país o cuenta en especial.

Dupe Remover (Eliminación de Duplicados), permite identificar todos los duplicados reales que se encuentren dentro de la base de datos teniendo en cuenta uno o más atributos, los cuales se convierten en atributos clase para identificación.

Table Matcher (Mezclar Tablas), permite buscar registros duplicados dentro de dos tablas teniendo en cuenta los atributos que resulten comunes en las dos tablas,

además se puede seleccionar uno o más de dichos atributos para realizar la búsqueda, durante este proceso es posible realizar las siguientes tareas con los registros encontrados.

- Excluir → Omite los registros encontrados.
- Eliminar → Da la posibilidad de eliminar los registros de las dos tablas, o dejar uno de alguna de las dos.
- Filtrar y Exportar → Permite filtrar los registros encontrados para ser mostrados en la ventana Data Table y posteriormente realizar la tarea de exportación de dichos registros.

Teniendo en cuenta todo lo anterior, es posible afirmar que Winpure ListCleaner es una poderosa y eficaz herramienta para la limpieza y transformación de datos que contiene muchos filtros eficientes y sencillos de utilizar.

Los principales inconvenientes que presenta esta herramienta comienzan por su alto costo en el mercado lo cual limita enormemente el acceso de los usuarios a ella, también que para su funcionamiento requiere obligatoriamente la presencia de Microsoft Access y por lo tanto la convierte en una herramienta que solo funciona bajo software propietario.

5.3 ALPHAMINER

AlphaMiner es desarrollado por el EBusiness Technology Institute (ETI) de la Universidad de Hong Kong [11]. AlphaMiner es un sistema de minería de Datos, Open Source, desarrollado en java de propósito general pero enfocado al ambiente empresarial. Implementa algoritmos de asociación, clasificación, clustering y regresión logística. Posee una gama amplia de funcionalidad para el usuario, al realizar cualquier proceso de minería dando la posibilidad al usuario de escoger los pasos del proceso KDD que más se ajusten a sus necesidades, por medio de nodos que el analista integra a un árbol KDD siguiendo la metodología Drag & Drop. Es por eso que el proceso KDD en AlphaMiner no sigue un orden estructurado, sino que sigue la secuencia brindada por el propósito u objetivo del analista, además brinda la visualización estadística de distintas maneras, permitiendo un análisis más preciso por parte del analista.

El principal objetivo de AlphaMiner es la inteligencia Comercial Económica o Inteligencia de Negocios (BI Business Intelligence) es uno de los medios más importantes para que las compañías tomen las decisiones comerciales más acertadas.

Las soluciones de BI son costosas y sólo empresas grandes pueden permitirse el lujo de tenerlas. Por tanto las compañías pequeñas tienen una gran desventaja. AlphaMiner proporciona las Tecnología de BI de forma Económica para dichas empresas para que puedan dar soporte a sus decisiones en el cambiante ambiente de negocios. AlphaMiner tiene dos componentes principales:

- Una base de conocimiento.
- Un árbol KDD, que permite integrar nodos artefacto que proporcionan varias funciones para crear, revisar, anular, interpretar y argumentar los distintos análisis de datos.

AlphaMiner implementa los siguientes pasos del proceso KDD:

- Acceso de diferentes formas a las fuentes datos.
- Exploración de datos de diferentes maneras.
- Preparación de datos.
- Vinculación de los distintos modelos mineros.
- Análisis a partir de los modelos.
- Despliegue de modelos al ambiente empresarial.

La característica más importante de AlphaMiner, es su capacidad para almacenar los datos después del núcleo de minería en una base de conocimiento, que puede ser reutilizada. Esta función aumenta su utilidad significativamente, y brinda un gran apoyo logístico al nivel estratégico de una empresa. Aventajando cualquier sistema tradicional de manipulación de datos. AlphaMiner proporciona una funcionalidad adicional para construir los modelos de minería de Datos, conformando una sinergia entre los distintos algoritmos de minería.

Al ser ALPHAMINER una herramienta para minería de datos, sus algoritmos para Limpieza de Datos son limitados.

5.4 TARIY KDD

Es una herramienta genérica para el Descubrimiento de Conocimiento en Bases de Datos, débilmente acoplada con el Sistema Gestor de Base de Datos PostgreSQL inicialmente. También puede establecer conexiones con cualquier Sistema Gestor de Base de Datos a través de JDBC (Java Database Connectivity) [12].

Comprende cuatro módulos que cubren la conexión a archivos planos y bases de datos relacionales, un módulo de utilidades con clases y librerías comunes, un módulo kernel que reúne las etapas de pre-procesamiento, minería y visualización, y el módulo de interfaz gráfica de usuario.

Es una excelente herramienta para Minería de Datos la cual contiene varios filtros de limpieza y transformación, la principal desventaja con respecto a EXDACLET es que tales filtros son muy pocos y pueden generar posibles limitaciones a la hora de limpiar y/o transformar los datos para la aplicación de técnicas de minería de datos.

Esta herramienta implementa los siguientes filtros para limpieza: *Remove Missing, Update Missing, Selection, Range, Reduction, Codification, Replace Value, Numeric Range y Discretize*

RemoveMissing elimina todas las transacciones que contengan campos vacíos.

UpdateMissing reemplaza los campos vacios de un atributo específico, por un valor otorgado por el analista.

Selection hace una selección de atributos y del atributo objetivo sobre un conjunto de entrada.

Range escoge una muestra sobre un conjunto de entrada.

Reduction hace una reducción en el número de transacciones, manteniéndolas o eliminándolas.

Codification realiza una codificación sobre el conjunto de datos de entrada.

ReplaceValue reemplaza uno o varios valores de un atributo seleccionado.

Numericrange elimina los valores de un atributo numérico, que están por fuera de un rango determinado por el analista.

Discretize transforma un valor numérico discontinuo en un rango continuo.

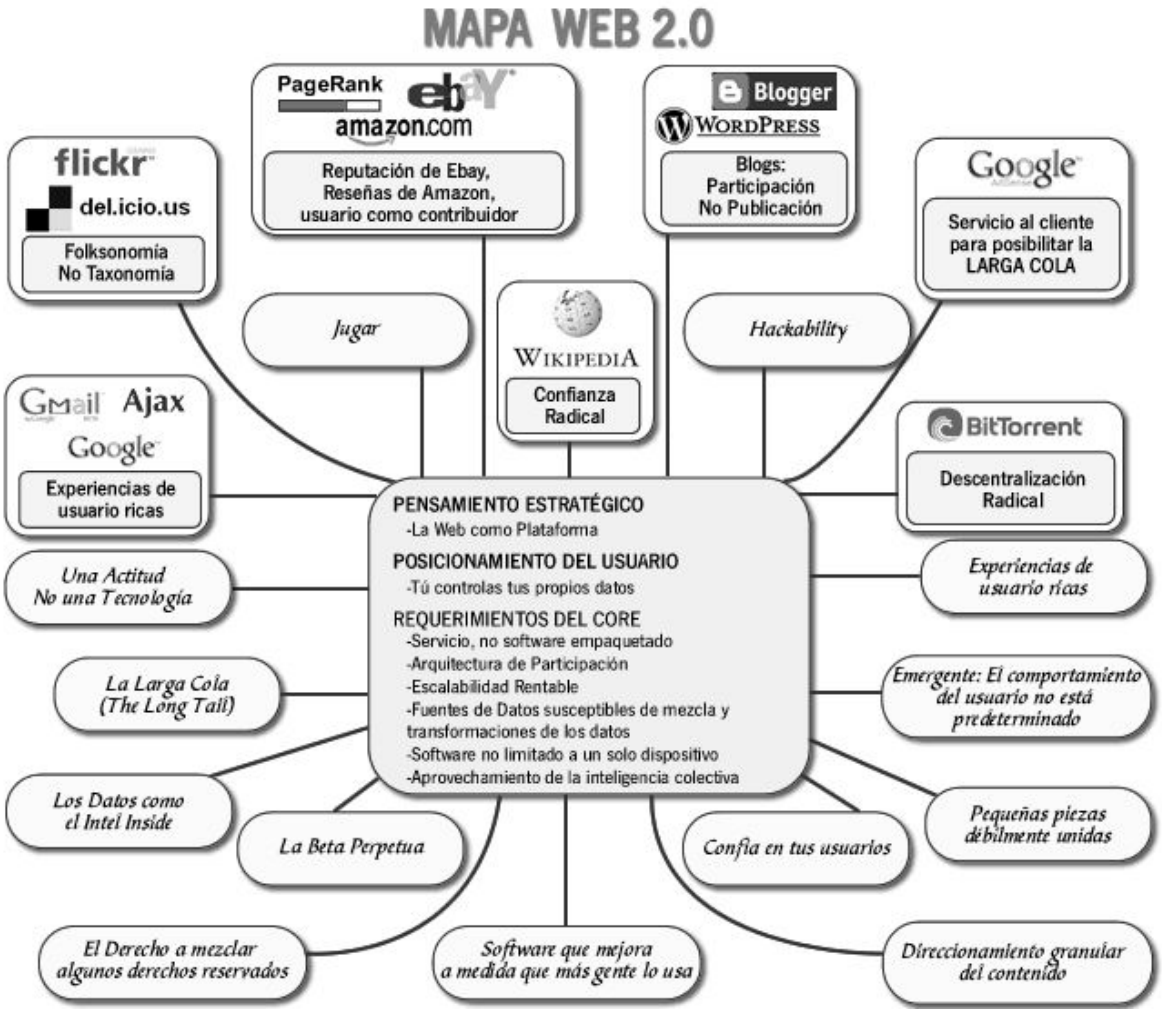
La principal ventaja que ofrece EXDACLET con respecto a todas las herramientas anteriormente mencionadas, es que es una aplicación Web y por lo tanto es centralizada y no requiere que los usuarios tengan un computador de altas especificaciones para trabajar con ella dado que reside en un servidor.

6. TECNOLOGIAS UTILIZADAS EN EXDACLET

6.1 WEB 2.0

La Web 2.0 es la representación de la evolución de las aplicaciones tradicionales hacia aplicaciones web enfocadas al usuario final. El Web 2.0 es una actitud y no precisamente una tecnología. Se trata de aplicaciones que generen colaboración y de servicios que reemplacen las aplicaciones de escritorio [29] [30]. En la figura 6.1 se muestra el mapa de la Web 2.0.

Figura 6.1 Mapa de la Web 2.0



Según O'Reilly, la "Web 2.0 es la revolución de los negocios en la industria de la computación, logrando así, que el internet se convierta en una plataforma" [30].

La forma más fácil de comprender lo que significa la Web 2.0 es a través de ejemplos. Se puede comparar servicios web que marcan claramente la evolución hacia la Web 2.0 en la tabla 6.1.

Tabla 6.1 Servicios en la Web 1.0 y Web 2.0

WEB 1.0	**WEB 2.0**
Doubleclick	Google AdSense (Servicios Publicidad)
Ofoto	Flickr (Comunidades fotográficas)
Akamai	BitTorrent (Distribución de contenidos)
mp3.com	Napster (Descargas de música)
Britannica Online	Wikipedia (Enciclopedias)
Sitios personales	Blogs (Páginas personales)
Especulación con dominios	Optimización en motores de búsqueda
Page views	Cost per click
CMSs	Wikis (Manejo de contenidos)
Categorías/Directorios	Tagging

La Web 2.0 no significa precisamente que existe una receta para que todas las aplicaciones Web entren en este esquema. Sin embargo, existen varias tecnologías que están utilizándose actualmente y que se deben examinar con más cuidado en busca de seguir evolucionando junto a la Web.

Algunas tecnologías que dan vida a un proyecto Web 2.0:

- Transformar software de escritorio hacia la plataforma Web.
- Respeto a los estándares del XHTML.
- Separación de contenido del diseño con uso de hojas de estilo.
- Sindicación de contenidos.
- Ajax (Asincronical JavaScript and xml).
- Uso de Flash, Flex o Lazlo.
- Uso de Ruby on Rails para programar páginas dinámicas.
- Utilización de redes sociales al manejar usuarios y comunidades.
- Dar control total a los usuarios en el manejo de su información.

- Proveer APis o XML para que las aplicaciones puedan ser manipuladas por otros.

El principal objetivo en conclusión que tiene la Web 2.0 es hacer que la interacción que se da entre el usuario e internet resulte lo más amigable posible involucrando no solo manejo de contenidos sino también el desarrollo e interacción con aplicaciones.

6.2 LENGUAJE DE PROGRAMACION PHP

PHP, acrónimo de "PHP: Hypertext Preprocessor", es un lenguaje "Open Source" interpretado de alto nivel, especialmente pensado para desarrollos Web y el cual puede ser embebido en páginas HTML. La mayoría de su sintaxis es similar a C, Java y Perl y es fácil de aprender. La meta de este lenguaje es permitir escribir a los creadores de páginas web, páginas dinámicas de una manera rápida y fácil, y en la actualidad en conjunto con la Web 2.0 y SGBD (Sistemas Gestores de Bases de Datos) tiene como segunda meta además del desarrollo de páginas, el desarrollo de aplicaciones [31].

PHP puede ser utilizado en cualquiera de los principales sistemas operativos del mercado, incluyendo Linux, muchas variantes Unix (incluyendo HP-UX, Solaris y OpenBSD), Microsoft Windows, Mac OS X, RISC OS y probablemente algunos más. PHP soporta la mayoría de servidores Web de hoy en día, incluyendo Apache, Microsoft Internet Information Service, Personal Web Server, Netscape e iPlanet, Oreilly Website Pro server, Caudium, Xitami, OmniHTTPd y muchos otros. PHP tiene módulos disponibles para la mayoría de los servidores, para aquellos otros que soporten el estándar CGI, PHP puede usarse como procesador CGI.

De modo que, con PHP el usuario tiene la libertad de elegir el sistema operativo y el servidor de su gusto. También tiene la posibilidad de usar programación procedimental o programación orientada a objetos. Aunque no todas las características estándar de la programación orientada a objetos están implementadas en la versión actual de PHP, muchas bibliotecas y aplicaciones grandes (incluyendo la biblioteca PEAR) están escritas íntegramente usando programación orientada a objetos.

Quizás la característica más importante y destacable de PHP es su soporte para una gran cantidad de bases de datos. Escribir una interfaz vía Web para una base de datos es una tarea simple con PHP. Las bases de datos que están soportadas actualmente se muestran en la tabla 6.2.

Tabla 6.2 Bases de datos soportadas por PHP

Adabas D	Ingres	Oracle (OCI7 and OCI8)
dBase	InterBase	Ovrimos
Empress	FrontBase	PostgreSQL
FilePro (read-only)	mSQL	Solid
Hyperwave	Direct MS-SQL	Sybase
IBM DB2	MySQL	Velocis
Informix	ODBC	Unix dbm

6.3 JAVASCRIPT

JavaScript es un lenguaje interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas Web, con una sintaxis semejante a la del lenguaje Java y el lenguaje C [32].

Al contrario que Java, JavaScript no es un lenguaje orientado a objetos propiamente dicho, ya que no dispone de Herencia, es más bien un lenguaje basado en prototipos, ya que las nuevas clases se generan clonando las clases base (prototipos) y extendiendo su funcionalidad [33][34].

Todos los navegadores interpretan el código JavaScript integrado dentro de las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del **DOM** (**Document Object Model** - 'Modelo de Objetos de Documento', es una forma de representar los elementos de un documento estructurado (tal como una página web HTML o un documento XML) como objetos que tienen sus propios métodos y propiedades.).

El lenguaje Javascript fue inventado por Brendan Eich en la empresa Netscape Communications, fabricante de los primeros navegadores web comerciales.

Apareció por primera vez en el producto de Netscape llamado Netscape Navigator 2.0 [32].

Tradicionalmente, se venía utilizando en páginas web HTML, para realizar tareas y operaciones en el marco de la aplicación únicamente cliente, sin acceso a funciones del servidor. JavaScript se ejecuta en el agente de usuario al mismo tiempo que las sentencias van descargándose junto con el código HTML [33].

Los autores inicialmente lo llamaron Mocha y más tarde LiveScript pero fue rebautizado como JavaScript en un anuncio conjunto entre Sun Microsystems y Netscape, el 4 de diciembre de 1995.

6.4 AJAX

AJAX, acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Éstas se ejecutan en el cliente, es decir, en el navegador de los usuarios y mantiene comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre la misma página sin necesidad de recargarla. Esto significa aumentar la interactividad, velocidad y usabilidad en la misma.

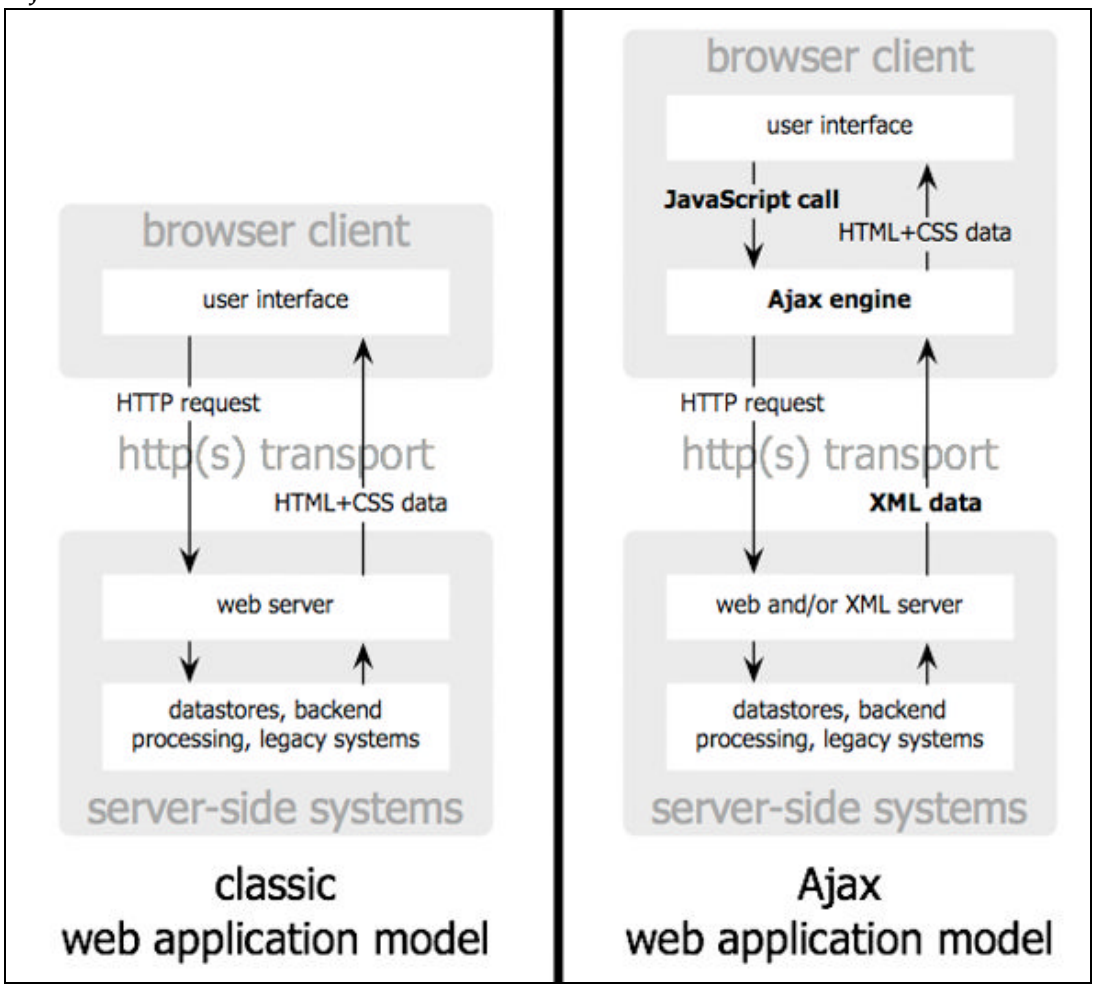
AJAX es una combinación de tres tecnologías ya existentes, **JavasScript, XHTML (o HTML)** y **hojas de estilos en cascada (CSS)** para el diseño que acompaña a la información [35]. En la figura 6.2 se muestra la comparación existente entre el modelo tradicional de desarrollo de aplicaciones Web con el modelo basado en AJAX.

6.4.1 Ventajas

- **Mejor uso del ancho de banda**: Al ser generado el código HTML localmente dentro del navegador, las páginas que utilizan AJAX se pueden cargar mucho más rápido dado que la información proveniente desde el servidor es de poco tamaño y el resto de la presentación de la página no tiene que ser re-dibujado en cada actualización de la misma.

- **Separación de los datos, formatos, estilos y funciones**: Aunque AJAX puede parecer una gran mezcla de lenguajes y técnicas, y que los programadores pueden optar y adoptar miles de formas para utilizarlos, resultan importantes por motivos como el envío de datos crudos o contenido a ser enviado vía Web el cual es incrustado en XML o derivada de bases de datos del lado del servidor, dar formato o estructurar la página Web la cual se construye en HTML o XHTML y se vuelve disponible para la manipulación en el DOM, los elementos de estilo de la pagina web los cuales son almacenados en CSS y la funcionalidad que se da por la combinación de JavaScript en el navegador del cliente, la comunicación cliente servidor y los códigos o programas del lado del servidor que reciben las peticiones del cliente y responden apropiadamente.

Figura 6.2 El modelo tradicional para aplicaciones Web comparado con el modelo AJAX



6.4.2 Desventajas

- Las páginas creadas dinámicamente no se guardan en el cache del navegador y por lo tanto no pueden ser recuperadas del historial del mismo.
- No se puede guardar un estado en especial de la página durante la navegación.
- **Tiempos de latencia por cada petición**: Pueden ocurrir algunos problemas visuales durante la carga de información tras realizarse una petición, la mejor forma de solución a esto es la generación de mensajes de carga y espera.
- **Confianza en JavaScript**: Esta es una desventaja debido a que JavaScript es frecuentemente implementado e interpretado de formas diferentes en diferentes navegadores.

6.5 XAJAX

XAJAX es una biblioteca de código abierto para PHP que permite crear de manera fácil y simple aplicaciones Web basadas en AJAX usando además HTML, CSS, y JavaScript [36].

Las aplicaciones desarrolladas con XAJAX pueden comunicarse asincrónicamente con funciones que se encuentran del lado del servidor y así actualizar el contenido de una página sin tener que recargarla nuevamente.

Es una biblioteca puramente centrada en PHP. Con la introducción de la misma, el manejo de AJAX en PHP se hace mucho más sencillo y sobre todo solo se necesita escribir sentencias muy cortas y simples en JavaScript, por lo cual es una de las ventajas frente a otras soluciones. Otras de las características son:

- XAJAX es compatible con Firefox, Mozilla, Internet Explorer y Safari.
- XAJAX puede ser usado para actualizar estilos, clases CSS, botones de selección, casillas de verificación y botones de radio o cualquier otro atributo de un elemento.

Cada función registrada para ser accesible a través de XAJAX puede tener distintos tipos de petición.

6.6 CSS (Cascading Style Sheets)

Las hojas de estilo en cascada (Cascading Style Sheets, CSS) son un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML) [37].

La idea que se encuentra detrás del desarrollo de CSS es separar la estructura de un documento de su presentación [38]. Las ventajas de utilizar CSS (u otro lenguaje de estilo) son:

- Control centralizado de la presentación de un sitio Web completo con lo que se agiliza de forma considerable la actualización del mismo.
- Los Navegadores permiten a los usuarios especificar su propia hoja de estilo local que será aplicada a un sitio Web, con lo que aumenta considerablemente la accesibilidad. Por ejemplo, personas con deficiencias visuales pueden configurar su propia hoja de estilo para aumentar el tamaño del texto o remarcar más los enlaces.
- Una página puede disponer de diferentes hojas de estilo según el dispositivo que la muestre o incluso a elección del usuario. Por ejemplo, para ser impresa, mostrada en un dispositivo móvil, o ser "leída" por un sintetizador de voz.
- El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño.

6.7 EXTJS

EXTJS es una biblioteca hecha en JavaScript basada en AJAX, utiliza al máximo objetos diseñados en JavaScript tales como Tablas, Ventanas, Mensajes de Dialogo, etc. Al ser una biblioteca hecha en JavaScript significa que puede ser utilizada en cualquier espacio de trabajo del lado del servidor [39]. Se utiliza bajo la licencia LGPL aunque también cuenta con licencias comerciales.

6.8 JSON

JSON, acrónimo de "JavaScript Object Notation", es un formato ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript pero no requiere el uso de XML [40].

La simplicidad de JSON ha dado lugar a la generalización de su uso, especialmente como alternativa a XML en AJAX. Una de las supuestas ventajas de JSON sobre XML como formato de intercambio de datos en este contexto es que es mucho más sencillo escribir un analizador semántico de JSON.

La lista de lenguajes soportados incluye ActionScript, C, C#, ColdFusion, Common Lisp, E, Java, JavaScript, ML, Objective CAML, Perl, PHP, Python, Rebol, Ruby, y Lua.

Una de las principales características de JSON es que permite la transferencia de los datos en todos sus tipos, incluyendo arrays, booleans, integers, etc. e incluso de funciones.

6.9 WAMP SERVER

Es un entorno para desarrollo Web el cual permite crear aplicaciones con Apache, PHP y Mysql. Es un paquete de instalación para Windows que configura los tres servicios en uno [41].

6.10 SERVIDOR APACHE

El servidor HTTP Apache es un software (libre) servidor HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etc.), Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual [42]. Cuando comenzó su desarrollo en 1995 se basó inicialmente en código del popular NCSA HTTPd 1.3, pero más tarde fue reescrito por completo. Su nombre se debe a que originalmente Apache consistía solamente en un conjunto de parches a aplicar al servidor de NCSA. Era, en inglés, a patchy server (un servidor "parcheado").Las principales ventajas que ofrece este servidor son:

- Modular.
- Open Source.
- Multi-plataforma.
- Extensible.
- Popular.
- Gratuito.

6.11 MYSQL

MySQL es un sistema de gestión de base de datos relacional, multihilo y multiusuario [43].

Los Sistemas de gestión de base de datos son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta.

MySQL Es muy utilizado en aplicaciones web, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores. Su popularidad como aplicación Web está muy ligada a PHP, que a menudo aparece en combinación con MySQL. MySQL es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones Web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones.

7. ANALISIS, DISEÑO Y DESARROLLO DE EXDACLET

En todas las disciplinas de la Ingeniería se hace evidente la importancia de los modelos ya que describen el aspecto y la conducta de "algo". Ese "algo" puede existir, estar en un estado de desarrollo o estar, todavía, en un estado de planeación. Es en este momento cuando los diseñadores del modelo deben tanto investigar como analizar los requerimientos del producto terminado y dichos requerimientos pueden incluir áreas tales como funcionalidad, desempeño y confiabilidad. Además, a menudo, el modelo es dividido en un número de vistas, cada una de las cuales describe un aspecto específico del producto o sistema en construcción.

7.1 LENGUAJE UNIFICADO DE MODELADO

(UML, por sus siglas en inglés, Unified Modeling Language). El modelado sirve no solamente para los grandes sistemas, aun en aplicaciones de pequeño tamaño se obtienen beneficios de modelado, sin embargo es un hecho que entre más grande y más complejo es el sistema, más importante es el papel que juega el modelado por una simple razón: "El hombre hace modelos de sistemas complejos porque no puede entenderlos en su totalidad".

Es así como UML es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

El punto importante para notar aquí es que UML es un "lenguaje" para especificar y no un método o un proceso. UML se usa para definir un sistema de software; para detallar los artefactos en el sistema; para documentar y construir, es el lenguaje en el que está descrito el modelo. UML se puede usar en una gran variedad de formas para soportar una metodología de desarrollo de software, pero no especifica en sí mismo qué metodología o proceso usar.

UML prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan.

UML se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real. Además ofrece nueve diagramas en los cuales modelar sistemas.

- Diagramas de Casos de Uso para modelar los procesos.
- Diagramas de Secuencia para modelar el paso de mensajes entre objetos.
- Diagramas de Colaboración para modelar interacciones entre objetos.
- Diagramas de Estado para modelar el comportamiento de los objetos en el sistema.
- Diagramas de Actividad para modelar el comportamiento de los Casos de Uso, objetos u operaciones.
- Diagramas de Clases para modelar la estructura estática de las clases en el sistema.
- Diagramas de Objetos para modelar la estructura estática de los objetos en el sistema.
- Diagramas de Componentes para modelar componentes.
- Diagramas de Implementación para modelar la distribución del sistema.

7.1.1 Diagramas de clases Las clases representan los bloques de construcción más importantes de cualquier sistema orientado a objetos. Una clase es una descripción de un conjunto de objetos que comparten los mismos atributos, operaciones, relaciones y semántica. Los Diagramas de Clases son utilizados durante el proceso de Análisis y Diseño de los sistemas informáticos donde se crea el diseño conceptual de la información que se manejará en el sistema, los componentes que se encargarán del funcionamiento y la relación entre uno y otro. Un diagrama de clases presenta las clases del sistema con sus relaciones estructurales y de herencia. La definición de clase incluye definiciones para atributos y operaciones. Durante el análisis del sistema, el diagrama se desarrolla buscando una solución ideal. Durante el diseño, se usa el mismo diagrama, y se modifica para satisfacer los detalles de las implementaciones. Un diagrama de clases está compuesto por los siguientes elementos:

- Clase: atributos, métodos y visibilidad.

- Relaciones: Herencia, Composición, Agregación, Asociación y Uso.

7.1.2 Diagramas de paquetes Cualquier sistema grande se debe dividir en unidades más pequeñas, de modo que las personas puedan trabajar con una cantidad de información limitada, a la vez y de modo que los equipos de trabajo no interfieran con el trabajo de los otros. Un paquete es una parte de un modelo. Cada parte del modelo debe pertenecer a un paquete. Pero para ser funcional, la asignación debe seguir un cierto principio racional, tal como funcionalidad común, implementación relacionada y punto de vista común. UML no impone una regla para componer los paquetes.

Los paquetes ofrecen un mecanismo general para la organización de los modelos/subsistemas agrupando elementos de modelado. Cada paquete corresponde a un sub-modelo (subsistema) del modelo (sistema). Los paquetes son unidades de organización jerárquica de uso general de los modelos de UML. Pueden ser utilizados para el almacenamiento, el control de acceso, la gestión de la configuración y la construcción de bibliotecas que contengan fragmentos reutilizables del modelo. Un paquete puede contener otros paquetes, sin límite de anidamiento pero cada elemento pertenece a (está definido en) sólo un paquete. El Diagrama de Paquetes, muestra como un sistema está dividido en agrupaciones lógicas mostrando las dependencias entre esas agrupaciones. Dado que normalmente un paquete está pensado como un directorio, los diagramas de paquetes suministran una descomposición de la jerarquía lógica de un sistema.

Los Paquetes están normalmente organizados para maximizar la coherencia interna dentro de cada paquete y minimizar el acoplamiento externo entre los paquetes. Los paquetes son buenos elementos de gestión. Cada paquete puede asignarse a un individuo o a un equipo, y las dependencias entre ellos pueden indicar el orden de desarrollo requerido.

Los paquetes se dibujan como rectángulos, las dependencias se muestran como flechas con líneas discontinuas. El operador "::" permite designar una clase definida en un contexto distinto del actual.

7.1.3 Diagramas de casos de uso Este diagrama muestra el comportamiento del sistema visto desde la perspectiva del usuario, indica la funcionalidad a cumplir,

es una especie de diagrama de comportamiento. UML define una notación gráfica para representar casos de uso llamada modelo de casos de uso, representados por elipses y los actores están representados por las figuras humanas. Una caja define los límites del sistema, por ejemplo, los casos de uso se muestran como parte del sistema que está siendo modelado, los actores no. Existen 3 relaciones principales entre los casos de uso:

- **Include**. En una forma de interacción, un caso de uso dado puede "incluir" otro. El primer caso de uso a menudo depende del resultado del caso de uso incluido. Esto es útil para extraer comportamientos verdaderamente comunes desde múltiples casos de uso a una descripción individual. La notación es una flecha rayada desde el caso de uso que lo incluye hasta el caso de uso incluido, con la etiqueta "«include»". Este uso se asemeja a una expansión de una macro donde el comportamiento del caso incluido es colocado dentro del comportamiento del caso de uso base. No hay parámetros o valores de retorno.
- **Extend**. En otra forma de interacción, un caso de uso dado, (la extensión) puede extender a otro. Esta relación indica que el comportamiento del caso de uso extensión puede ser insertado en el caso de uso extendido bajo ciertas condiciones. La notación es una flecha rayada desde el caso de uso extensión al caso de uso extendido, con la etiqueta «extend». Esto puede ser útil para lidiar con casos especiales, o para acomodar nuevos requisitos durante el mantenimiento del sistema y su extensión.
- **Generalization**. En la tercera forma de relación entre casos de uso, existe una relación generalización/especialización. Un caso de uso dado puede estar en una forma especializada de un caso de uso existente. La notación es una línea solida terminada en un triángulo dibujado desde el caso de uso especializado al caso de uso general.

7.1.4 El Proceso Racional Unificado o RUP (Rational Unified Process) Los orígenes de RUP se remontan al modelo espiral, es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. RUP es en realidad un refinamiento realizado por Rational Software del más genérico Proceso Unificado.
Sus principales características son:

- Forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo).

- Pretende implementar las mejores prácticas en Ingeniería de Software.
- Desarrollo iterativo.
- Administración de requisitos.
- Uso de arquitectura basada en componentes.
- Control de cambios.
- Modelado visual del software.
- Verificación de la calidad del software.

El RUP es un producto de Rational (IBM). Se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. Incluye artefactos (que son los productos tangibles del proceso como por ejemplo, el modelo de casos de uso, el código fuente, etc.) y roles (papel que desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso). El RUP divide el proceso de desarrollo en ciclos, teniendo un producto final al final de cada ciclo, cada ciclo se divide en fases que finalizan con un hito donde se debe tomar una decisión importante:

- **Inicio**: se hace un plan de fases, se identifican los principales casos de uso y se identifican los riesgos.
- **Elaboración**: se hace un plan de proyecto, se completan los casos de uso y se eliminan los riesgos.
- **Construcción**: se concentra en la elaboración de un producto totalmente operativo y eficiente y el manual de usuario.
- **Transición**: se implementa el producto en el cliente y se entrena a los usuarios, como consecuencia de esto suelen surgir nuevos requisitos a ser analizados.

Fases del RUP:

- Establece oportunidad y alcance.
- Identifica las entidades externas o actores con las que se trata.
- Identifica los casos de uso.

7.2 ANALISIS UML

Para el análisis y diseño de EXDACLET se utilizó el RUP (Rational Unified Process) con la notación UML.

7.2.1 Funciones

Ref#	Función	Cat.	Atributo	Detalles y Restricciones	Cat.
R1	Ejecutar la Herramienta	Ev	Interfaz		Ob
R1.1	Mostrar Ventana de logueo	Ev	Interfaz	Usuarios Autorizados	Ob
R1.2	Mostrar Interfaz Gráfica Principal	Ev	Interfaz	Usuarios Autorizados	Ob
R2	Mostrar mensajes de error	Ev	Interfaz	Cuadros de Dialogo	Ob
R3	Permitir carga y transferencia de archivos	Ev	Interfaz		Ob
R3.1	Mostrar resumen del archivo plano	Ev	Interfaz	Definir la estructura, el formato y el tipo de datos	Ob
R3.2	Recorrer Archivo y cargar datos	Oc	Información	El archivo debe pertenecer a los formatos CSV,XLS,SQL o ARFF	Ob
R4	Permitir la aplicación de filtros de limpieza	Ev	Interfaz	Debe existir una tabla de datos cargada	Ob
R4.1	Permitir limpiar datos nulos en campos numéricos	Ev	Interfaz		Op
R4.2	Permitir limpiar datos nulos en campos de texto	Ev	Interfaz		Op
R4.3	Permitir borrar espacios de cadenas de texto	Ev	Interfaz		Op
R4.4	Permitir definir reglas para limpieza de datos	Ev	Interfaz		Op
R4.5	Permitir limpiar campos de correo electrónico	Ev	Interfaz		Op
R5	Permitir la aplicación de filtros de transformación	Ev	Interfaz	Debe existir una tabla de datos cargada	Ob
R5.1	Permitir Discretizar valores de campos numéricos	Ev	Interfaz		Op
R5.2	Permitir Normalizar valores de campos numéricos	Ev	Interfaz		Op
R5.3	Permitir transformar campos de texto a Mayúsculas, minúsculas o a una longitud determinada	Ev	Interfaz		Op
R5.4	Permitir buscar caracteres no imprimibles en la tabla	Ev	Interfaz		Op

Ref#	Función	Cat	Atributo	Detalles y Restricciones	Cat.
R5.5	Permitir acciones de transformación sobre atributos (cambiar tipo, nombre, agregar, eliminar, limpiar)	Ev	Interfaz		Op
R5.6	Permitir binarizar una tabla de datos	Ev	Interfaz		Op
R5.7	Permitir codificar y/o decodificar una tabla.	Ev	Interfaz		Op
R6	Permitir la aplicación de filtros de selección	Ev	Interfaz	Debe existir una tabla de datos cargada	Ob
R6.1	Proveer algoritmos para tareas de selección	Ev	Interfaz		Op
R6.1.1	Implementar algoritmo SOUNDEX	Ev	Interfaz		Op
R6.1.2	Implementar algoritmo LEVENSHTEIN	Ev	Interfaz		Op
R6.1.3	Implementar algoritmo DOUBLE METAPHONE	Ev	Interfaz		Op
R6.1.4	Implementar algoritmo JARO-WINKLER	Ev	Interfaz		Op
R6.2	Permitir la aplicación de algoritmos para unión de tablas	Ev	Interfaz		Op
R7	Mostrar estadísticas detalladas de una tabla.	Ev	Interfaz	Debe existir una tabla cargada	Ob
R8	Permitir eliminar tablas o archivos	Ev	Interfaz		Op
R9	Permitir guardar tablas de resultados	Ev	Interfaz		Op
R10	Permitir exportar tablas	Ev	Interfaz		Op
R11	Proveer opciones de edición de tablas	Ev	Interfaz		Op
R12	Permitir transferencia asíncrona de información	Oc	Información		Ob
R13	Agilizar transferencia de datos tras la aplicación de filtros	Oc	Información		Ob

Categoria: Evidente → Ev, Oculto → Oc, Obligatorio → Ob, Opcional → Op

7.2.2 Casos de uso A continuación se muestran los Casos de Uso que hacen parte de EXDACLET.

CASOS DE USO DE PRECARGA

Caso de uso Importar

Caso de Uso	Importar Archivo
Actor Principal	Usuario
Otros Actores	Sistema, Sistema Gestor de Bases de Datos (SGBD)
Precondiciones	El usuario debe haberse registrado en el sistema previamente.
Poscondiciones	El archivo seleccionado es cargado en el directorio asignado para el usuario, ubicado en el servidor.
Flujo Básico	
1. El **usuario** selecciona el modulo de precarga. 2. El **usuario** selecciona el tipo de archivo a cargar. 3. El **sistema** verifica el tipo de archivo. 4. El **sistema** copia el archivo seleccionado en el directorio asignado para el usuario, ubicado en el servidor. 5. Si el archivo es de formato ARFF entonces se llama al Caso de Uso **ImportarARFF**; Si el archivo es de formato SQL entonces se llama al Caso de Uso **ImportarSQL**; Si el archivo es de formato XLS entonces se llama al Caso de Uso **ImportarXLS**; Si el archivo es de formato CSV entonces se llama al Caso de Uso **ImportarCSV**.	
Flujos Alternativos	
3a. Si el archivo no puede ser copiado al Servidor. 1. El sistema muestra el mensaje "The file cannot be uploaded". 3b. Si el archivo ya existe en el directorio asignado al **usuario** en el servidor. 1. El sistema muestra el mensaje "The file already exists". 3c.Si el archivo no corresponde con el formato específico. 1. El sistema muestra el mensaje "Invalid file extension". 3d. Si el tamaño del archivo no es consistente. 1. El sistema muestra el mensaje "Invalid file size". 3e. Si el archivo ya existe con una extensión diferente. 1. El sistema muestra el mensaje "Invalid file, the file name already exists". El **sistema** pasa nuevamente al Caso de Uso **Importar Archivo**.	

Caso de uso Importar CSV

Caso de Uso	Importar CSV
Flujo Básico	
1. El **usuario** especifica la estructura del archivo y la observa a través de una pre-visualización del mismo. 2. El **usuario** define los tipos de datos de cada uno de los atributos del archivo y los observa a través de una pre-visualización. 3. Se crea en el **SGBD** la tabla asociada al archivo. 4. El **sistema** Verifica la integridad de los datos en el archivo y los importa al **SGBD**. 5. La nueva tabla es registrada en el **sistema**.	
Flujo Alternativo	
4a. Si el archivo no cumple con el formato de archivo CSV (Comma Separated Values). 1. El **sistema** muestra un mensaje de error informando esta situación. 2. Se pasa al caso de uso **Importar CSV.**	
Nota	
1. Si el proceso de importación es cancelado por el usuario, se elimina el archivo físicamente, del SGBD y del sistema.	

Caso de uso Importar Arff

Caso de Uso	Importar Arff
Flujo Básico	
1. El sistema verifica el formato del archivo. 2. Se crea en el **SGBD** la tabla asociada al archivo. 3. El **sistema** Verifica la integridad de los datos en el archivo y los importa al **SGBD**. 4. La nueva tabla es registrada en el **sistema**.	
Flujos Alternativos	
1a. Si el formato no coincide con el formato especificado. 1. El sistema muestra un mensaje en alusión al error que contiene el archivo. 2. Se pasa al caso de uso **Importar Archivo.** 2a. Si la tabla a crear en el **SGBD** ya ha sido creada previamente. 1. El sistema muestra un mensaje error y cancela la importación del archivo. 2. Se pasa al caso de uso **Importar Archivo.**	

Caso de uso Importar XLS

Caso de Uso	Importar XLS
Flujo Básico	
1. El **usuario** especifica la estructura del archivo y la observa a través de una pre-visualización del mismo. 2. El **usuario** define los tipos de datos de cada uno de los atributos del archivo y los observa a través de una pre-visualización. 3. Se crea en el **SGBD** la tabla asociada al archivo. 4. El **sistema** Verifica la integridad de los datos en el archivo y los importa al **SGBD**. 5. La nueva tabla es registrada en el **sistema**.	
Flujo Alternativo	
4a. Si el archivo no cumple con el formato de archivo de Microsoft Excel. 1. El **sistema** muestra un mensaje de error informando esta situación. 2. Se pasa al caso de uso **Importar XLS.**	
Nota	
1. Si el proceso de importación es cancelado por el usuario, se elimina el archivo físicamente, del SGBD y del sistema.	

Caso de uso Importar SQL

Caso de Uso	Importar SQL
Flujo Básico	
1. El sistema verifica el formato del archivo y el contenido de las sentencias. 2. Se crea en el **SGBD** la tabla asociada al archivo. 3. El **sistema** Verifica la integridad de los datos en el archivo y los importa al **SGBD**. 4. La nueva tabla es registrada en el **sistema**.	
Flujos Alternativos	
1a. Si el formato no coincide con el formato especificado. 1. El sistema muestra un mensaje en alusión al error que contiene el archivo. 2. Se pasa al caso de uso **Importar Archivo.** 2a. Si la tabla a crear en el **SGBD** ya ha sido creada previamente. 1. El sistema muestra un mensaje error y cancela la importación del archivo. 2. Se pasa al caso de uso **Importar Archivo.**	

CASOS DE USO CLEANING PROCEDURE

Caso de Uso Seleccionar Filtro de Limpieza.

Caso de Uso	Seleccionar Filtro de Limpieza.
Actor Principal	Usuario
Otros Actores	Sistema, Sistema Gestor de Base de Datos (SGBD).
Precondiciones	El usuario debe haber iniciado sesión en el sistema.
Poscondiciones	Se aplica un determinado filtro de limpieza a los datos.
Flujo Básico	
1. El **usuario** selecciona la tabla sobre la cual desea aplicar un filtro. 2. El **usuario** selecciona el módulo de Cleaning Procedure. 3. El **usuario** selecciona el filtro a aplicar sobre la tabla previamente seleccionada. 4. Si el **usuario** selecciona el filtro Number Null Clean, se llama al caso de uso **Number Null Clean**; Si el **usuario** selecciona el filtro String Null Clean (Mode), se llama al caso de uso **String Null Clean (Mode)**; Si el **usuario** selecciona el filtro String Null Clean Based in Conditions, se llama al caso de uso **String Null Clean Based in Conditions**; Si el **usuario** selecciona el filtro Trim, se llama al caso de uso **Trim**; Si el **usuario** selecciona el filtro Attribute Clear, se llama al caso de uso **Attribute Clear**; Si el **usuario** selecciona el filtro Expert Rule Editor, se llama al caso de uso **Expert Rule Editor**; Si el **usuario** selecciona el filtro Email Cleaner, se llama al caso de uso **Email Cleaner**.	
Flujo Alternativo	
3a. Si el usuario selecciona un filtro y no tiene una tabla seleccionada. 1. El sistema muestra un mensaje de error y pide seleccionar una tabla.	

Caso de Uso Number Null Clean

Caso de Uso	Number Null Clean
Flujo Básico	
1. El **usuario** selecciona el atributo sobre el cual será aplicado el filtro. 2. El **usuario** selecciona el valor por el cual serán remplazados los campos nulos. 3. El **sistema** pregunta al **usuario** si el filtro será aplicado sobra la tabla seleccionada o sobre una nueva tabla. 4. El **sistema** calcula el valor seleccionado por el usuario. 5. El **sistema** modifica los campos nulos por el valor con el cual serán completados.	
Nota	
1. Este filtro solo se puede aplicar a los atributos numéricos.	

Caso de Uso String Null Clean (Moda)

Caso de Uso	String Null Clean (Moda)
Flujo Básico	
6. El **usuario** selecciona el atributo sobre el cual será aplicado el filtro. 7. El **usuario** selecciona el tipo de moda que desea aplicar sobre el atributo. 8. El **sistema** pregunta al **usuario** si el filtro será aplicado sobra la tabla seleccionada o sobre una nueva tabla. 9. Si el tipo de moda seleccionada es en base a condiciones, entonces se pasa al caso de uso **String Null Clean Based on Conditions** 10. El **sistema** ejecuta el filtro sobre el atributo seleccionado.	
Nota	
1. Este filtro solo se puede aplicar a los atributos de tipo cadena.	

Caso de Uso String Null Clean Based on Conditions

Caso de Uso	String Null Clean Based on Conditions
Flujo Básico	
1. El **usuario** selecciona los atributos que servirán como condición para aplicar el filtro sobre el atributo seleccionado en el caso de uso **String Null Clean (Moda).** 2. El **sistema** genera la interfaz de condiciones en base a los atributos seleccionados por el **usuario**. 3. El **usuario** define las condiciones que se deben cumplir para cada atributo con el fin de ejecutar el filtro. 4. El sistema retorna al paso 5 del caso de uso **String Null Clean (Moda).**	
Nota	
1. Cuando el **usuario** cancela la ejecución de este caso de uso y si decidió aplicar el filtro **String Null Clean** en una tabla nueva, entonces, dicha tabla será eliminada del **SGBD** y del **sistema**.	

Caso de Uso Trim

Caso de Uso	Trim
Flujo Básico	
1. El **usuario** selecciona los atributos que serán objeto del filtro. 2. El **sistema** elimina los espacios tanto al inicio como al final de las cadenas de los atributos seleccionados. 3. El **sistema** aplica los cambios en la tabla**.**	
Nota	
1. Este filtro se aplica únicamente sobre los campos de cadenas de texto	

Caso de Uso Attribute Clear

Caso de Uso	Attribute Clear
Flujo Básico	
1. El **usuario** selecciona el atributo que serán objeto del filtro. 2. El **sistema** elimina el contenido del atributo seleccionado. 3. El **sistema** aplica los cambios en la tabla**.**	

Expert Rule Editor

Caso de Uso	Expert Rule Editor
Flujo Básico	
1. El **usuario** escribe en forma de condicional una o más reglas para aplicar en la tabla que ha seleccionado previamente. 2. El **sistema** verifica que la estructura de la o las reglas sea la adecuada. 3. El **sistema** aplica los cambios en la tabla del **SGBD.**	
Flujo Alternativo	
2a. El **sistema** evalúa separadamente cada una de las reglas escritas, si alguna de estas no es adecuada en su gramática o semántica. 1. El **sistema** muestra un mensaje de error indicando donde se encuentra el error. 2. El **sistema** retorna al modo de edición de las condiciones.	

Caso de Uso Email Cleaner

Caso de Uso	Email Cleaner
Flujo Básico	
1. El usuario selecciona el atributo sobre el cual será aplicado el filtro. 2. El sistema verifica que los datos del atributo correspondan a direcciones de correo electrónico validas.	
Flujo Alternativo	
2a. Si al menos una de las direcciones de correo electrónico no coincide con el formato. 1. Se crea en el SGBD una tabla temporal donde se insertan las direcciones de correo que no correspondan con el formato y una sugerencia. 2. El usuario modifica las direcciones de correo electrónico. 3. El sistema aplica los cambios.	

CASOS DE USO TRANSFORM PROCEDURE

Caso de Uso Seleccionar Filtro de transformación.

Caso de Uso	Seleccionar Filtro de transformación.
Actor Principal	Usuario
Otros Actores	Sistema, Sistema Gestor de Bases de Datos (SGBD).
Precondiciones	El usuario debe haber iniciado sesión en el sistema.
Poscondiciones	Se aplica un determinado filtro a los datos.
Flujo Básico	
1. El **usuario** selecciona la tabla sobre la cual desea aplicar un filtro. 2. El **usuario** selecciona el módulo de Transform Procedure. 3. El **usuario** selecciona el filtro a aplicar sobre la tabla previamente seleccionada. 4. Si el **usuario** selecciona el filtro Discretize, se llama al caso de uso **Discretize**; si el **usuario** selecciona el filtro Normalize, se llama al caso de uso **Normalize**; si el **usuario** selecciona el filtro Char Replace, se llama al caso de uso **Char Replace**; si el **usuario** selecciona el filtro Upper Case, se llama al caso de uso **Upper Case**; si el **usuario** selecciona el filtro Lower Case, se llama al caso de uso **Lower Case**; si el **usuario** selecciona el filtro Truncate String, se llama al caso de uso **Truncate String**; si el **usuario** selecciona el filtro Non-Printable Character Search, se llama al caso de uso **Non-Printable Character Search**; si el **usuario** selecciona el filtro Change Attribute Type, se llama al caso de uso	

Change Attribute Type; si el **usuario** selecciona el filtro Change Attribute Name, se llama al caso de uso **Change Attribute Name**; si el **usuario** selecciona el filtro Add Attribute, se llama al caso de uso **Add Attribute**; si el **usuario** selecciona el filtro Delete Attributes, se llama al caso de uso **Delete Attributes**; si el **usuario** selecciona el filtro Table Encoder, se llama al caso de uso **Table Encoder**; si el **usuario** selecciona el filtro Table Decoder, se llama al caso de uso **Table Decoder**.
Flujo Alternativo
3a. Si el usuario selecciona un filtro y no tiene una tabla seleccionada. 1. El sistema muestra un mensaje de error y pide seleccionar una tabla.

Caso de Uso Discretize

Caso de Uso	Discretize
Flujo Básico	
1. El **usuario** selecciona el atributo sobre el cual será aplicado el filtro. 2. El **usuario** escribe la cantidad de intervalos que quiere formar con los datos. 3. El **sistema** calcula los intervalos, modifica el tipo de dato y aplica los intervalos.	
Flujo Alternativo	
2a. Si el **usuario** deja el número por defecto para la cantidad de intervalos. 1. El **sistema** determina la cantidad de intervalos mediante un fórmula.	
Nota	
1. El filtro solo se aplica sobre campos numéricos.	

Caso de Uso Normalize

Caso de Uso	Normalize
Flujo Básico	
1. El **usuario** selecciona el atributo sobre el cual será aplicado el filtro. 2. El **sistema** aplica la fórmula para normalizar los valores de la tabla. 3. El **sistema** actualiza la tabla con los valores ya normalizados.	
Nota	
1. El filtro solo se aplica sobre campos numéricos.	

Caso de Uso Char Replace

Caso de Uso	Char Replace
Flujo Básico	
1. El **usuario** selecciona los atributos sobre los cuales será aplicado el filtro. 2. El **usuario** escribe el caracter a reemplazar en la tabla (Antiguo Caracter). 3. El **usuario** escribe el caracter que reemplazará al antiguo caracter (Nuevo Caracter). 4. El **sistema** busca en los atributos el "Antiguo Caracter" y lo remplaza por el "Nuevo caracter".	
Nota	
1. El filtro solo se aplica sobre campos que contienen cadenas de texto.	

Caso de Uso UpperCase

Caso de Uso	UpperCase
Flujo Básico	
1. El **usuario** selecciona los atributos sobre los cuales será aplicado el filtro. 2. El **sistema** convierte todos a mayúsculas los caracteres que encuentre en los atributos seleccionados. 3. El **sistema** actualiza la tabla con las cadenas de texto en mayúsculas.	
Nota	
1. El filtro solo se aplica sobre campos que contienen cadenas de texto.	

Caso de Uso LowerCase

Caso de Uso	LowerCase
Flujo Básico	
1. El **usuario** selecciona los atributos sobre los cuales será aplicado el filtro. 2. El **sistema** convierte todos a minúsculas los caracteres que encuentre en los atributos seleccionados. 3. El **sistema** actualiza la tabla con las cadenas de texto en minúsculas.	
Nota	
1. El filtro solo se aplica sobre campos que contienen cadenas de texto.	

Caso de Uso Truncate String

Caso de Uso	Truncate String
Flujo Básico	
1. El **usuario** selecciona el atributo sobre el cual será aplicado el filtro. 2. El **usuario** selecciona el tipo de truncado que puede ser por izquierda o derecha. 3. El **usuario** escribe el tamaño de cadena que será truncado. 4. El **sistema** corta las cadenas de texto que encuentre en el atributo seleccionado y aplica los cambios en la tabla.	
Flujo Alternativo	
4a. Si el campo de texto está vacío. 1. El sistema no aplica ningún cambio sobre el campo	
Nota	
1. El filtro solo se aplica sobre campos que contienen cadenas de texto.	

Caso de Uso Non-Printable Character Search

Caso de Uso	Non-Printable Character Search
Flujo Básico	
1. El **usuario** selecciona los atributos sobre los cuales será aplicado el filtro. 2. El **sistema** busca los caracteres no imprimibles sobre los atributos de la tabla seleccionados por el usuario. 3. El **sistema** crea una tabla temporal en el SGBD donde son insertados todos los registros que contienen caracteres no imprimibles.	
Flujo Alternativo	
2a. Si el **sistema** no encuentra caracteres no imprimibles en los atributos seleccionados. 1. El **sistema** no aplica ningún cambio sobre el atributo.	
Nota	
1. El filtro solo se aplica sobre campos que contienen cadenas de texto.	

Caso de Uso Change Attribute Type

Caso de Uso	Change Attribute Type
Flujo Básico	
1. El **usuario** selecciona el atributo sobre el cual será aplicado el filtro. 2. El **usuario** selecciona el nuevo tipo de dato que tendrá el atributo seleccionado. 3. El **sistema** Verifica que los datos del atributo cumplan con el tipo dato que el usuario seleccionó. 4. El **sistema** modifica el tipo de dato del atributo.	
Flujo Alternativo	
3a. Si existen datos que no cumplen con el tipo de dato especificado. 1. El **sistema** genera una tabla temporal con dichos datos que permite que estos sean modificados o eliminados para que el atributo cumpla completamente con el tipo de dato. 2. El **sistema** retorna al paso número 3 del flujo básico.	

Caso de Uso Change Attribute Name

Caso de Uso	Change Attribute Name
Flujo Básico	
1. El **usuario** selecciona el atributo el cual será objeto del cambio de nombre. 2. El **usuario** escribe el nuevo nombre que se le asignará al atributo previamente seleccionado. 3. El **sistema** realiza el cambio de nombre en la tabla del **SGBD**.	

Caso de Uso Add Attribute

Caso de Uso	Add Attribute
Flujo Básico	
1. El **usuario** selecciona el tipo de atributo a crear. 2. El **usuario** escribe el nuevo nombre para el atributo a crear. 3. El **sistema** crea el nuevo atributo en el **SGBD** con el tipo especificado previamente**.**	

Caso de Uso Delete Attributes

Caso de Uso	Delete Attributes
Flujo Básico	
1. El **usuario** selecciona el atributo a eliminar. 2. El **sistema** elimina de la tabla en el **SGBD** el atributo previamente seleccionado**.**	

Caso de Uso Table Encoder

Caso de Uso	Table Encoder
Flujo Básico	
1. El **usuario** selecciona los atributos que desea codificar. 2. El **sistema** identifica los diferentes valores de los atributos previamente seleccionados. 3. El **sistema** asigna un código a cada uno de los diferentes valores identificados. 4. El **sistema** crea una tabla en el **SGBD** que contiene el diccionario de datos de la tabla codificada. 5. El **sistema** cambia los datos de la tabla por los respectivos códigos asignados.	

Caso de Uso Table Decoder

Caso de Uso	Table Decoder
Flujo Básico	
1. El **usuario** selecciona el diccionario de datos con el cual quiere decodificar su tabla. 2. El **sistema** cambia los códigos de la tabla por los respectivos datos que contiene el diccionario de datos.	
Flujo Alternativo	
2a. Si el diccionario de datos seleccionado no coincide con los atributos que contiene la tabla. 1. El **sistema** muestra un mensaje de error.	

CASOS DE USO PARA FILTROS DE SELECCION

Caso de Uso Seleccionar Filtro de Selección.

Caso de Uso	Seleccionar Filtro de Selección.
Actor Principal	Usuario
Otros Actores	Sistema, Sistema Gestor de Bases de Datos (SGBD).
Precondiciones	El usuario debe haber iniciado sesión en el sistema.
Poscondiciones	Se crea una tabla temporal con los datos que son seleccionados por el filtro.
Flujo Básico	
1. El **usuario** selecciona la tabla sobre la cual desea aplicar un filtro. 2. El **usuario** selecciona el módulo de Selection Filters. 3. El **usuario** selecciona el filtro a aplicar sobre la tabla previamente seleccionada. 4. Si el **usuario** selecciona el filtro Max Length, se llama al caso de uso **Max Length**; Si el **usuario** selecciona el filtro Min Length, se llama al caso de uso **Min Length**; si el **usuario** selecciona el filtro Jaro-Winkler Search, se llama al caso de uso **Jaro-Winkler Search**; si el **usuario** selecciona el filtro Double Metaphone Search, se llama al caso de uso **Double Metaphone Search**; si el **usuario** selecciona el filtro LD Search, se llama al caso de uso **LD Search**; si el **usuario** selecciona el filtro Soundex Search, se llama al caso de uso **Soundex Search**; si el **usuario** selecciona el filtro Duplicates Search, se llama al caso de uso **Duplicates Search**; si el **usuario** selecciona el filtro Table Details, se llama al caso de uso **Table Details**; si el **usuario** selecciona el filtro Table Union, se llama al caso de uso **Table Union**.	
Flujo Alternativo	
3a. Si el usuario selecciona un filtro y no tiene una tabla seleccionada. 1. El sistema muestra un mensaje de error y pide seleccionar una tabla.	

Caso de Uso Max Length

Caso de Uso	Max Length
Flujo Básico	
1. El **usuario** selecciona el atributo sobre el cual se realizará la búsqueda de los registros que no cumplan con un tamaño máximo. 2. El **usuario** escribe la cantidad máxima de caracteres que puede contener el campo previamente seleccionada. 3. El **sistema** genera una tabla temporal que contiene todos los registros que no cumplen en el atributo seleccionado con el tamaño máximo especificado por el **usuario.** 4. El **usuario** realiza las acciones pertinentes sobre la nueva tabla.	
Flujo Alternativo	
3a. Si el usuario elimina uno o varios resultados de la nueva tabla 1. El **sistema** elimina también esos registros de la tabla base. 2. El **sistema** actualiza la tabla base y la nueva tabla. 3b. Si el usuario modifica los registros de la nueva tabla. 1. El **sistema** actualiza la tabla base.	

Caso de Uso Min Length

Caso de Uso	Min Length
Flujo Básico	
1. El **usuario** selecciona el atributo sobre el cual se realizará la búsqueda de los registros que no cumplan con un tamaño mínimo. 2. El **usuario** escribe la cantidad mínima de caracteres que puede contener el campo previamente seleccionada. 3. El **sistema** genera una tabla temporal que contiene todos los registros que no cumplen en el atributo seleccionado con el tamaño mínimo especificado por el **usuario.** 4. El **usuario** realiza las acciones pertinentes sobre la nueva tabla.	
Flujo Alternativo	
3a. Si el usuario elimina uno o varios resultados de la nueva tabla 1. El **sistema** elimina también esos registros de la tabla base. 2. El **sistema** actualiza la tabla base y la nueva tabla. 3b. Si el usuario modifica los registros de la nueva tabla. 1. El **sistema** actualiza la tabla base.	

Caso de Uso Jaro Winkler Search

Caso de Uso	Jaro Winkler Search
Flujo Básico	
1. El **usuario** selecciona los atributos que serán llave principal de búsqueda de posibles duplicados. 2. El **sistema** genera una tabla temporal que contiene todos los atributos de la tabla base y todos los registros que fueron encontrados como posibles duplicados además de dos atributos de referencia. 3. El **usuario** realiza las acciones pertinentes sobre la nueva tabla.	
Flujo Alternativo	
3a. Si el usuario elimina uno o varios resultados de la nueva tabla 1. El **sistema** elimina también esos registros de la tabla base. 2. El **sistema** actualiza la tabla base y la nueva tabla. 3b. Si el usuario descarta uno o varios resultados de la nueva tabla 1. El **sistema** mantiene esos registros en la tabla base. 2. El **sistema** actualiza la tabla base y la nueva tabla. 3c. Si el usuario modifica los registros de la nueva tabla. 1. El **sistema** actualiza la tabla base.	

Caso de Double Metaphone Search

Caso de Uso	Double Metaphone Search
Flujo Básico	
1. El **usuario** selecciona los atributos que serán llave principal de búsqueda de posibles duplicados. 2. El **sistema** genera una tabla temporal que contiene todos los atributos de la tabla base y todos los registros que fueron encontrados como posibles duplicados además de dos atributos de referencia. 3. El **usuario** realiza las acciones pertinentes sobre la nueva tabla.	
Flujo Alternativo	
3a. Si el usuario elimina uno o varios resultados de la nueva tabla 1. El **sistema** elimina también esos registros de la tabla base. 2. El **sistema** actualiza la tabla base y la nueva tabla. 3b. Si el usuario descarta uno o varios resultados de la nueva tabla 1. El **sistema** mantiene esos registros en la tabla base. 2. El **sistema** actualiza la tabla base y la nueva tabla. 3c. Si el usuario modifica los registros de la nueva tabla. 1. El **sistema** actualiza la tabla base.	

Caso de Uso LD Search (Levenshtein Distance Search)

Caso de Uso	LD Search
Flujo Básico	
1. El **usuario** selecciona los atributos que serán llave principal de búsqueda de posibles duplicados. 2. El **sistema** genera una tabla temporal que contiene todos los atributos de la tabla base y todos los registros que fueron encontrados como posibles duplicados además de 2 atributos de referencia. 3. El **usuario** realiza las acciones pertinentes sobre la nueva tabla.	
Flujo Alternativo	
3a. Si el usuario elimina uno o varios resultados de la nueva tabla 1. El **sistema** elimina también esos registros de la tabla base. 2. El **sistema** actualiza la tabla base y la nueva tabla. 3b. Si el usuario descarta uno o varios resultados de la nueva tabla 1. El **sistema** mantiene esos registros en la tabla base. 2. El **sistema** actualiza la tabla base y la nueva tabla. 3c. Si el usuario modifica los registros de la nueva tabla. 1. El **sistema** actualiza la tabla base.	

Caso de Uso Soundex Search

Caso de Uso	Soundex Search
Flujo Básico	
1. El **usuario** selecciona los atributos que serán llave principal de búsqueda de posibles duplicados. 2. El **sistema** genera una tabla temporal que contiene todos los atributos de la tabla base y todos los registros que fueron encontrados como posibles duplicados. 3. El **usuario** realiza las acciones pertinentes sobre la nueva tabla.	
Flujo Alternativo	
3a. Si el usuario elimina uno, varios o todos los resultados de la nueva tabla 1. El **sistema** elimina también esos registros de la tabla base. 2. El **sistema** actualiza la tabla base. 3b. Si el usuario descarta uno, varios o todos los resultados de la nueva tabla 1. El **sistema** mantiene esos registros en la tabla base. 2. El **sistema** actualiza la tabla base. 3c. Si el usuario modifica los registros de la nueva tabla. 1. El **sistema** actualiza la tabla base.	

Caso de Uso Duplicates Search

Caso de Uso	Duplicates Search
Flujo Básico	
1. El **usuario** selecciona los atributos que serán llave principal de búsqueda de duplicados. 2. El **sistema** genera una tabla temporal que contiene todos los atributos de la tabla base y todos los registros que fueron encontrados como duplicados. 3. El **usuario** realiza las acciones pertinentes sobre la nueva tabla.	
Flujo Alternativo	
3a. Si el usuario elimina uno, varios o todos los resultados de la nueva tabla 1. El **sistema** elimina también esos registros de la tabla base. 2. El **sistema** actualiza la tabla base. 3b. Si el usuario descarta uno, varios o todos los resultados de la nueva tabla 1. El **sistema** mantiene esos registros en la tabla base. 2. El **sistema** actualiza la tabla base. 3c. Si el usuario modifica los registros de la nueva tabla. 1. El **sistema** actualiza la tabla base.	

Caso de Uso Table Details

Caso de Uso	Table Details
Flujo Básico	
1. El **usuario** selecciona los atributos de los cuales desea ver información. 2. El **sistema** genera una tabla temporal con la información de los mismos.	

Caso de Uso Table Union

Caso de Uso	Table Union
Flujo Básico	
1. El **usuario** selecciona dos tablas, que serán utilizadas para realizar la unión. 2. El **usuario** selecciona los atributos de las dos tablas que serán unidos. 3. El **sistema** solicita al usuario un nombre para la tabla resultado de la unión. 4. El **sistema** crea la tabla.	
Flujo Alternativo	
2a. Si el número de atributos seleccionados para las dos tablas no es el mismo. 1. El **sistema** muestra un mensaje de error informando de esta situación.	

CASOS DE USO ADMINISTRACION

Caso de Uso Login.

Caso de Uso	Login
Actor Principal	Usuario
Otros Actores	Sistema, Sistema Gestor de Bases de Datos (SGBD).
Precondiciones	El **usuario** no debe estar logueado en el **sistema**.
Poscondiciones	El **usuario** ingresa satisfactoriamente al **sistema**.
Flujo Básico	
1. El **usuario** ingresa su nombre de usuario. 2. El **usuario** ingresa su respectiva contraseña. 3. El **sistema** valida la información del **usuario** y lo registra como usuario logueado. 4. El **sistema** muestra la interfaz de trabajo.	
Flujo Alternativo	
3a. Los datos suministrados por el **usuario** no son correctos. 1. El **sistema** muestra un mensaje de error. 3b. El **usuario** que intenta ingresar ya se encuentra logueado en el sistema. 1. El **sistema** muestra un mensaje de error.	
Nota	
El **usuario** debe hacer parte del registro del sistema.	

Caso de Uso Logout.

Caso de Uso	Logout
Actor Principal	Usuario
Otros Actores	Sistema, Sistema Gestor de Base de Datos (SGBD).
Precondiciones	El **usuario** debe estar logueado en el **sistema**.
Poscondiciones	El **usuario** sale del **sistema**.
Flujo Básico	
1. El **sistema** registra al **usuario** como no logueado. 2. El **sistema** muestra la pantalla de login de la aplicación.	

7.2.3 Diagramas de casos de uso Entre las figuras 7.1 y 7.5 se muestran los Diagramas de Casos de Uso.

Figura 7.1 Diagrama de casos de uso de Acceso

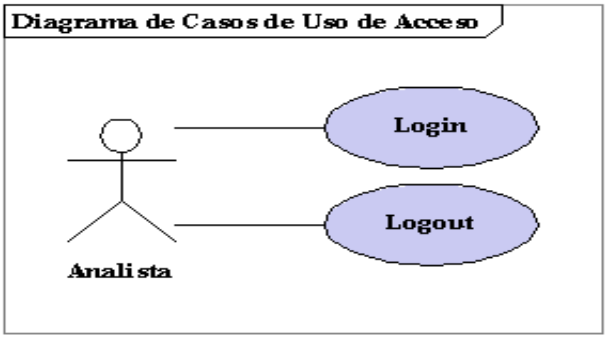


Figura 7.2 Diagrama de casos de uso Cleaning Procedures

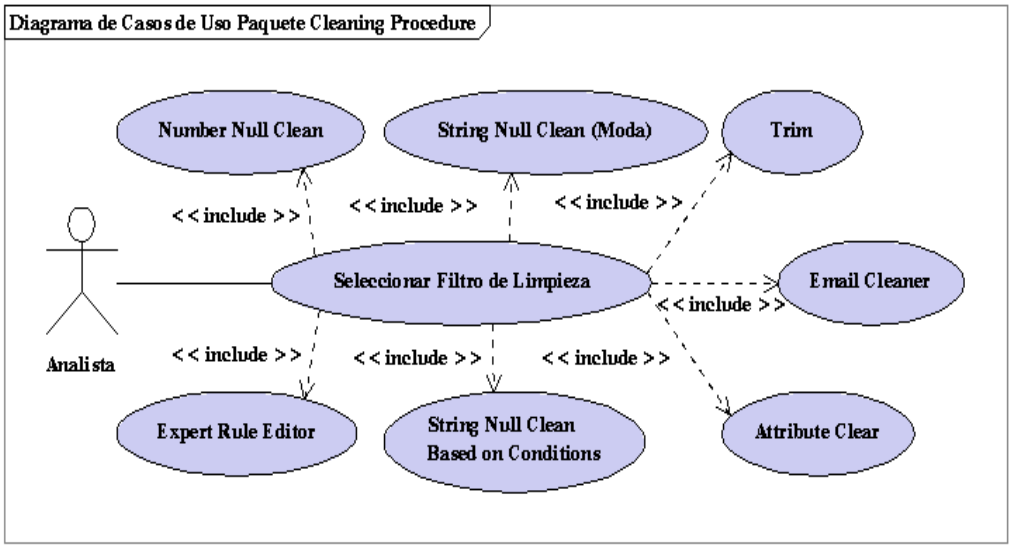


Figura 7.3 Diagrama de casos de uso Pre-Load

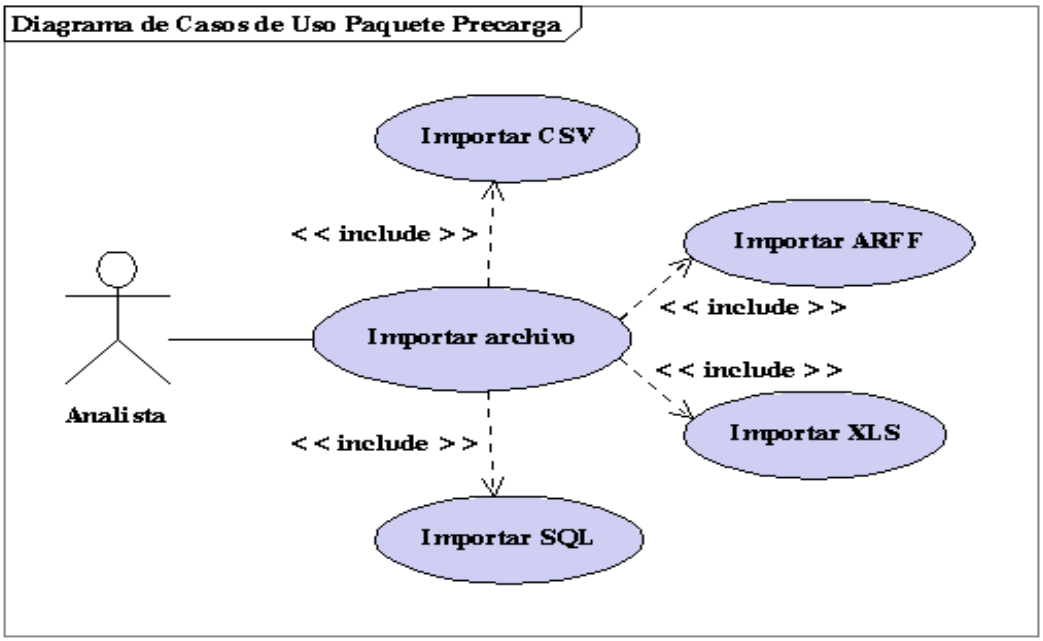


Figura 7.4 Diagrama de casos de uso Selection Filters

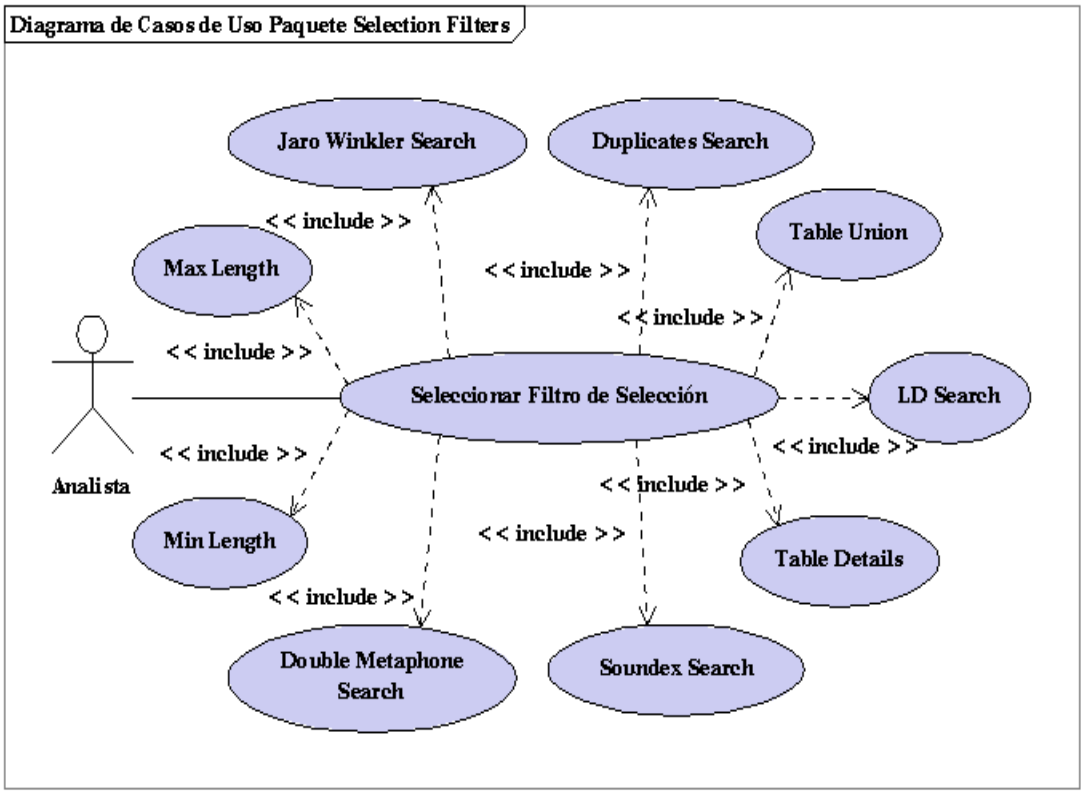
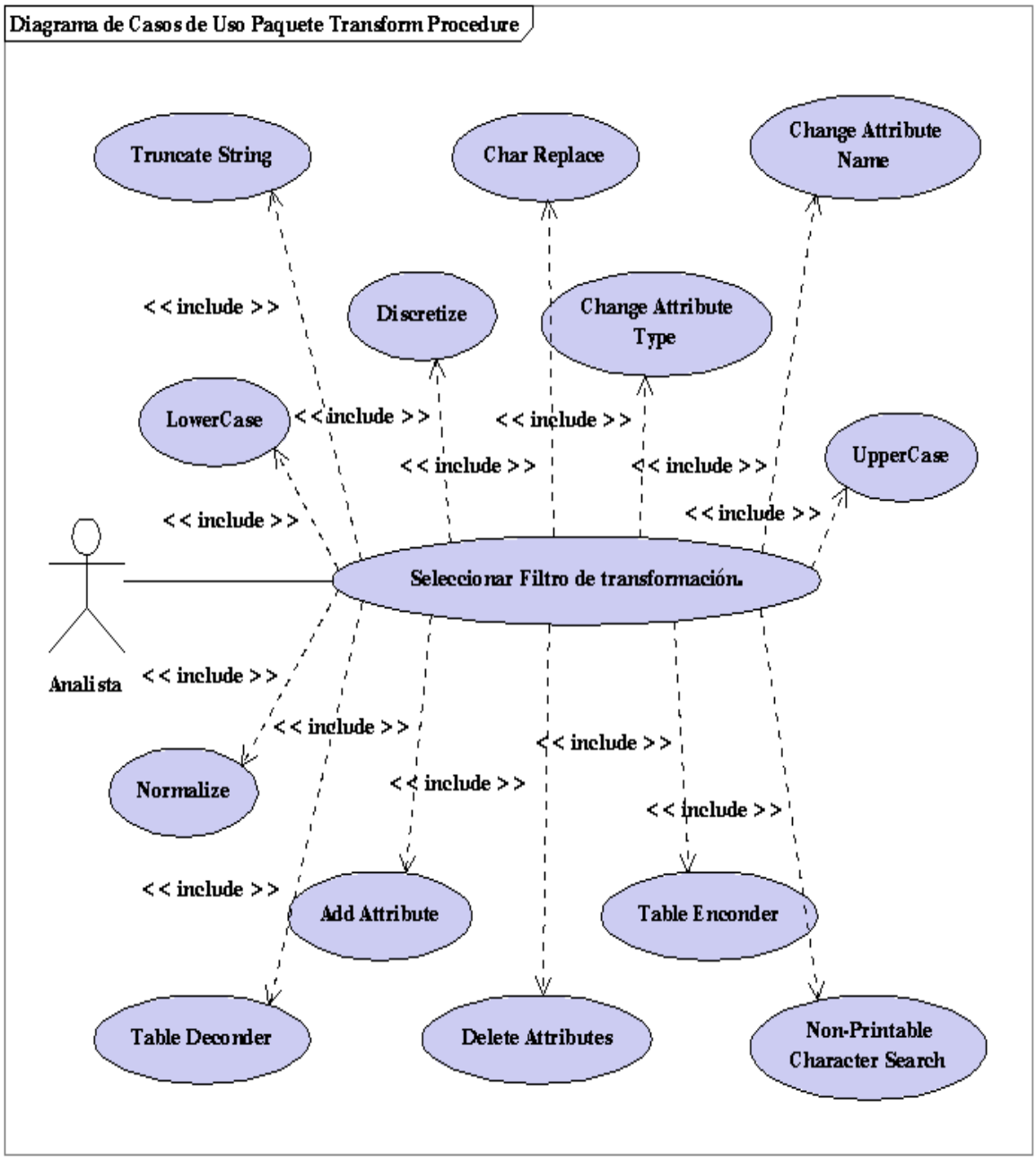


Figura 7.5 Diagrama de casos de uso Transform Procedures



7.2.4 Diagramas de secuencia Entre las figuras 7.6 y 7.13 se muestran los Diagramas de Secuencia del Sistema (DSS).

Figura 7.6 DSS Double Metaphone

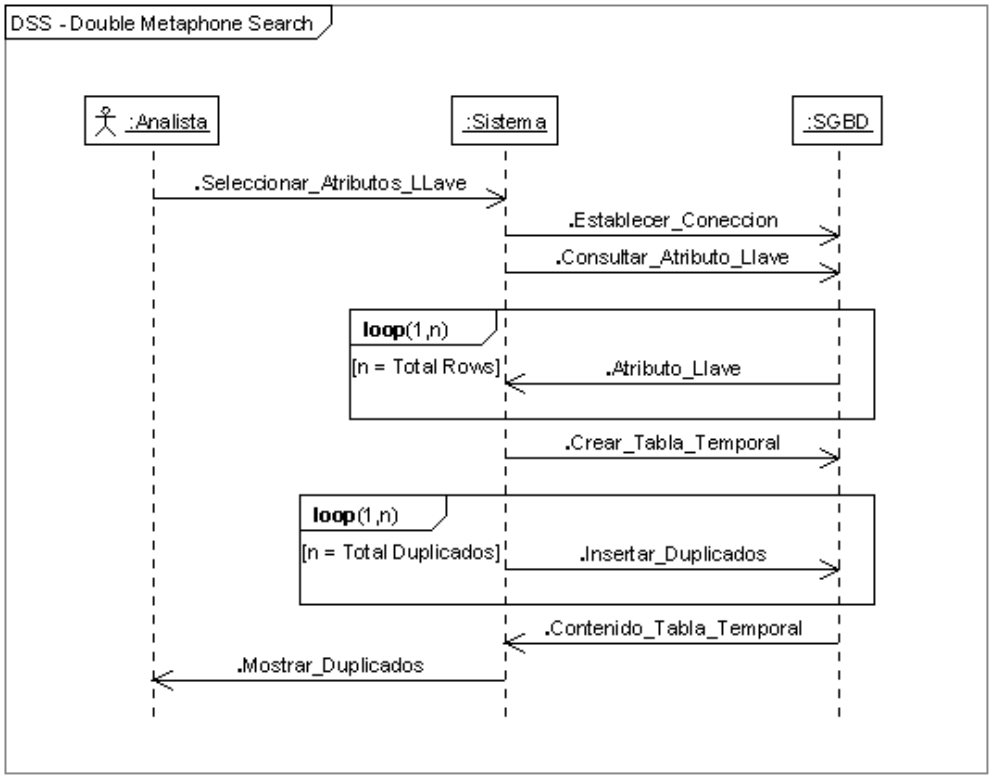


Figura 7.7 DSS Duplicates Search

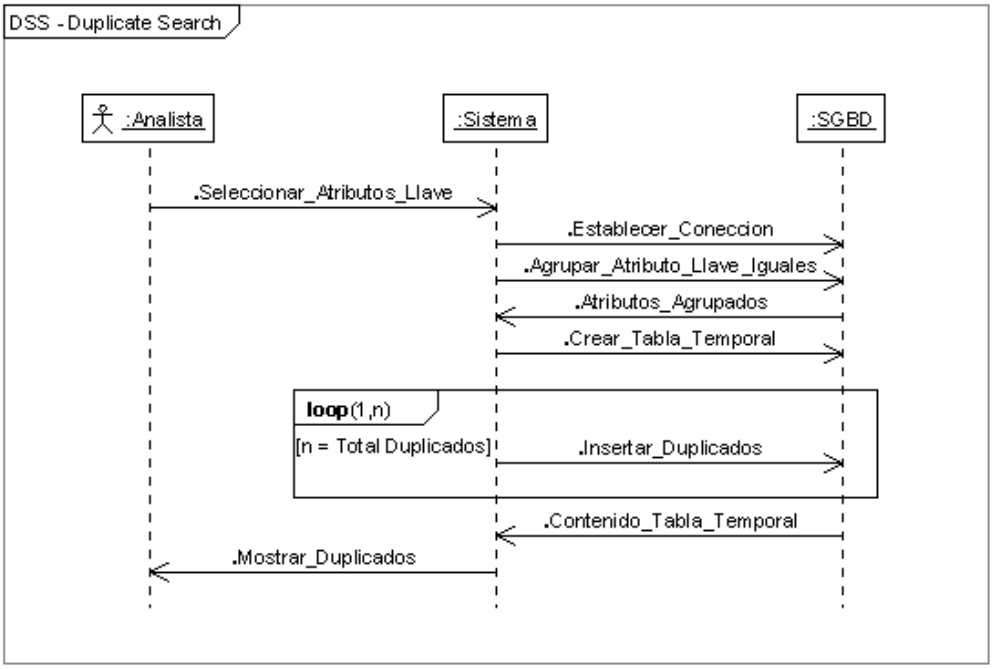


Figura 7.8 DSS Levenshtein Distance

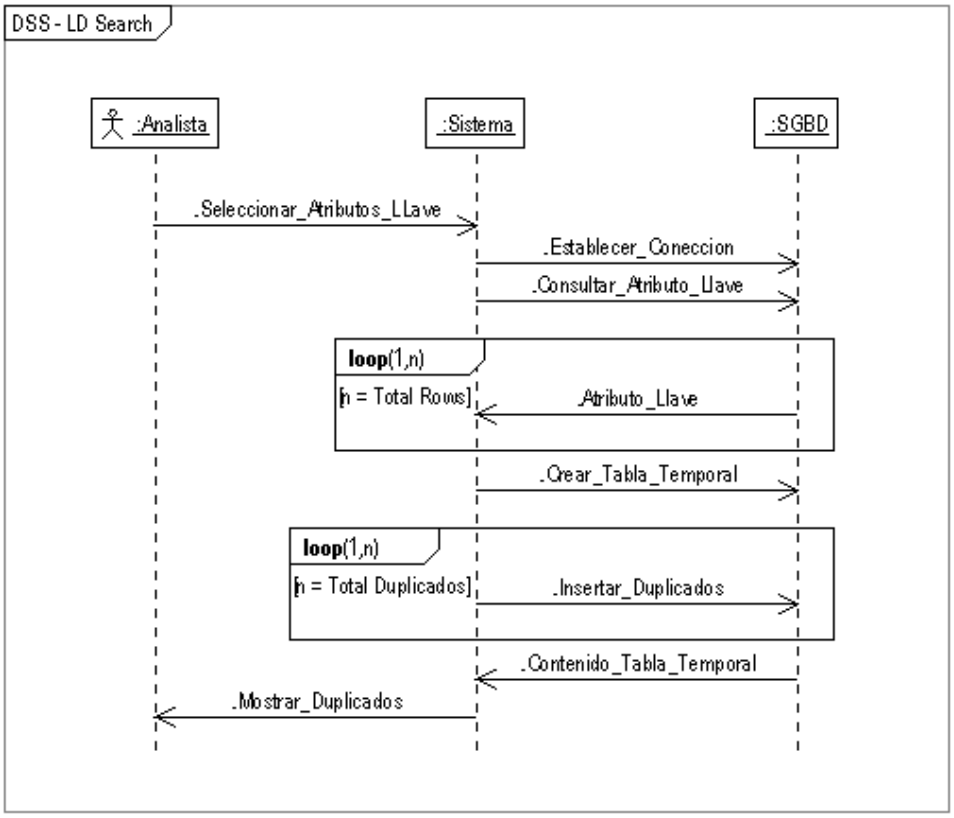


Figura 7.9 DSS Soundex

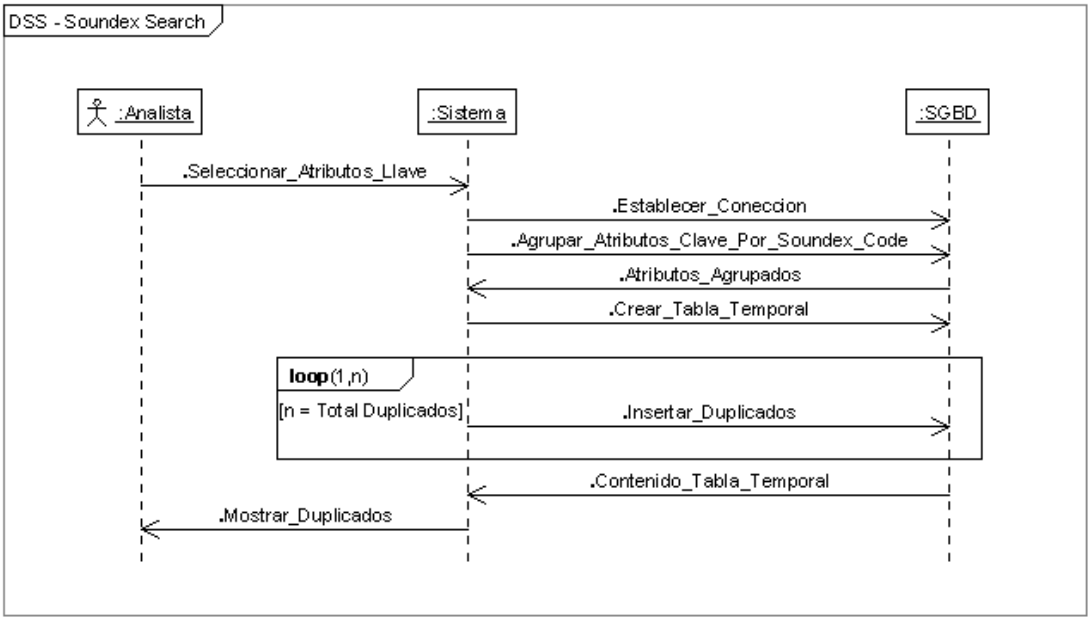


Figura 7.10 DSS Jaro-Winkler

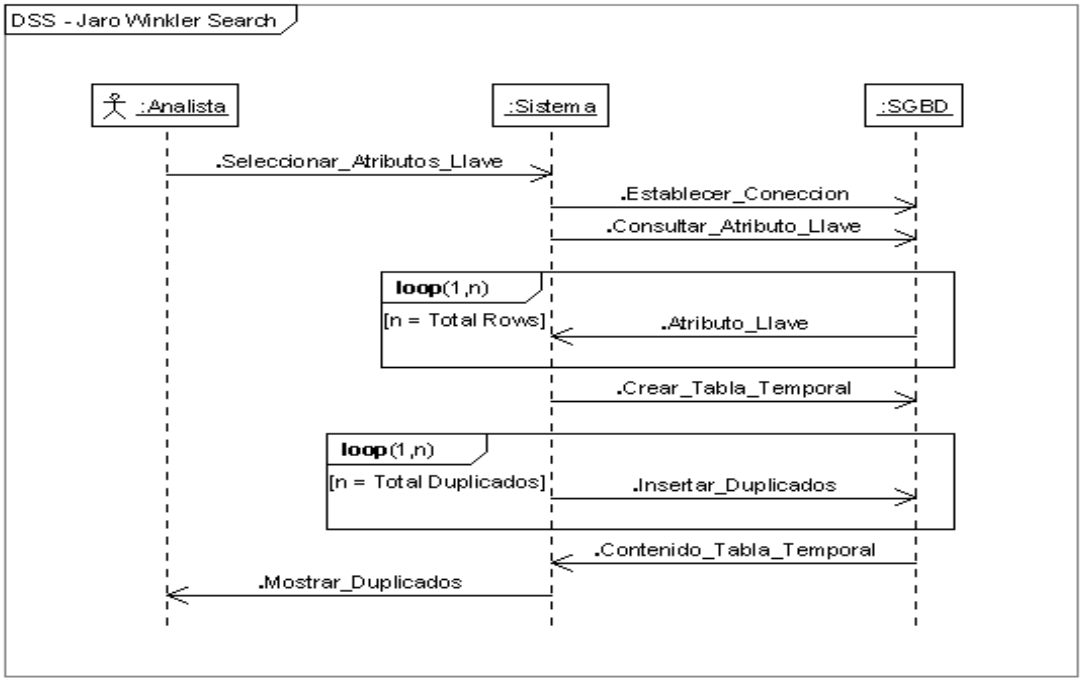


Figura 7.11 DSS Change Attribute Type

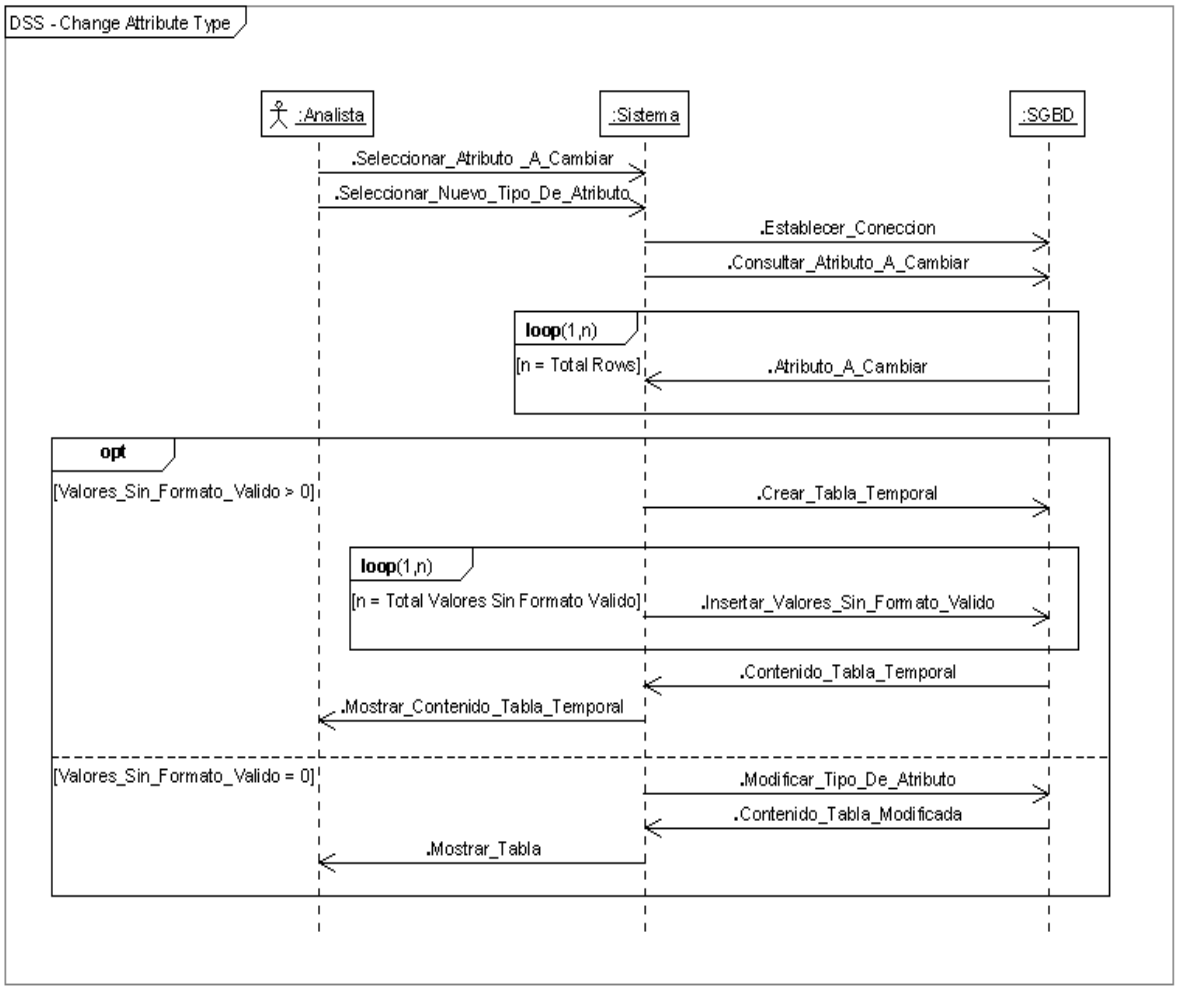


Figura 7.12 DSS Email Cleaner

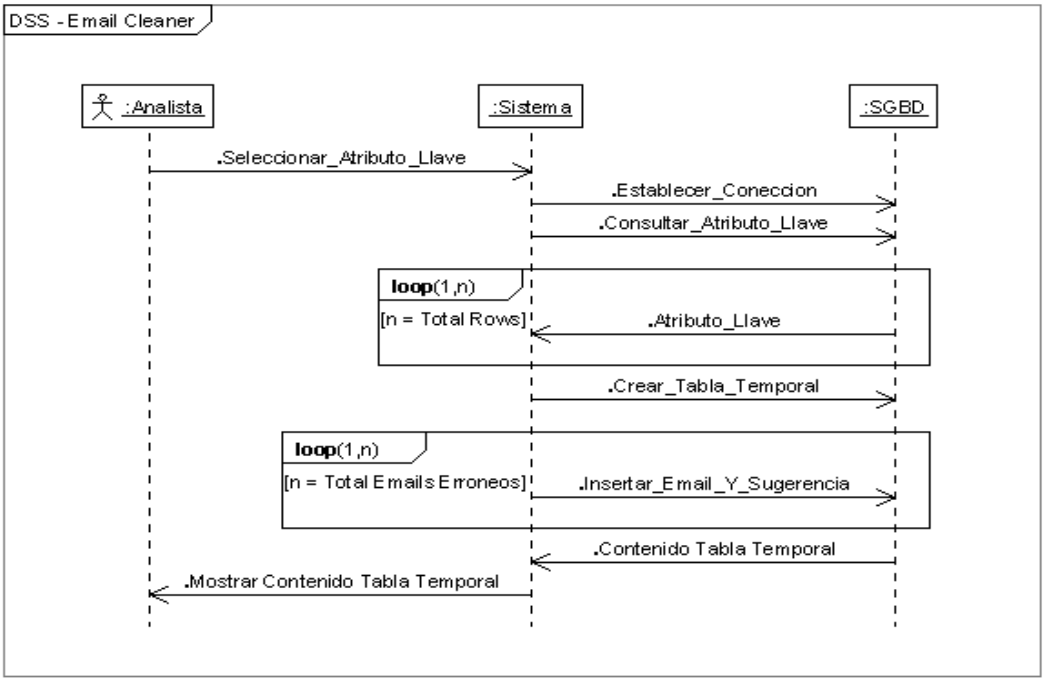
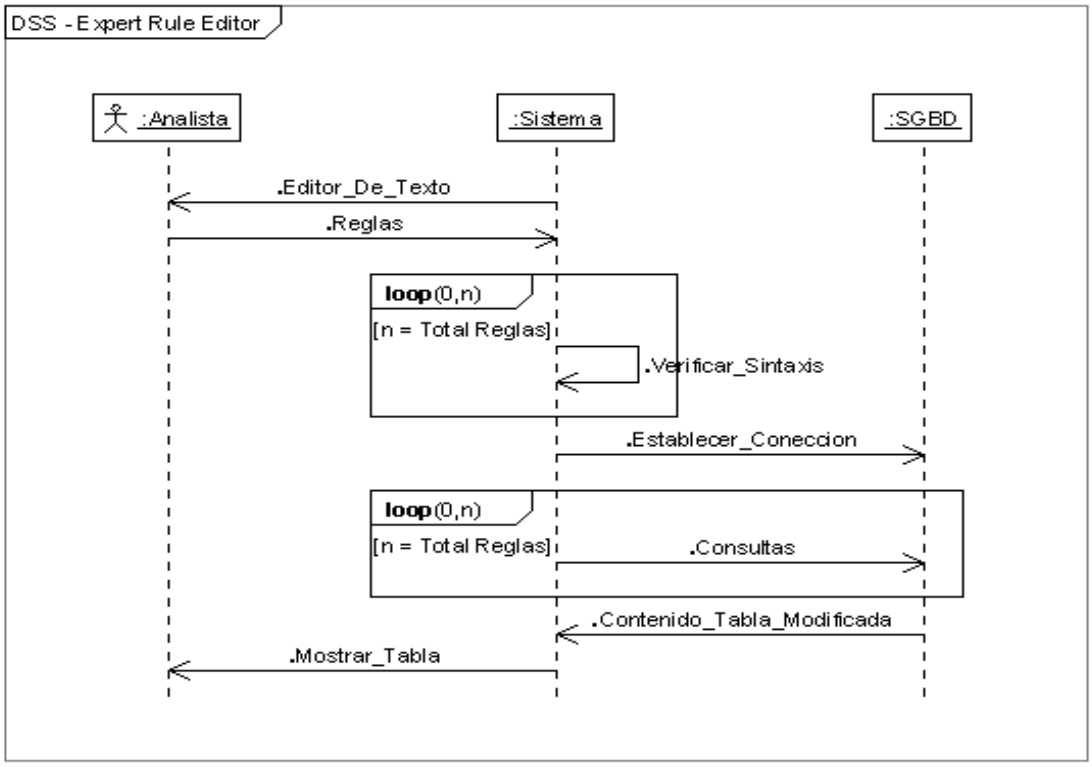


Figura 7.13 DSS Expert Rule Editor



7.3 DISEÑO UML

7.3.1 Diagramas de clase Entre las figuras 7.14 y 7.22 se muestran los Diagramas de Secuencia del Sistema (DSS).

Figura 7.14 DC iniciar interfaz principal

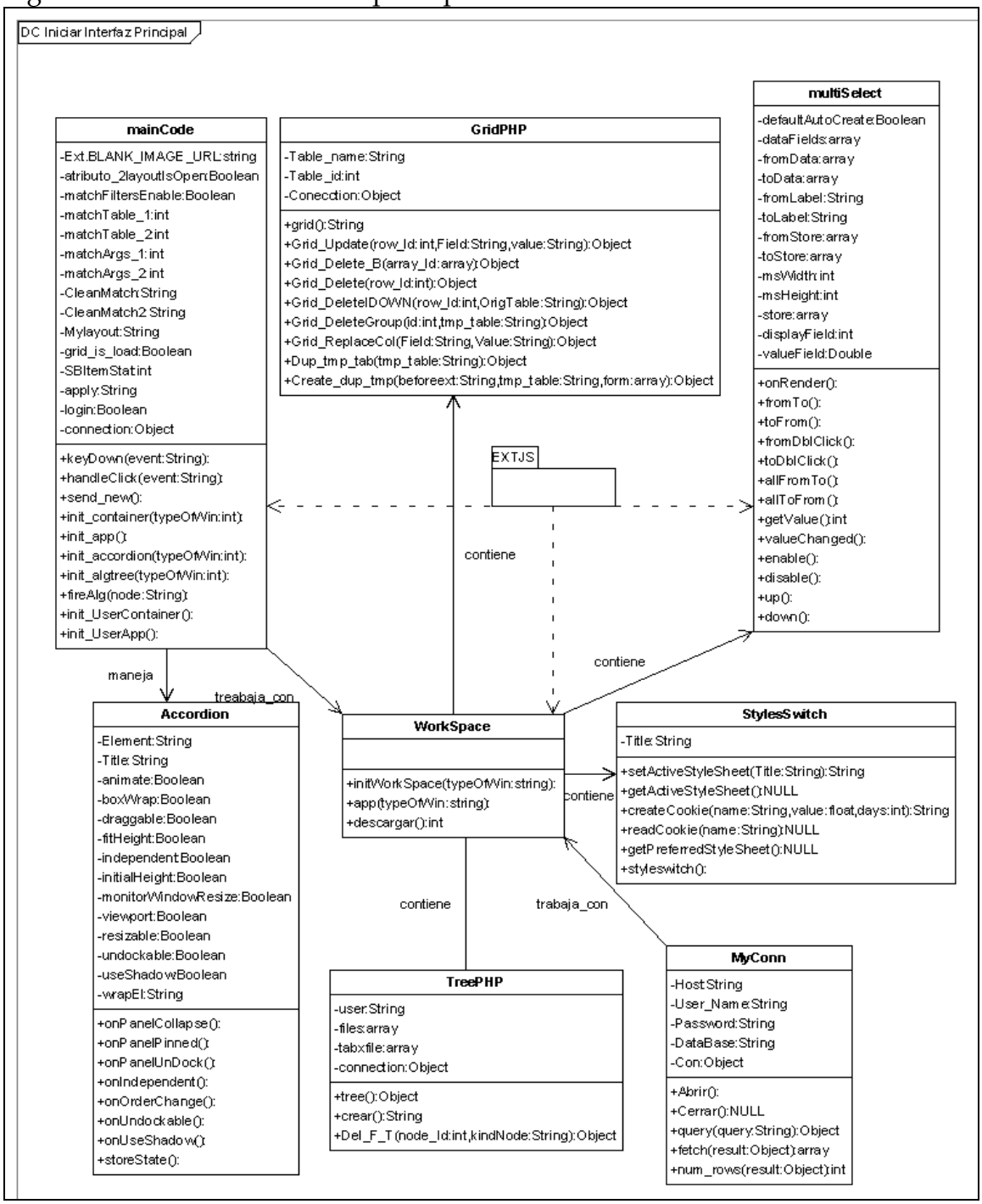


Figura 7.15 DC interfaz precarga, limpieza, selección y transformación

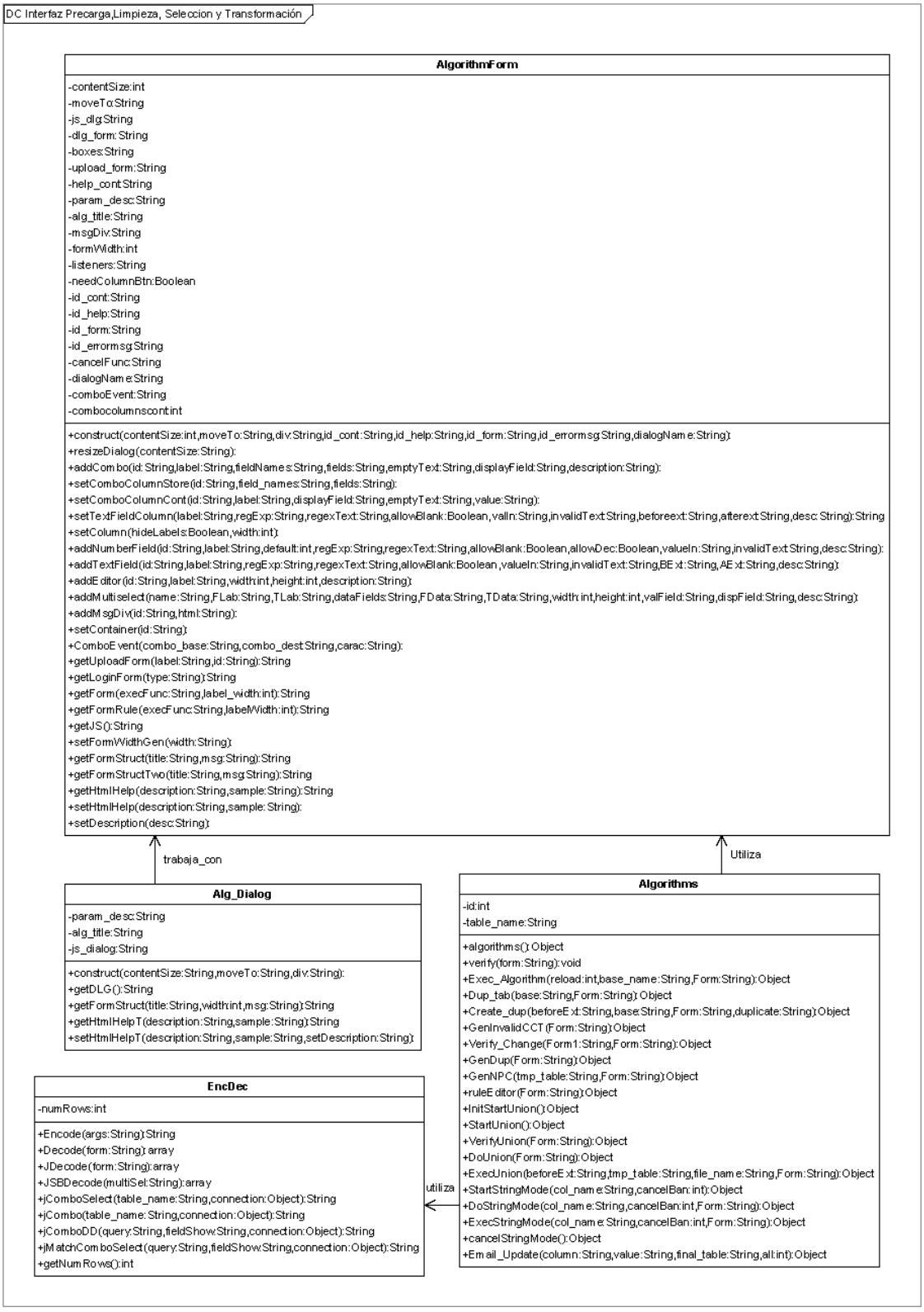


Figura 7.16 DC precarga de archivos

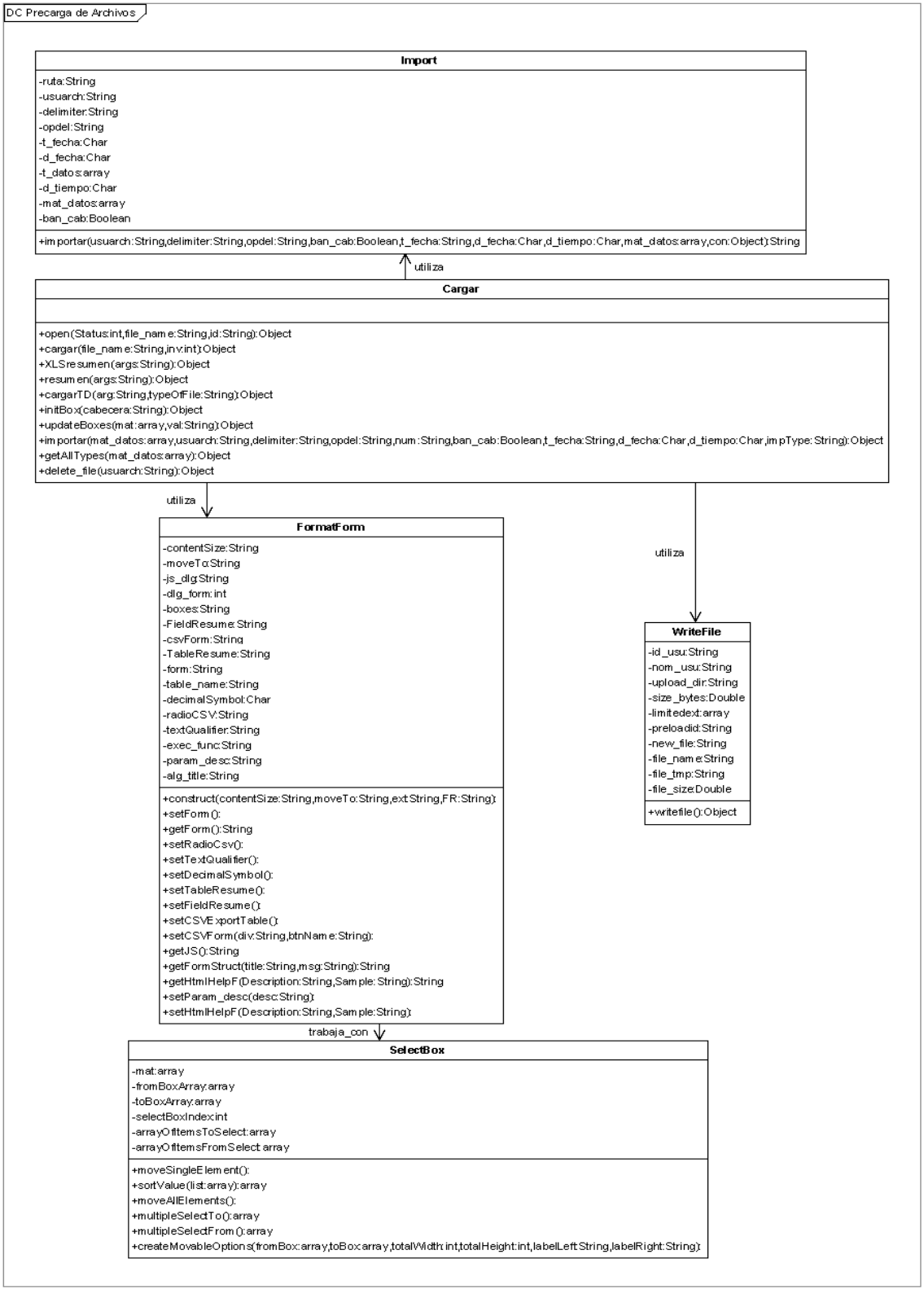


Figura 7.17 DC estadísticas

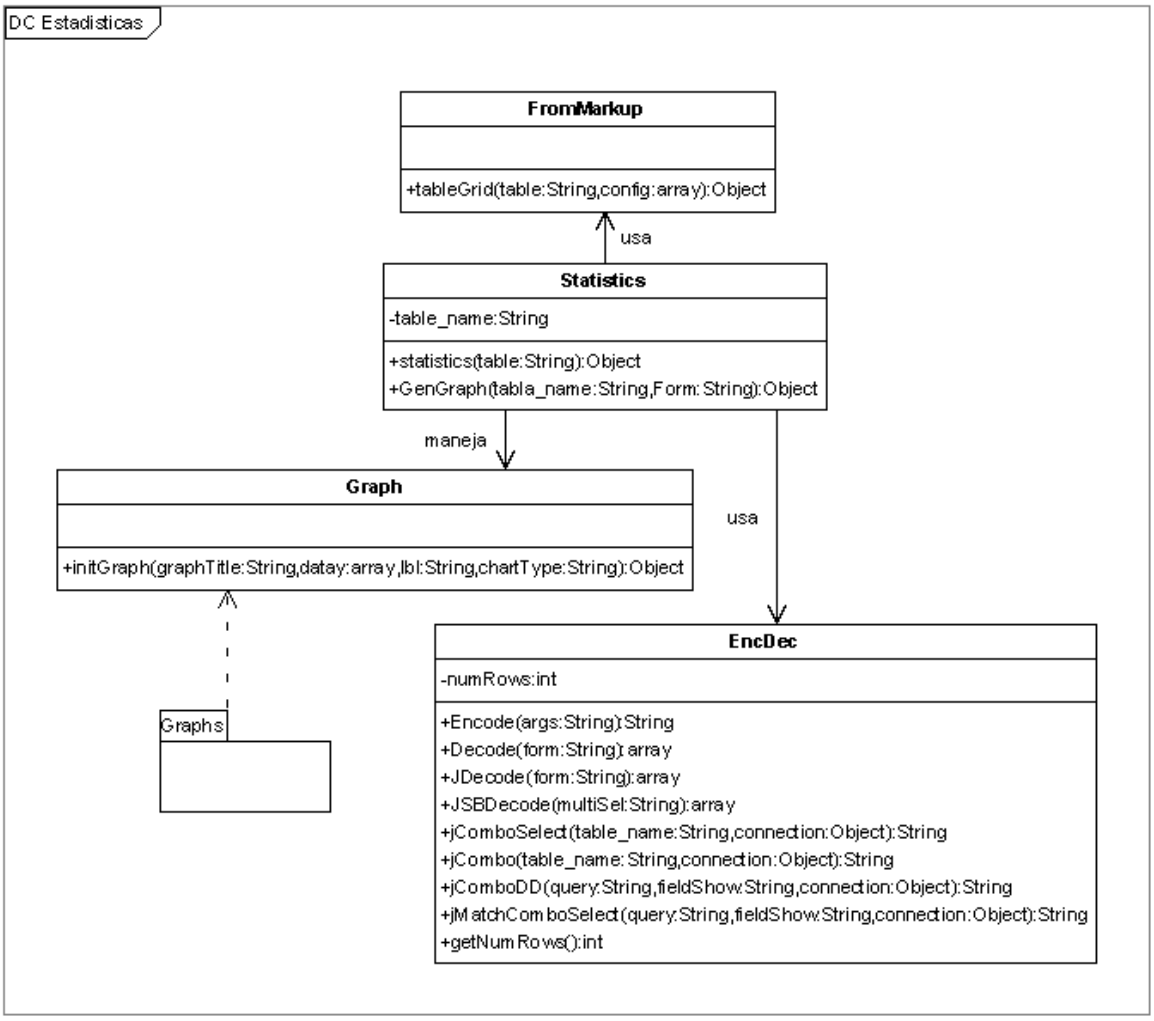


Figura 7.18 DC manejo de usuarios

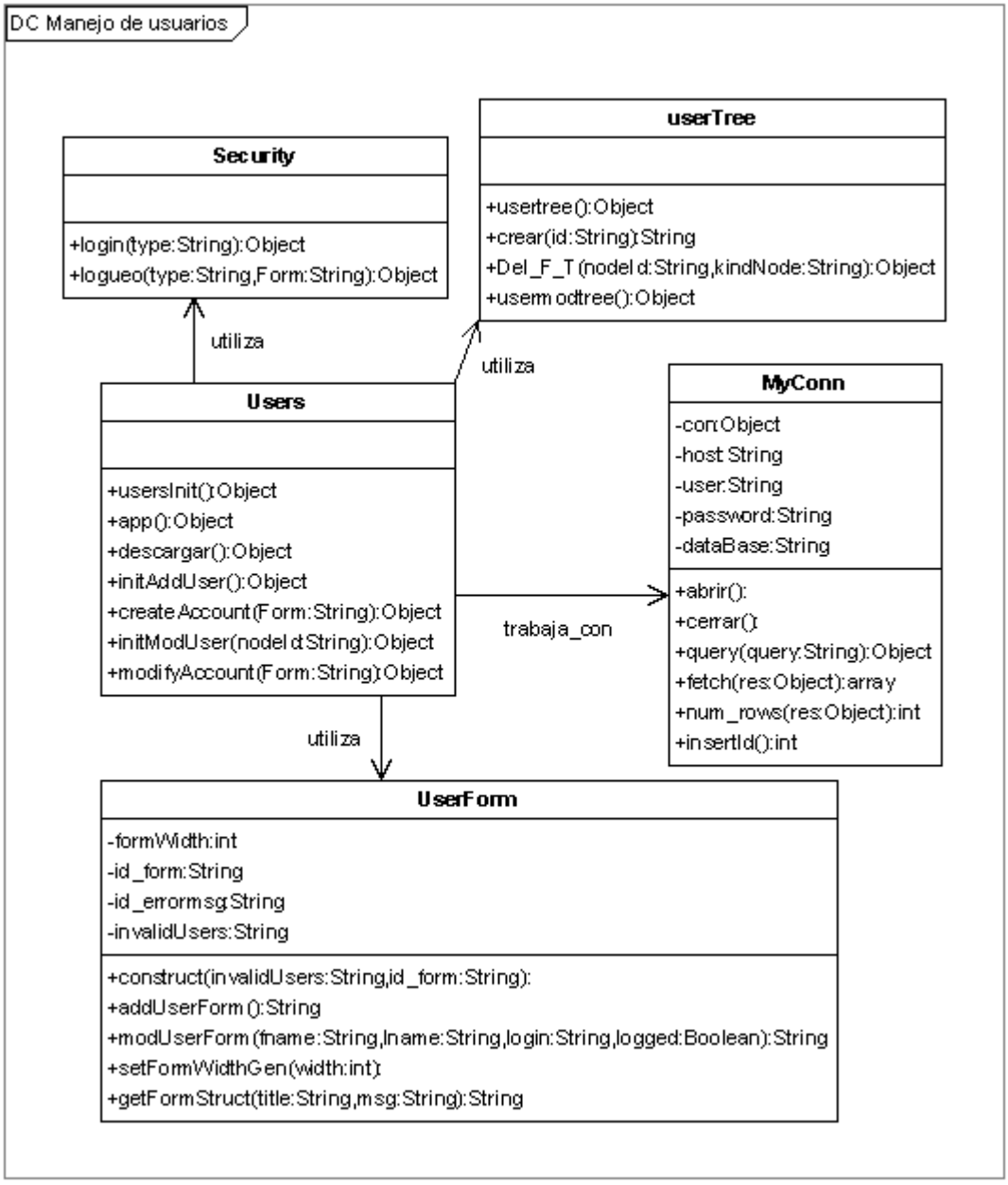


Figura 7.19 DC filtros de limpieza

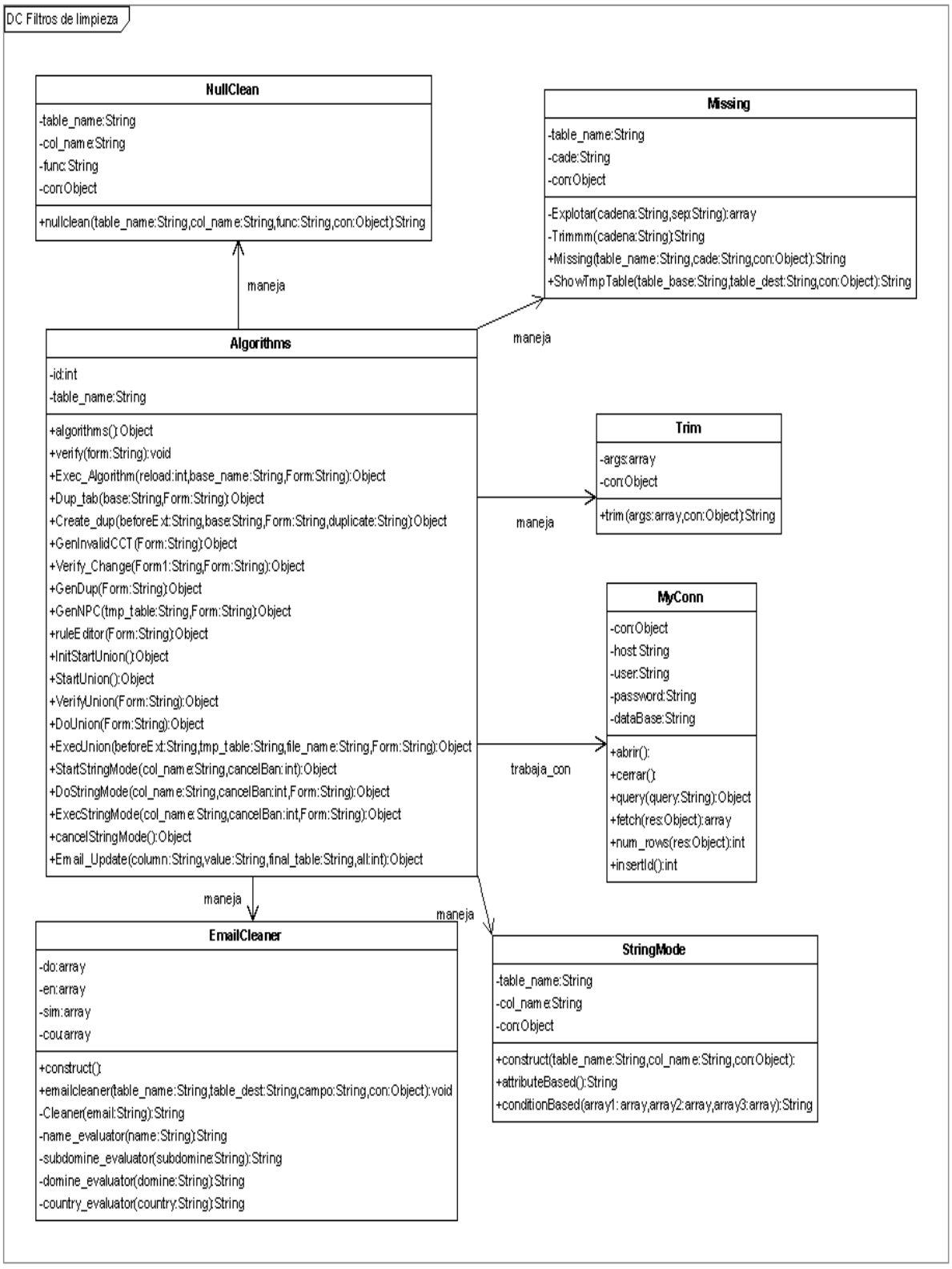


Figura 7.20 DC filtros de transformación

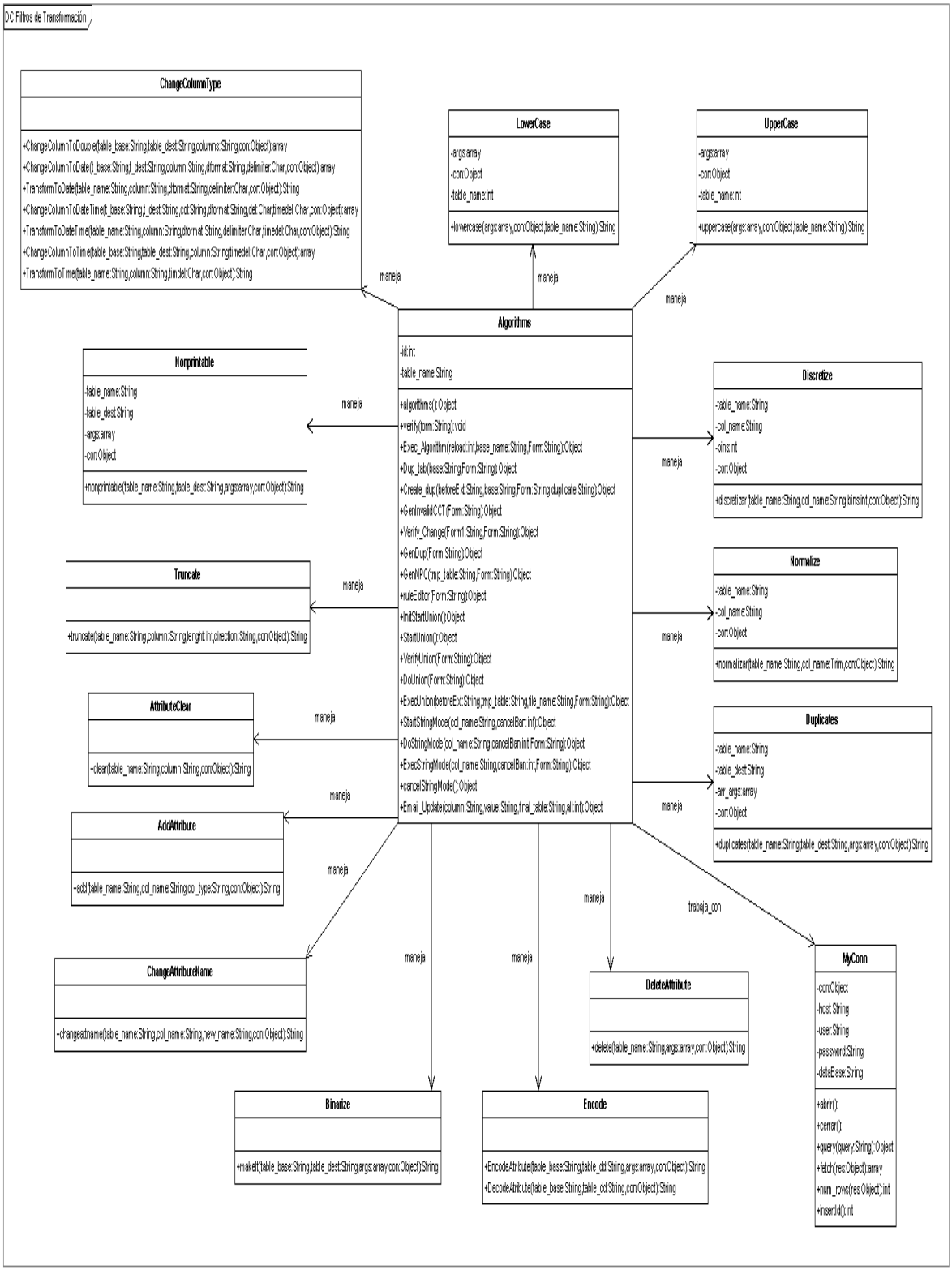


Figura 7.21 DC filtros de selección

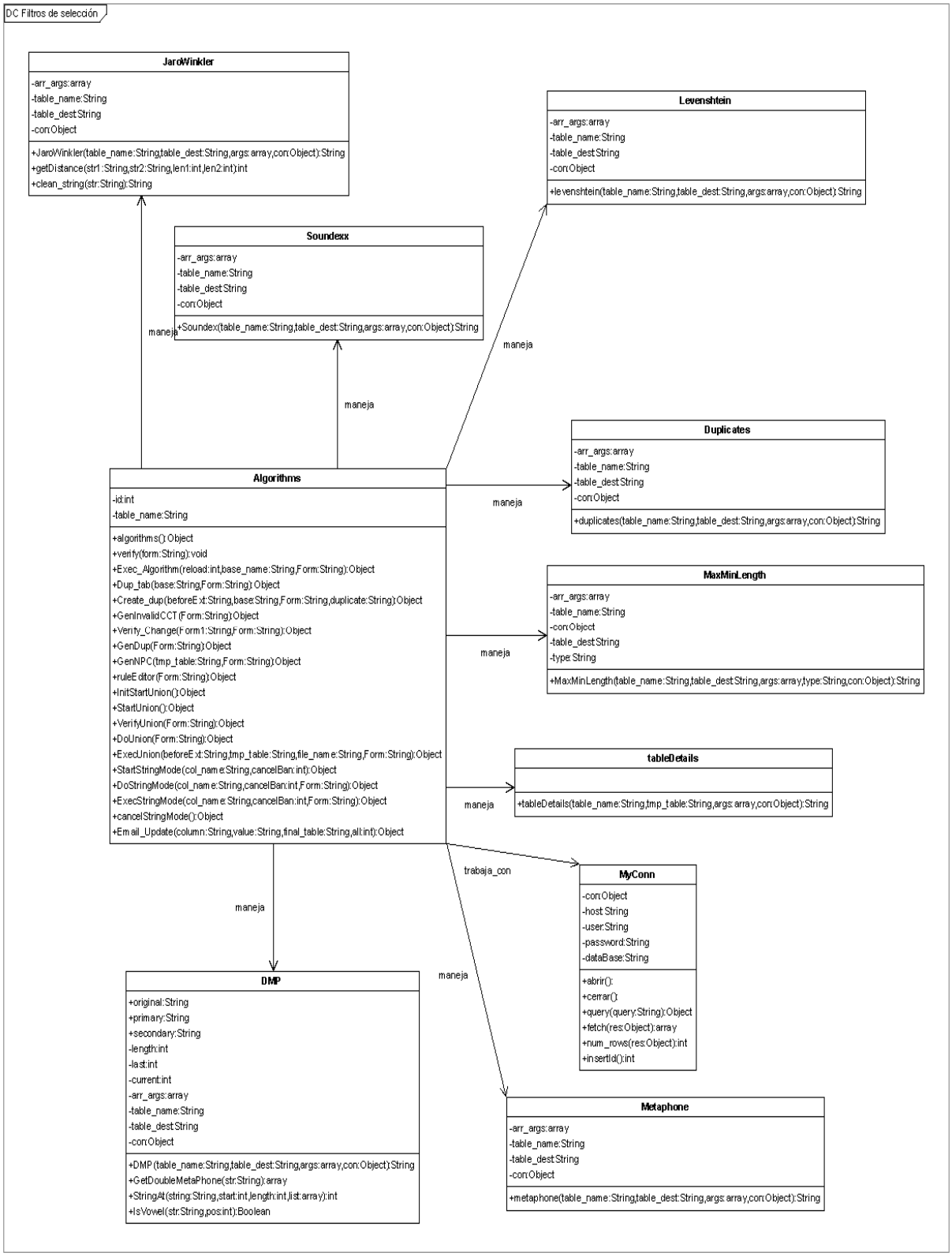
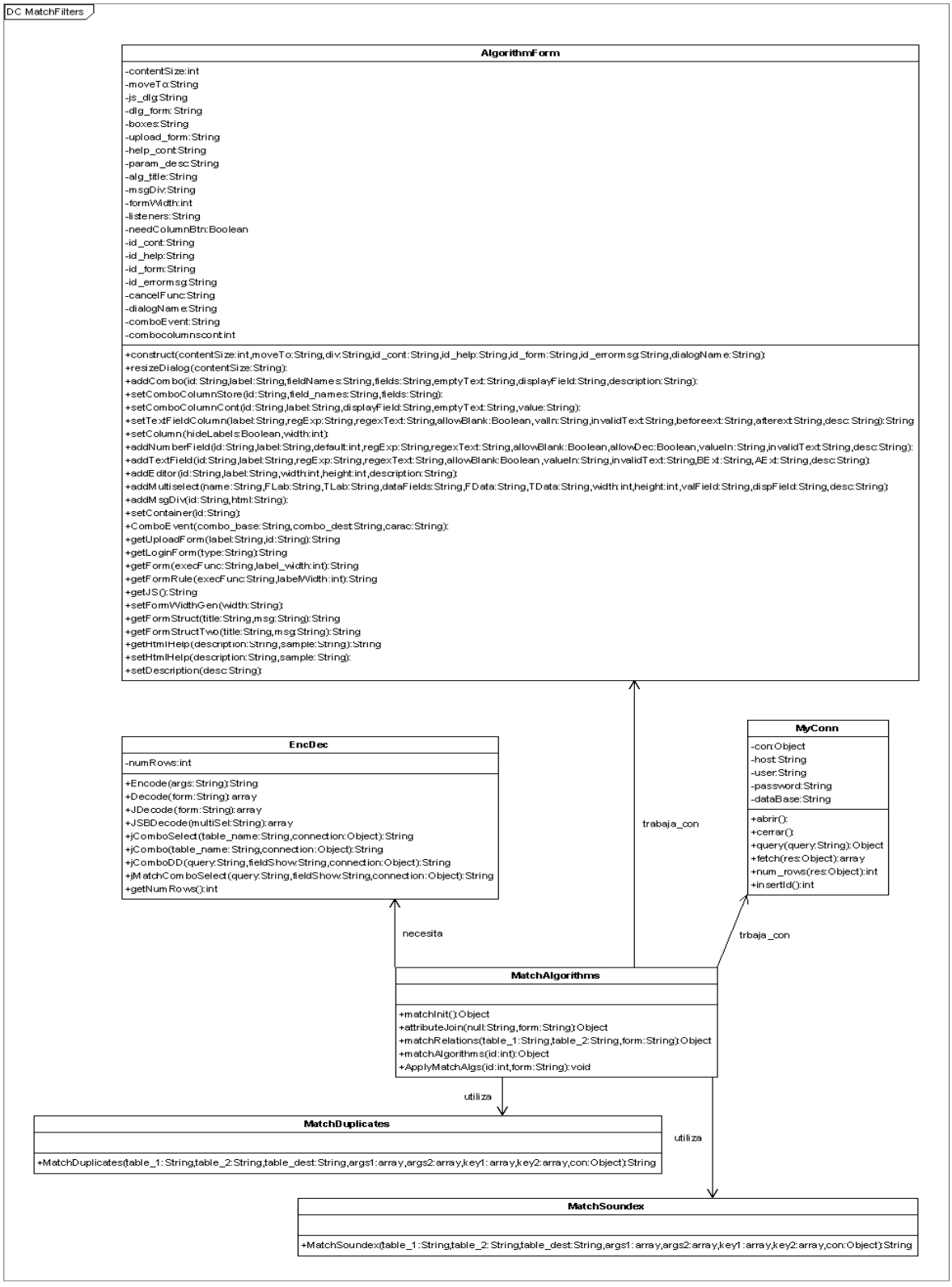


Figura 7.22 DC MatchFilters



7.3.2 Diagramas de paquetes Entre las figuras 7.23 Y 7.28 se muestran los diagramas de paquetes de EXDACLET.

Figura 7.23 Paquete principal EXDACLET

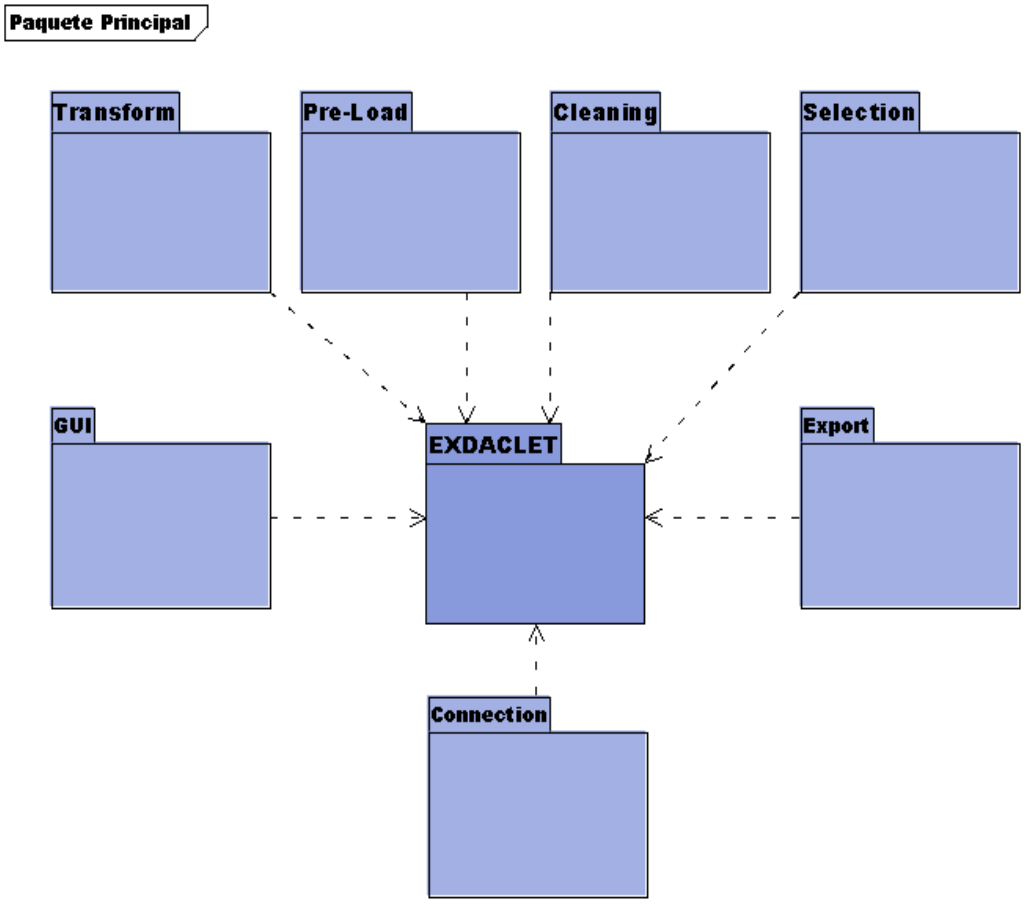


Figura 7.24 Paquete Pre-load

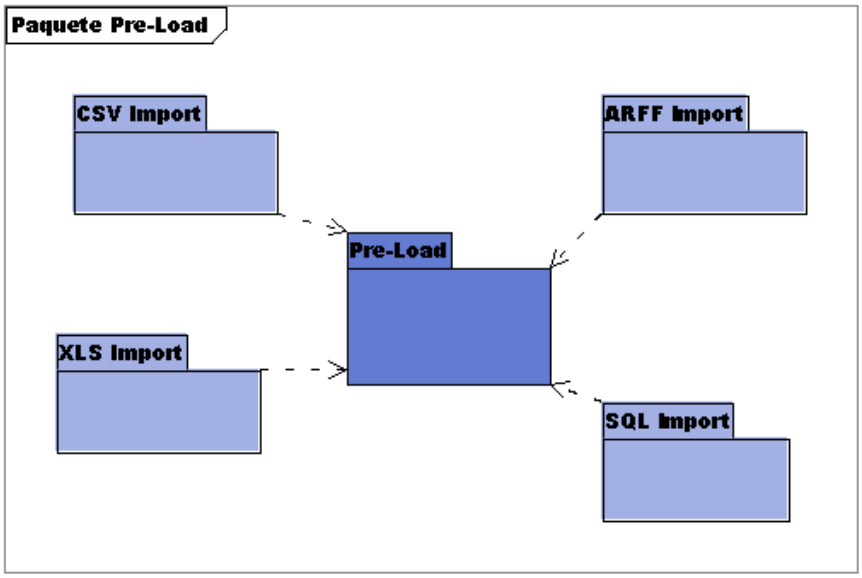


Figura 7.25 Paquete Selection

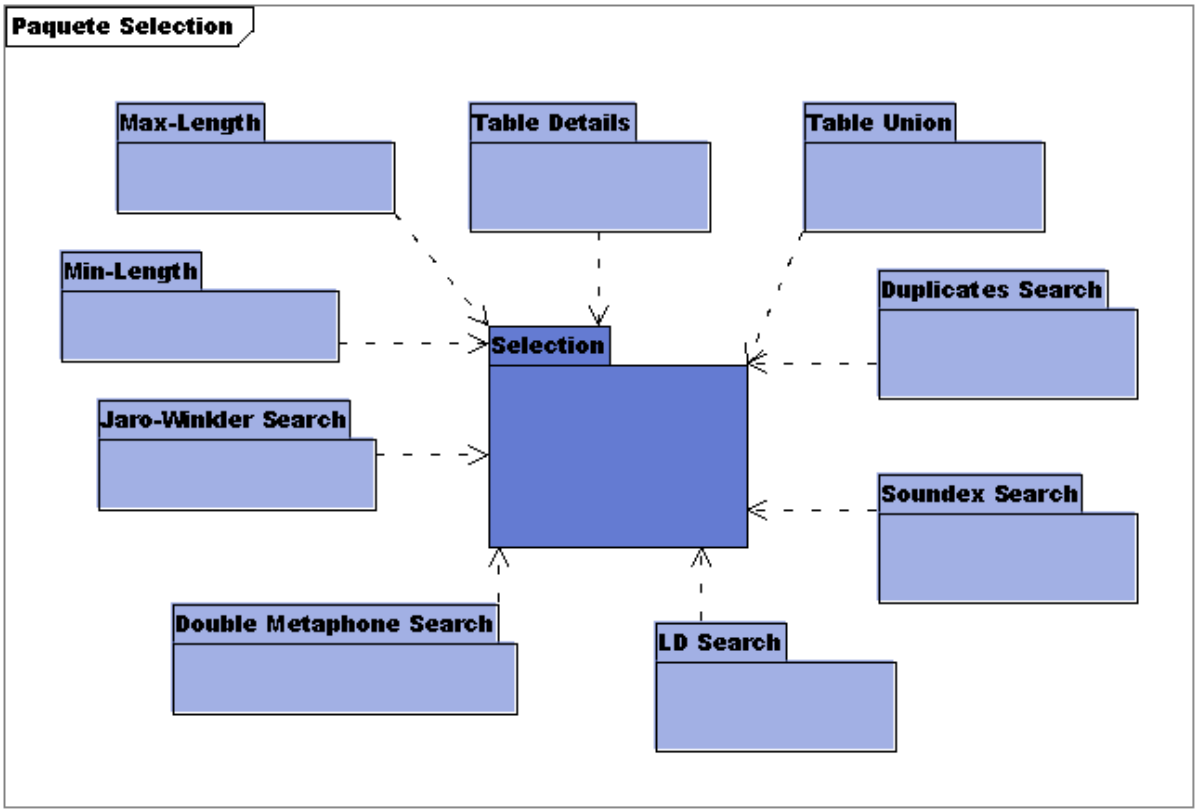


Figura 7.26 Paquete Transform

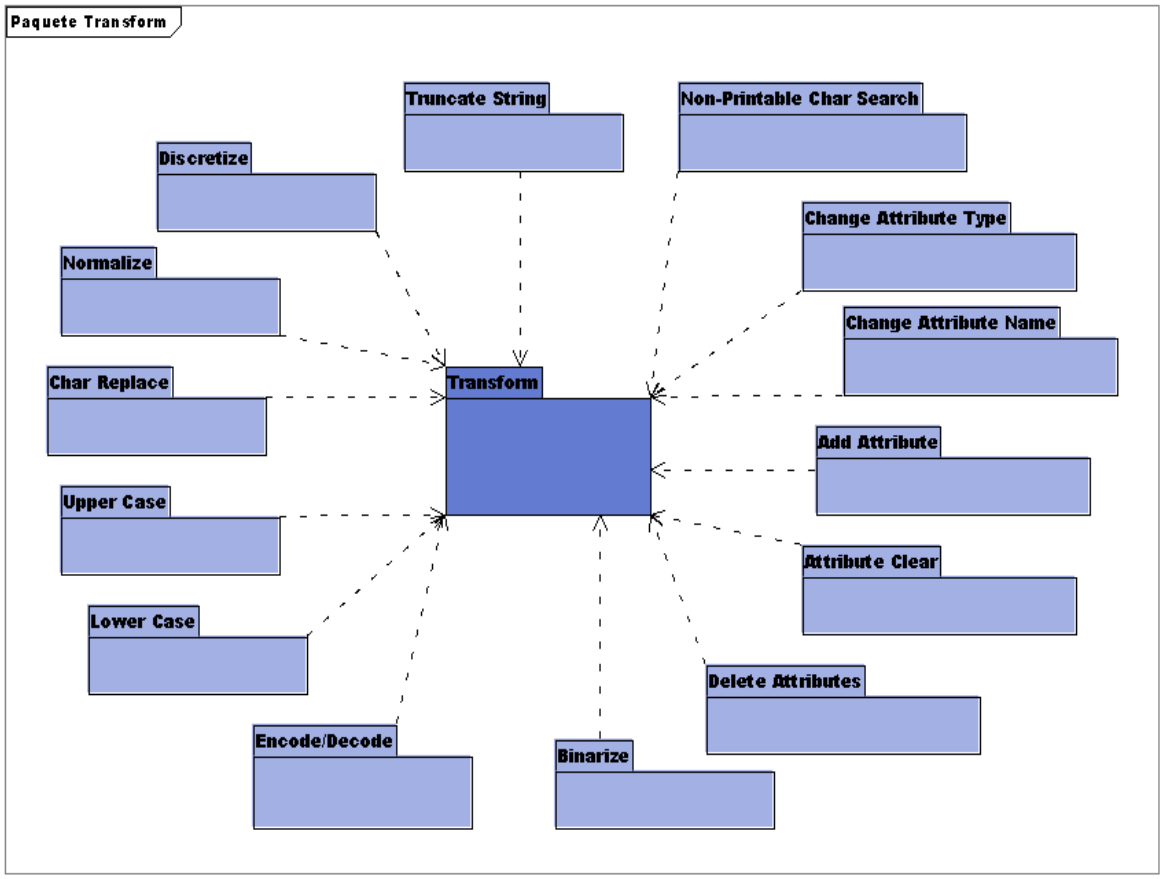


Figura 7.27 Paquete Cleaning

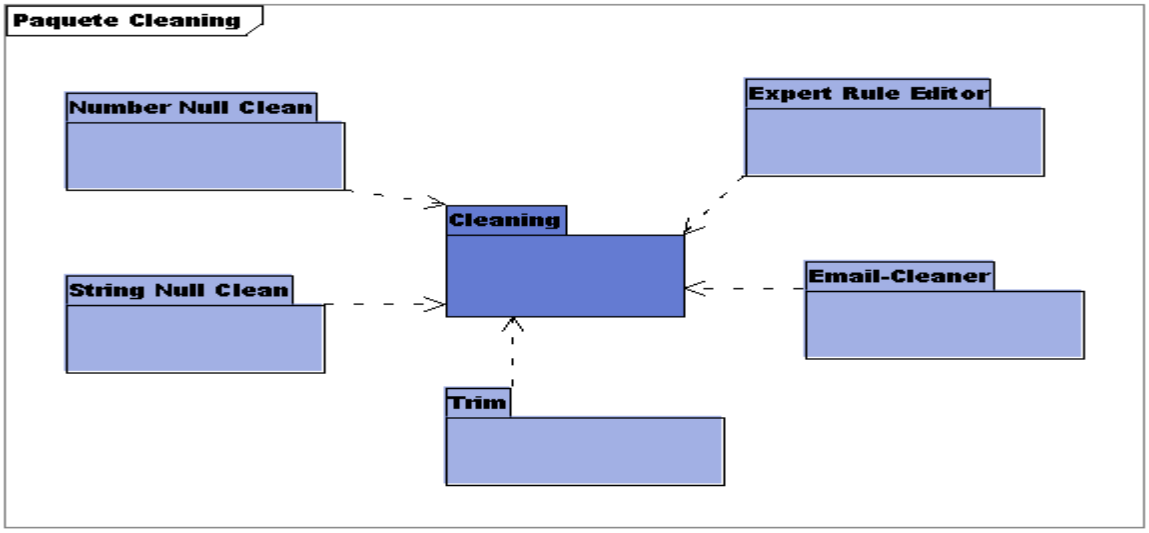
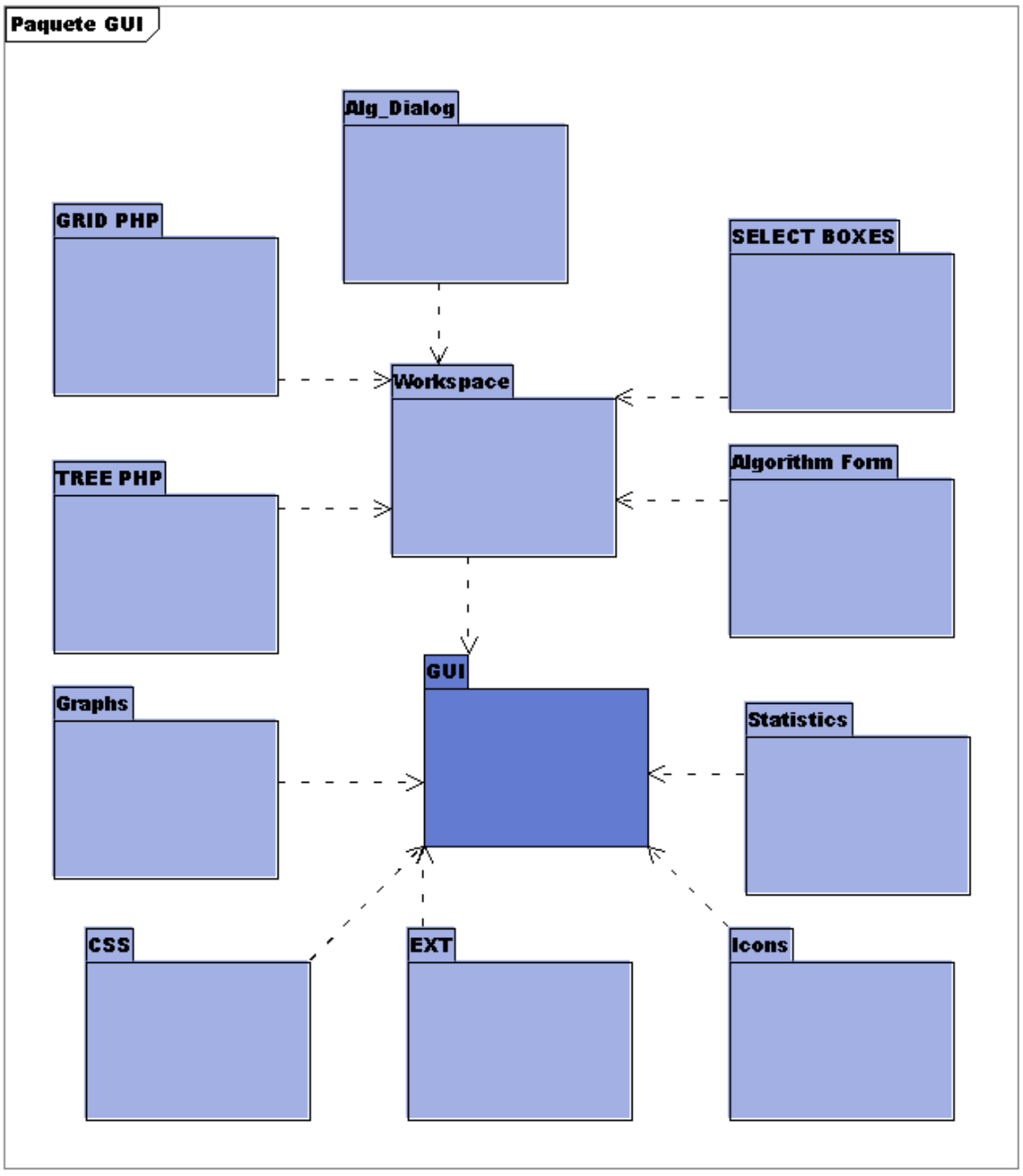


Figura 7.28 Paquete GUI



8. IMPLEMENTACION

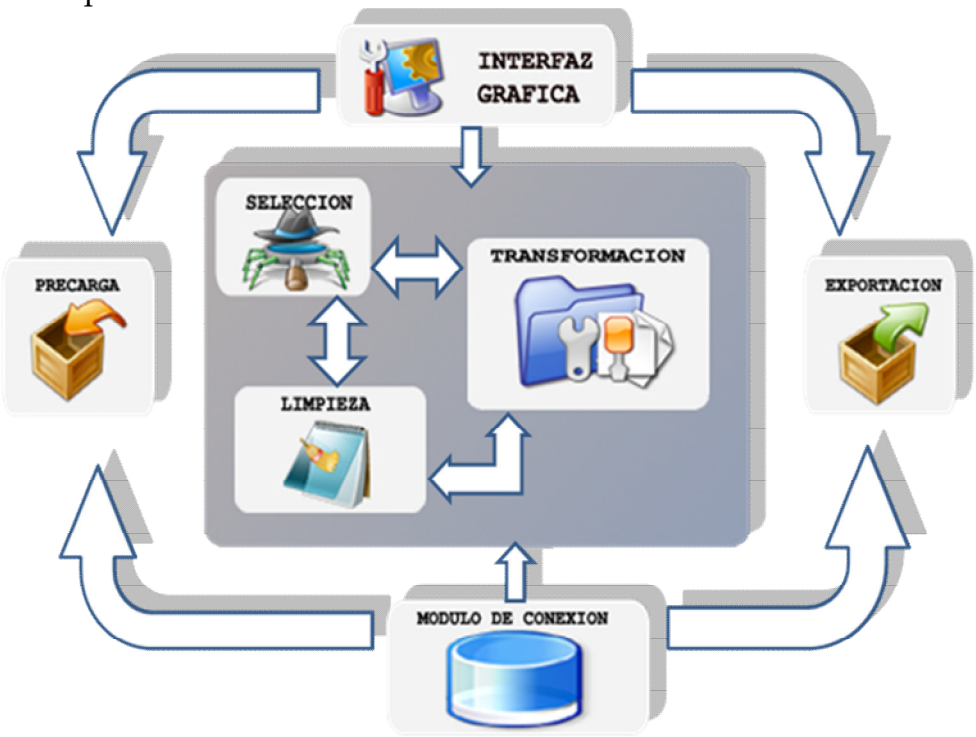
8.1 INTRODUCCION

Los sistemas operativos sobre los cuales se trabajó durante el desarrollo e implementación de la herramienta son Windows XP SP 2 y Fedora Core 8. Los lenguajes de programación en los cuales está elaborado EXDACLET son PHP 5.2.4, JavaScript 1.2, y HTML. EL sistema gestor de bases de datos en el cual se trabajó es MySQL 5.0.45. Teniendo en cuenta que es una herramienta orientada a la Web utiliza por lo tanto tecnologías AMP (Apache, MySQL, PHP) que permiten agrupar los lenguajes y el gestor de bases de datos anteriormente mencionados.

8.2 ARQUITECTURA DE EXDACLET

Los siguientes son los módulos de software que componen esta herramienta y su estructura se muestra en la figura 8.1.

Figura 8.1 Arquitectura de EXDACLET



8.2.1 Módulo de conexión Este módulo es el encargado de mantener una comunicación constante entre los filtros de precarga, limpieza, selección y transformación que ejecute el usuario en el sistema, con el Sistema Gestor de Bases de Datos que almacena los resultados de dichos filtros en tablas de datos, además tiene como objetivo la identificación de usuarios en base a atributos que ellos contienen (login y password), con el fin de que solo personas autorizadas tengan acceso a la manipulación de la herramienta.

8.2.2 Módulo de importación Este módulo es el encargado de transferir a la máquina servidor todos los archivos planos a los cuales el usuario les aplicará uno a más filtros. Su objetivo se centra en verificar la consistencia física de cada uno de los archivos que se quieran enviar al servidor y cargar en la herramienta.

8.2.3 Módulo del kernel de EXDACLET En este módulo se encuentran los paquetes fundamentales para la aplicación de todo el proceso de Limpieza de Datos. Contiene los módulos de Limpieza, Transformación y Selección.

El módulo de limpieza contiene todos los filtros necesarios para el tratamiento de datos nulos que se encuentren en las tablas de datos utilizadas por el usuario.

El módulo de transformación contiene todos los posibles procedimientos requeridos por el usuario para la transformación de los datos perdidos (missing) o fuera de rango (outliner) a datos reales y coherentes que puedan ayudar a posteriores procesos de análisis de datos, inteligencia de negocios o tareas de minería de datos.

El módulo de selección contiene filtros para realizar o aplicar búsquedas detalladas sobre conjuntos de datos con el fin de reducir la dimensionalidad de las tablas de datos que se utilicen y al igual que el módulo de transformación buscar que esos datos sean coherentes y más acordes a la realidad.

8.2.4 Módulo de exportación Este módulo es el encargado de convertir las tablas de datos que reposan en el sistema por medio del Sistema Gestor de Bases de Datos a todos los formatos de archivo plano contemplados por la herramienta,

todo esto con el fin de mantener una copia física de los resultados aplicados durante todo el proceso de Limpieza de Datos.

8.2.5 Módulo de interfaz gráfica Este módulo da soporte visual a todos los demás módulos y se encarga de brindar al usuario una experiencia muy amigable durante la manipulación de la herramienta de modo tal que resulten sencillos todos los experimentos que el usuario desee realizar mediante la interacción con todos los componentes que abarca la herramienta.

8.3 ARQUITECTURA DE PAQUETES Y CLASES

A continuación se describe de manera general la estructura de paquetes de EXDACLET y las clases que pertenecen a cada paquete. Posteriormente, se amplia y se detalla la forma como fueron desarrolladas dichas clases.

*Paquete Pre-load (*Carga e importación de archivos): Dentro de este paquete se encuentran las clases necesarias para generar en la herramienta las tablas de datos en base a los archivos planos soportados.

Paquete Cleaning (Limpieza de datos): En este paquete se encuentran las clases que implementan los filtros de limpieza esenciales para una tabla de datos.

Paquete Transform (Transformación de datos): En este paquete se encuentran las clases que implementan todos los procedimientos de transformación necesarios para garantizar la integridad y veracidad de los datos, incluidos en las tablas manejadas por los usuarios.

Paquete Selection (Selección de datos): En este paquete se encuentran las clases que permiten la ejecución de algoritmos para el reconocimiento de patrones en palabras (Búsqueda de: Duplicados, Homónimos, etc.).

Paquete GUI (Interfaz gráfica): Contiene todas las clases que implementan la interfaz gráfica de la herramienta.

8.3.1 Paquete Pre-Load La clase que compone este paquete es la siguiente:

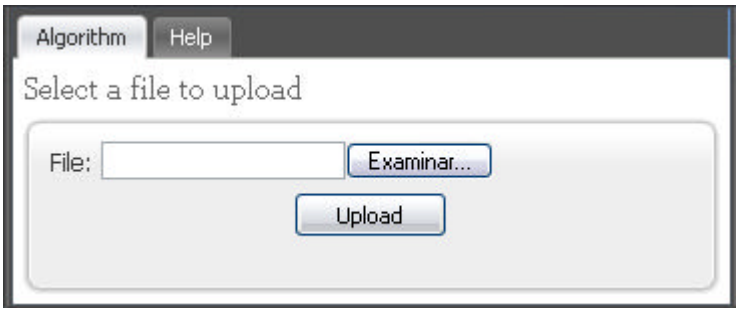
Clase Cargar: Esta clase permite establecer si los archivos que el usuario intenta cargar, cumplen con alguno de los formatos soportados por EXDACLET, una vez verificado e identificado dicho formato, si el archivo es XLS o CSV, entonces ejecuta le método **XLSresumen** o **CSVresumen** respectivamente para determinar la estructura que compone al archivo que se está importando, una vez definida su estructura entonces ejecutan los métodos **Importar CSV** o **Importar XLS;** de otra forma si el archivo es ARFF o SQL, entonces pasa directamente a la verificación de la integridad del archivo mediante los métodos **ImportarARFF** o **ImportarSQL.**

Además de los métodos anteriormente mencionados se encuentran los siguientes métodos que ayudan al proceso de importación:

- **WriteFile:** Se encarga de verificar que el archivo que el usuario intenta cargar no existe físicamente en el espacio reservado en el servidor para la carga de archivos, también verifica la consistencia externa del archivo, es decir, se encarga de la forma cómo está escrito su nombre, la extensión que lo acompaña y el tamaño que tiene, si todo es correcto entonces el archivo se traslada de la máquina cliente al servidor.
- **CargarTD:** Si el archivo tiene la extensión XLS o CSV este método se ejecuta con el fin de mostrar un listado de tipos de datos disponibles para ser asignados a cada atributo de la tabla que se encuentra en el archivo.
- **DeleteFile:** este método se ejecuta cuando, a pesar de que el archivo esté físicamente bien, es decir, haya sido trasladado de la máquina cliente al servidor mediante el método **writeFile**, el usuario procede a la cancelación de la importación del mismo. En este caso, este método elimina el archivo que ha sido copiado en el servidor.

En la figura 8.2 se muestra la presentación estándar para la transferencia de archivos al servidor.

Figura 8.2 Ventana de selección de archivos



8.3.2 Paquete Cleaning Este paquete se compone de las siguientes clases:

Number Null Clean: Esta clase tiene como objetivo llenar todos los campos vacios de un atributo seleccionado por el analista en base a una serie de funciones estadísticas, por esta razón solamente es aplicable sobre campos de tipo numérico. Si el analista lo considera necesario es posible que el filtro pueda ser aplicado sobre una nueva tabla de datos copia de la original manteniendo una historia de los procedimientos aplicados. Contiene los siguientes métodos:

- **Average:** Completa todos los campos vacios de un atributo en base al promedio de los valores que este contiene.
- **Max:** Completa todos los campos vacios con el mayor valor encontrado en el atributo.
- **Min:** Completa todos los campos vacios con el menor valor encontrado en el atributo.
- **Supr:** Elimina todas los registros de una tabla en los cuales el atributo seleccionado contiene valores nulos.
- **Variance:** Completa todos los campos vacios de un atributo teniendo en cuenta la varianza de los valores que el atributo contiene.
- **Sum:** Completa todos los campos vacios de un atributo con el valor resultado de la suma de todos los valores que contiene el atributo.
- **Random_mid:** Completa todos los campos vacios de un atributo con un valor aleatorio entre el promedio mas o menos la desviación estándar de los valores que contiene dicho atributo.
- **Random_full:** Completa todos los campos vacios de un atributo con un valor aleatorio comprendido entre el menor y mayor valor que contenga dicho atributo.

- **Mode:** Completa todos los campos vacios de una atributo con la moda de los valores que contiene ese atributo.
- **Zero:** llena todos los campos vacios con el valor cero (0).
- **Class_rule:** Completa todos los campos vacios de un atributo con el valor resultado de dividir la suma del mayor y el menor valor que contiene el atributo entre dos (2).

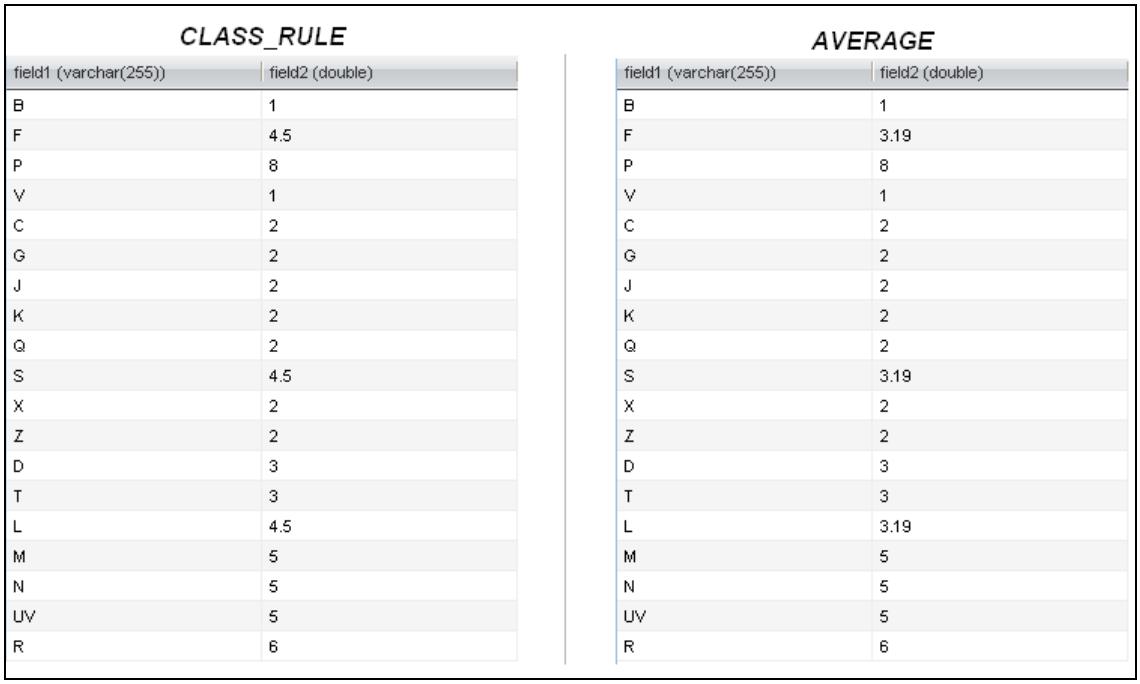
Un ejemplo del funcionamiento de este filtro se comenta a continuación.

En la figura 8.3 se muestra la tabla de datos de entrada en la que se pueden observar 3 campos vacios sobre el atributo de tipo numérico (double) field2, al aplicar el filtro NumberNullClean a través de las opciones **average** y **classRule** se generan dos nuevas tablas de datos respectivamente, en las cuales se observa que los campos nulos han sido reemplazados en la primera tabla por el promedio de los valores (3.19) que contiene el atributo field2 y en la segunda tabla por la marca de clase de los mismos valores (4.5) como se puede observar en la figura 8.4. Si el atributo field2 fuera de tipo numérico (integer), entonces los valores para el promedio y la marca de clase son redondeados a 3 y 5 respectivamente.

Figura 8.3 Datos de entrada antes de aplicar NumberNullClean

field1 (varchar(255))	field2 (double)
B	1
F	
P	8
V	1
C	2
G	2
J	2
K	2
Q	2
S	
X	2
Z	2
D	3
T	3
L	
M	5
N	5
UV	5
R	6

Figura 8.4 Datos de salida una vez aplicado el filtro NumberNullClean con average y class_rule



CLASS_RULE

field1 (varchar(255))	field2 (double)
B	1
F	4.5
P	8
V	1
C	2
G	2
J	2
K	2
Q	2
S	4.5
X	2
Z	2
D	3
T	3
L	4.5
M	5
N	5
UV	5
R	6

AVERAGE

field1 (varchar(255))	field2 (double)
B	1
F	3.19
P	8
V	1
C	2
G	2
J	2
K	2
Q	2
S	3.19
X	2
Z	2
D	3
T	3
L	3.19
M	5
N	5
UV	5
R	6

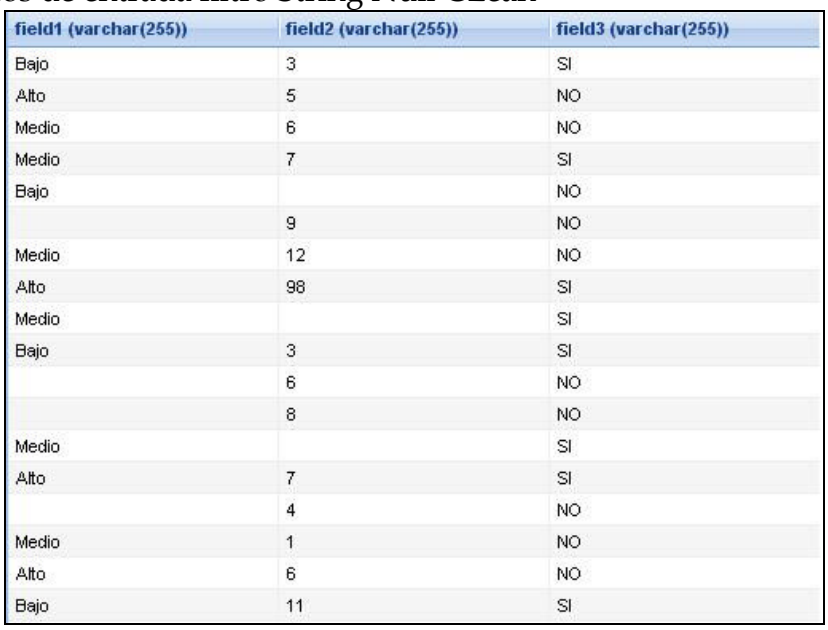
String Null Clean: Esta clase permite llenar todos los campos vacios que contenga un atributo teniendo en cuenta la moda del mismo o teniendo en cuenta la moda pero basada en ciertas condiciones que uno o más del resto de atributos de la tabla deben cumplir. Esta clase se aplica para valores de tipo cadena y contiene los siguientes métodos.

- **AttributeBased:** Este método es el encargado de llenar todos los campos vacios del atributo con el valor más común que se encuentre en el mismo (moda). Un ejemplo del funcionamiento del filtro String Null Clean attribute Based se describe a continuación.

  En la figura 8.5 se muestra la tabla de datos de entrada, antes de ser aplicado el filtro String Null Clean Attribute Based. En la tabla se puede observar que en el atributo "field1" existen 4 campos vacios.

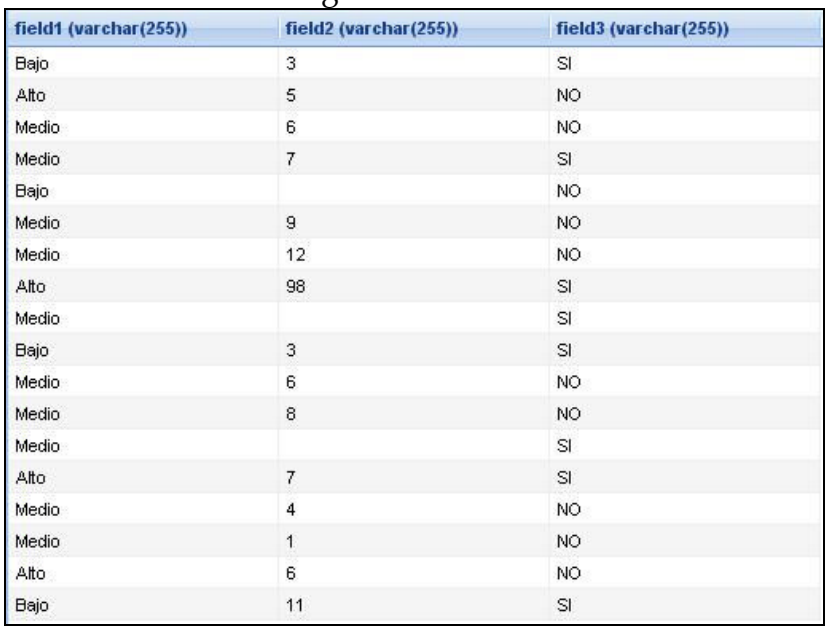
  Al aplicar el filtro, se reemplazan todos los campos vacios por "Medio", la tabla resultante una vez se ha aplicado el filtro se muestra en la figura 8.6.

Figura 8.5 Datos de entrada filtro String Null CLean



field1 (varchar(255))	field2 (varchar(255))	field3 (varchar(255))
Bajo	3	SI
Alto	5	NO
Medio	6	NO
Medio	7	SI
Bajo		NO
	9	NO
Medio	12	NO
Alto	98	SI
Medio		SI
Bajo	3	SI
	6	NO
	8	NO
Medio		SI
Alto	7	SI
	4	NO
Medio	1	NO
Alto	6	NO
Bajo	11	SI

Figura 8.6 Datos de salida filtro String Null CLean



field1 (varchar(255))	field2 (varchar(255))	field3 (varchar(255))
Bajo	3	SI
Alto	5	NO
Medio	6	NO
Medio	7	SI
Bajo		NO
Medio	9	NO
Medio	12	NO
Alto	98	SI
Medio		SI
Bajo	3	SI
Medio	6	NO
Medio	8	NO
Medio		SI
Alto	7	SI
Medio	4	NO
Medio	1	NO
Alto	6	NO
Bajo	11	SI

- **ConditionBased:** Este método se encarga de llenar los campos vacios de un atributo teniendo en cuenta las condiciones que el usuario establece y que deben cumplir uno o más atributos, es decir, extrae el valor que teniendo en

cuenta una serie de condiciones es el más común para ese atributo y lo llena en todos los campos vacios que contenga el mismo. Un ejemplo del funcionamiento del filtro String Null Clean Condition Based se menciona acontinuación.

Teniendo en cuenta el mismo conjunto de datos de entrada tomado en el ejemplo anterior se establece como condición de que el atributo "field2" sea igual a 3. De esta forma los campos vacios del atributo "field1", son reemplazados por "Bajo", dado que el valor más común del atributo "field1" cuando el campo "field2" es igual a 3 es "Bajo". En la figura 8.7 se muestra el resultado una vez se ha aplicado este filtro.

Figura 8.7 Datos de salida filtro String Null CLean (condition based)

field1 (varchar(255))	field2 (varchar(255))	field3 (varchar(255))
Bajo	3	SI
Alto	5	NO
Medio	6	NO
Medio	7	SI
Bajo		NO
Bajo	9	NO
Medio	12	NO
Alto	98	SI
Medio		SI
Bajo	3	SI
Bajo	6	NO
Bajo	8	NO
Medio		SI
Alto	7	SI
Bajo	4	NO
Medio	1	NO
Alto	6	NO
Bajo	11	SI

Trim: Esta clase permite eliminar todos los espacios en blanco que se encuentren al comienzo y al final de cadenas de texto de los atributos seleccionados por el analista.

Missing (Expert Rule Editor): Esta clase tiene como objetivo la revisión de cada una de las sentencias que escribe el analista a manera de condicional "if", analizando que estas sentencias no contengan código no adecuado, permitiendo comandos básicos para asignaciones, operaciones y observación de resultados.

El módulo principal se encuentra en el método **missing**, requiere como parámetros el nombre de la tabla que será objeto del o de los condicionales, además de el texto a analizar. Este método retorna una serie de consultas que son ejecutadas individualmente.

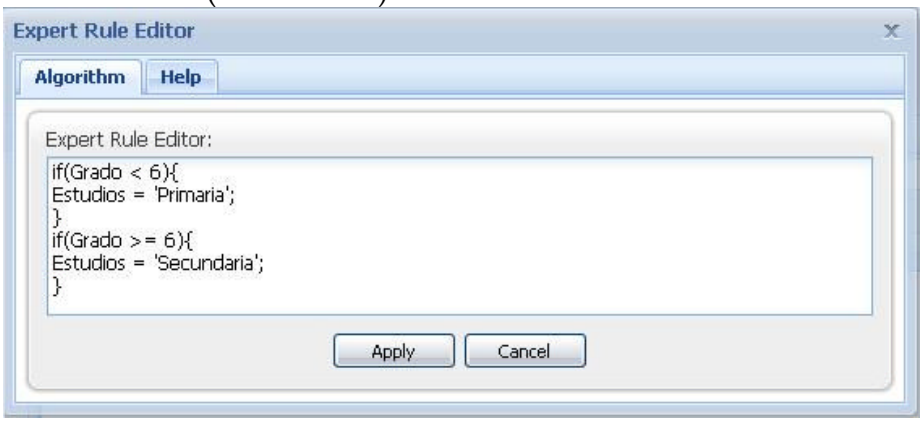
Hay un tipo de condiciones que se ejecutan de forma particular, y estas son las condiciones que involucran ver un resumen de los datos, para esto se utiliza el método **ShowTmpTable** cuyo objetivo es crear una tabla temporal que contenga los datos resultado de la consulta, para que luego el módulo de interfaz se encargue de mostrar esta tabla en pantalla.

Un ejemplo del funcionamiento de esta clase se comenta a continuación, en donde el analista tiene como datos de entrada una tabla con 3 atributos (Nombre, Grado, Estudios) que se muestra en la figura 8.8, y su objetivo es completar los campos vacios del atributo "Estudios", teniendo en cuenta las condiciones que se muestran en la figura 8.9.

Figura 8.8 Datos de entrada (Rule Editor)

Nombre (varchar(255))	Grado (double)	Estudios (varchar(255))
Eduardo	5	
Juan	4	
Franco	8	
Marco	3	
Miguel	11	

Figura 8.9 Condiciones (Rule Editor)



Una vez aplicado el filtro, la tabla resultante se muestra en la figura 8.10, en donde todos los estudiantes que pertenecen a los grados inferiores, es decir, del grado 5 hacia abajo, son de Primaria y los demás son de Secundaria.

Figura 8.10 Datos de salida (Rule Editor)

Nombre (varchar(255))	Grado (double)	Estudios (varchar(255))
Eduardo	5	Primaria
Juan	4	Primaria
Franco	8	Secundaria
Marco	3	Primaria
Miguel	11	Secundaria

Email Cleaner: Esta clase tiene como función la verificación y posterior generación de correcciones a direcciones de correo electrónico (Email) que puedan estar mal escritos, es decir, que no cumplen con la estructura característica de una dirección de correo electrónico.

8.3.3 Paquete Transform Este paquete está conformado por las siguientes clases con sus respectivos métodos:

Discretize: Esta clase se encarga de ejecutar el filtro de discretización sobre un atributo de tipo numérico mediante el método **Discretizar,** el cual recibe como parámetros el nombre del atributo que se desea discretizar, la tabla a la cual pertenece dicho atributo, el número de intervalos (bins) en los cuales se desea discretizar el atributo y una conexión al Sistema Gestor de Base de Datos usando los métodos abrir(), query(), fetch() y cerrar() de la clase **MyConn,** a través de ella se extraen los datos originales para ser tratados y posteriormente ser actualizados de acuerdo a los resultados que genere este método durante su ejecución.

Un ejemplo de aplicación del filtro se observa en la figura 8.11 en donde un atributo de tipo numérico (field2) es seleccionado en base a la tabla de la figura 8.12, los resultados una vez aplicado el filtro se muestran en la figura 8.13.

Figura 8.11 Filtro Discretize

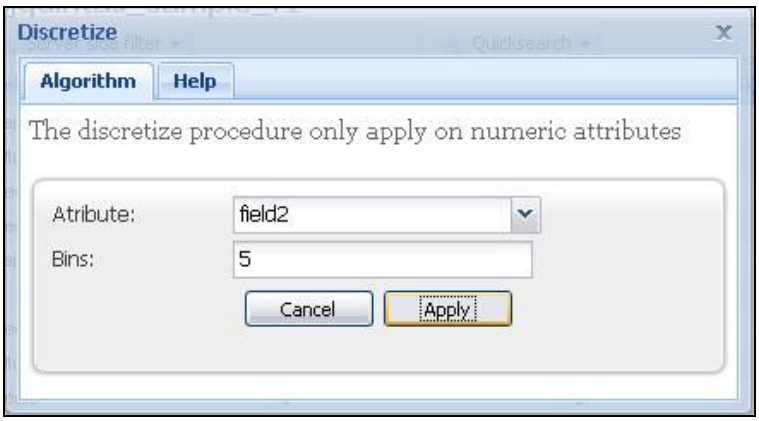


Figura 8.12 Datos de entrada filtro Discretize

field1 (varchar(255))	field2 (int(11))
Bajo	3
Alto	5
Medio	6
Medio	7
Bajo	0
	9
Medio	12
Alto	0
Medio	3
Bajo	6

Figura 8.13 Datos de salida filtro Discretize

field1 (varchar(255))	field2 (varchar(255))
Bajo	(2.4 - 4.8]
Alto	(4.8 - 7.2]
Medio	(4.8 - 7.2]
Medio	(4.8 - 7.2]
Bajo	[0 - 2.4]
	(7.2 - 9.6]
Medio	(9.6 - 12]
Alto	[0 - 2.4]
Medio	(2.4 - 4.8]
Bajo	(4.8 - 7.2]

Normalize: Esta clase ejecuta el filtro de normalización de valores de un atributo de tipo numérico mediante el método **Normalize** que recibe como parámetros el atributo al cual se le desea aplicar el filtro, el nombre de la tabla a la que pertenece el atributo y una conexión al Sistema Gestor de Base de Datos con la que se ejecutan las consultas necesarias para establecer el promedio y la desviación estándar de los valores que componen el atributo para posteriormente actualizarlos teniendo en cuenta la fórmula de normalización que se muestra en la figura 8.14.

Figura 8.14 Fórmula de normalización

$$Z = \frac{X - \mu}{\sigma}$$

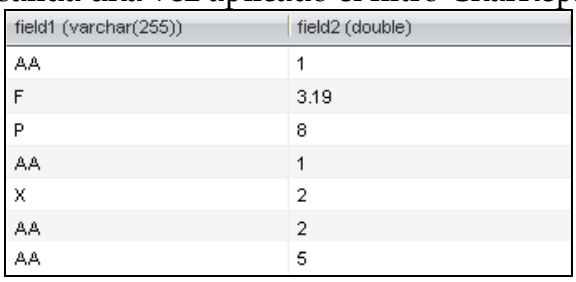
En donde Z = Valor a obtener, X = Valor a normalizar, μ = Media, σ = Desviación Estandar.

CharReplace: Esta clase permite la aplicación del filtro de transformación para reemplazar caracteres que se encuentren en uno o más atributos, por un caracter o cadena de caracteres proporcionada por el analista, todo esto con el fin de transformar y estandarizar una tabla de datos. Para desarrollar esta tarea, se utiliza el método **CharReplace** el cual recibe como parámetros el o la cadena de caracteres a buscar, el o la cadena de caracteres por la que serán reemplazadas las apariciones del primer atributo, un vector con los nombres de los atributos sobre los cuales se aplicará la búsqueda, el nombre de la tabla a la que pertenecen esos atributos y una conexión que servirá para realizar la consulta de búsqueda de los caracteres y su posterior reemplazo. En la figura 8.15 se muestra el conjunto de datos de entrada antes de ser aplicado este filtro. En él se encuentran 4 valores iguales dentro del atributo field1 (V___V), que serán reemplazados por la cadena (AA), el resultado una vez aplicado este filtro se muestra en la figura 8.16

Figura 8.15 Datos de entrada antes de ejecutar el filtro CharReplace

field1 (varchar(255))	field2 (double)
V___V	1
F	3.19
P	8
V___V	1
X	2
V___V	2
V___V	5

Figura 8.16 Datos de salida una vez aplicado el filtro CharReplace



field1 (varchar(255))	field2 (double)
AA	1
F	3.19
P	8
AA	1
X	2
AA	2
AA	5

UpperCase: Esta clase se encarga de transformar todos los valores de los atributos de tipo cadena seleccionados por el analista a Mayúsculas, todo esto como un punto de inicialización para lograr la estandarización de los valores que contiene una determinada tabla de datos. La transformación se realiza mediante el método **toUpper** el cual recibe como parámetros los atributos a los que se les quiere hacer la transformación y una conexión mediante la cual se realizará la consulta al Sistema Gestor de Base de Datos usando los métodos abrir(), query(), fetch() y cerrar() de la clase **MyConn,** para realizar la transformación de los datos. En la tabla 8.1 se muestra un ejemplo de transformación de cadenas de texto a mayúsculas.

Tabla 8.1 Tabla de ejemplo para el filtro UpperCase

Valor Inicial	Valor Final
Jhonnathan	JHONNATHAN
RoBeRt	ROBERT
LiliaN	LILIAN
uPPer cAse convert	UPPER CASE CONVERT

LowerCase: Esta clase se encarga de transformar todos los valores de los atributos de tipo cadena seleccionados por el usuario a Minúsculas, este resulta otro método de estandarización de valores que contiene una determinada tabla de datos. La transformación se realiza mediante el método **toLower** que recibe como parámetros al igual que el método toUpper de la clase **UpperCase,** los atributos a los que se les quiere hacer la transformación y una conexión mediante la cual se realizará la consulta al Sistema Gestor de Base de Datos usando los métodos abrir(), query(), fetch() y cerrar() de la clase **MyConn,** para realizar la transformación de los datos. En la tabla 8.2 se muestra un ejemplo de transformación de cadenas de texto a minúsculas.

Tabla 8.2 Tabla de ejemplo para el filtro LowerCase

Valor Inicial	Valor Final
Jhonnathan	jhonnathan
RoBeRt	robert
LOUIS	lilian
uPPer cAse convert	upper case convert

TruncateString: Esta clase se encarga de aplicar el filtro de transformación mediante el cual se cortan las cadenas de un determinado atributo en base a un tamaño fijo de caracteres y teniendo en cuenta la orientación en la cual serán cortados dichos caracteres. Este filtro se encuentra en el método **TruncateSTR** el cual recibe como parámetros, el atributo sobre el cual será aplicado el filtro, el nombre de la tabla a la que pertenece dicho atributo, la cantidad de caracteres que serán eliminados de las cadenas que componen el atributo, la orientación de la eliminación que hace referencia si los caracteres serán eliminados por la parte izquierda o derecha de la cadena y una conexión con el Sistema Gestor de Base de Datos para actualizar la tabla de datos que se está utilizando. Un ejemplo del funcionamiento de esta clase se muestra en la tabla 8.3 la cual consta de cuatro atributos, la primera muestra la cadena original, la segunda el número de caracteres a eliminar y en las dos restantes el resultado de la eliminación por derecha y por izquierda respectivamente. La conexión con el Sistema Gestor de Base de Datos se realiza a través del uso de la clase **MyConn.**

Tabla 8.3 Ejemplo TruncateString

Original String	Length	Left-Truncate	Rigth-Truncate
12345ASD	3	45ASD	123
UNIVERSALPIC	6	SALPIC	UNIVER
WIRELESS	2	RELESS	WI

NonPrintable: Esta clase es la encargada de buscar dentro de una tabla de datos seleccionada por el analista, todos aquellos caracteres que resulten ser no imprimibles, es decir, que no son caracteres alfa-numéricos, con el fin de que sean mostrados dentro de una tabla temporal editable que permitirá el reemplazo de todos y cada uno de los caracteres encontrados por los que resulten ser convenientes para el analista, todo esto se logra gracias a la interacción entre el método **nonprintable** perteneciente a esta clase y los métodos **algDialog** de la clase **AlgDialog** y los métodos **standargrid** y **define_grid** de la clase **gridClass**. La

descripción de los métodos de las clases **algDialog** y **gridClass** se describen en la sección **8.3.5 Paquete GUI.**

Los parámetros que recibe el método **nonprintable** son la lista de atributos en los cuales se buscarán los caracteres no imprimibles, el nombre de la tabla a la cual pertenecen tales atributos y una conexión con el Sistema Gestor de Base de Datos, que permite la creación de la tabla temporal y la búsqueda campo a campo de todos los caracteres no imprimibles. En la tabla 8.4 se muestra un listado con algunos de los posibles caracteres no imprimibles contemplados en EXDACLET.

Tabla 8.4 Tabla de caracteres no imprimibles

CARACTER	ASCII	CARACTER	ASCII	CARACTER	ASCII	CARACTER	ASCII
☺	1	"	34	ÿ	152	┬	194
☻	2	#	35	ø	155	├	195
♥	3	$	36	£	156	─	196
♦	4	%	38	Ø	157	┼	197
♣	5	'	39	×	158	ã	198
♠	6	(	40	*f*	159	╚	200
•	7	)	41	ª	166	╔	201
◘	8	*	42	º	167	╩	202
○	9	+	43	¿	168	╦	203
◙	10	-	45	®	169	╠	204
♂	11	<	60	¬	170	═	205
♀	12	=	61	½	171	╬	206
♪	13	>	62	¼	172	¤	207
♫	14	¿	63	¡	173	ð	208
☼	15	[	91	«	174	Ð	209
►	16	\	92	»	175	┘	217
◄	17	]	93	░	176	┌	218
↕	18	^	94	▒	177	█	219
‼	19	~	126	▓	178	▄	220
¶	20	⌂	127	│	179	¦	221
§	21	â	131	┤	180	▀	223
▬	22	å	134	©	184	ß	225
↨	23	ê	136	╣	185	õ	228
↑	24	è	138	║	186	µ	230
↓	25	î	140	╗	187	þ	231
→	26	æ	145	╝	188	Þ	232
←	27	Æ	146	¢	189	□	233
∟	28	ô	147	¥	190	ý	236
↔	29	ò	149	┐	191	±	241
▲	30	û	150	└	192	¾	243
▼	31	ù	151	┴	193	÷	246

ChangeAttributeType: Esta clase permite la transformación del tipo de datos que tiene un atributo a uno especificado por el analista, el principal objetivo se centra en la estandarización de los datos que componen dicho atributo. Los métodos que hacen parte de esta clase son:

- ***ToString*:** El cual transforma todos los valores que contenga el atributo a tipo cadena, este método no realiza ningún tipo de verificación sobre los datos dado que cualquier tipo de datos al cual se encuentre asociado un atributo puede ser transformado a cadena.
- ***ChangeColumnToDouble y ChangeColumnToInt:*** Recibe como parámetros el nombre del atributo que se quiere verificar, el nombre de la tabla de datos a la cual está asociado el atributo y una conexión con el Sistema Gestor de Base de Datos mediante la clase **MyConn**. Es el encargado de generar una tabla temporal con todos los datos que tras un análisis interno sobre la consistencia de los mismos, no cumplieron con el tipo de formato especificado por el analista, esta tabla temporal es editable solamente en el atributo que se quiere transformar. Una vez tratados los valores que en un principio no cumplieron con el nuevo tipo de datos, se procede a analizar a través de este método si nuevamente todos los valores cumplen con el tipo de dato, el proceso se repite indefinidamente hasta que todos los valores del atributo estén correctos.

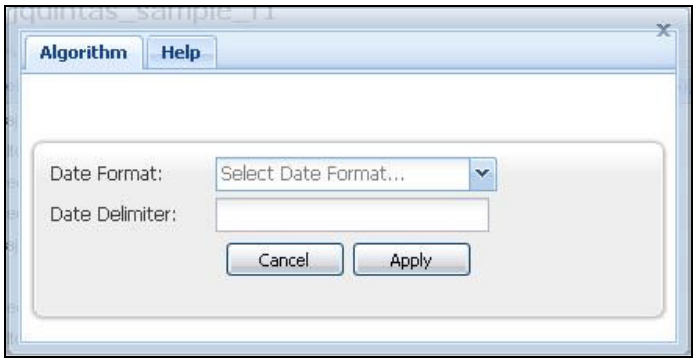
Si al pasar por este método no se encontraron valores que no coincidieran con el nuevo tipo de datos especificado o si los hubo, estos fueron tratados (corregidos o eliminados) entonces mediante la conexión se actualizan los valores del atributo y su tipo.

Cuando el analista requiere transformar un atributo a tipo Date, DateTime o Time (formatos de fecha, fecha y tiempo o tiempo), esta clase ejecuta un método intermedio llamado **VerifyChange**, el cual genera una interfaz de usuario que se muestra en la figura 8.17 en la cual se especifican los delimitadores que tiene los valores que componen el atributo a transformar, una vez especificados estos valores, la clase lanza los métodos correspondientes al cambio de tipo de datos seleccionados. Si el cambio es a tipo Date, entonces se ejecuta el método **ChangeColumnToDate** que recibe como parámetros, el nombre del atributo a transformar, el nombre de la tabla asociada al atributo, el formato en el cual se encuentra la fecha, el delimitador de fecha y una conexión con el Sistema Gestor de Base de Datos resultado de ser instanciada mediante el método abrir() de la clase **MyConn**. Este método se encarga de verificar que todos los valores que se

encuentran en el atributo cumplan con alguno de los formatos válidos existentes para la escritura de un valor de tipo fecha. Si durante este proceso aparecen valores que no cumplen con la estructura de fecha válida entonces este método genera una tabla temporal editable con los valores que no cumplieron con el formato del nuevo tipo de datos, cuando estos valores son corregidos o no se encontraron errores durante el proceso mencionado anteriormente, entonces se llama al método **TransformToDate** que es el encargado de convertir todos los valores de fecha encontrados en base a los delimitadores y formatos en los cuales fueron especificados, al formato estándar utilizado en EXDACLET para el manejo de valores de tipo fecha (date). Una vez transformados, se procede a cambiar el tipo de datos.

Si el cambio es a tipo Datetime o Time, entonces se ejecutan los métodos **ChangeColumnToDateTime** y **ChangeColumnToTime** respectivamente, que de la misma forma que el método **ChangeColumnToDate** verifican la integridad de los valores que se encuentran en el atributo. Si se encuentran inconsistencias, entonces el analista revisa y corrige dichas inconsistencias. Una vez que todos los valores son correctos, entonces se pasa a los métodos **TransformToDateTime y TransformToTime** respectivamente los cuales transforman todos esos valores y los estandarizan para ser ingresados en la tabla de datos.

Figura 8.17 Interfaz de definición de delimitador de fechas



ChangeAttributeName: Esta clase es la encargada de cambiar el nombre de un atributo mediante el método **ChangeAttName,** el cual recibe como parámetros el nombre del atributo a transformar, el nombre de la tabla a la cual se encuentra asociado el atributo, el nuevo nombre especificado por el analista y una conexión. El curso normal de los eventos empieza cuando el usuario selecciona el atributo al cual le quiere cambiar el nombre y específica el nuevo nombre que le será

asignado, ese nuevo nombre es verificado con el fin de que no exista un atributo igual. El paso siguiente consiste en determinar que la nueva cadena este correcta en su estructura y finalmente se procede a actualizar el atributo si durante todo el proceso ocurre un error, la operación es cancelada y se muestra un mensaje de error.

AddAtribute: Esta clase permite agregar un atributo a una determinada tabla de datos mediante el método **AddAtt** que se compone de los siguientes parámetros:

- *TableName:* hace referencia al nombre de la tabla en la cual será adicionado el atributo.
- *AttributeName:* hace referencia al nombre del atributo que será adicionado.
- *AttributeType:* en el que se define el tipo de datos que identifica al nuevo atributo.
- *Connection:* mediante la cual se establece una conexión con el Sistema Gestor de Base de Datos para actualizar la tabla de datos que se está modificando, usando el método query() de la clase **MyConn**.

El flujo normal de los eventos comienza cuando el usuario define el nombre que identificará al nuevo atributo y el tipo de datos que le será asignado, durante el evento de escritura del nuevo nombre se verifica que ese nombre no exista dentro de la misma tabla de datos, posteriormente y mediante la conexión se establecen los cambios pertinentes sobre la tabla con una sentencia SQL. Si todo es correcto la tabla de datos se recargará nuevamente de otro modo se mostrará un mensaje de error.

DeleteAttribute: Esta clase se encarga de eliminar un atributo de una determinada tabla de datos mediante el método **DeleteAtt** el cual recibe como parámetros el nombre del atributo a eliminar, la tabla de datos a la que pertenece ese atributo y una conexión con el Sistema Gestor de Base de Datos mediante la cual se ejecuta la sentencia SQL que elimina el atributo de esa tabla de datos usando el método query() de la clase **MyConn**. El curso normal de los eventos se desarrolla cuando el analista carga una tabla de datos, selecciona el filtro de transformación **Delete Attributes** y escoge el atributo que desea eliminar, si todo es correcto se recargará la tabla de datos. Si ocurre un error se cancela la operación y se mostrará un mensaje de error.

ClearAttribute: Esta clase se encarga de limpiar todo el contenido de un determinado atributo sin tener en cuenta el tipo de datos que lo identifica a través del método **AttClear** el cual se ejecuta teniendo en cuenta los siguientes parámetros:

- *AttName:* Contiene el nombre del atributo al cual se le quiere vaciar su contenido.
- *TableName:* Contiene el nombre de la tabla en donde se encuentra el atributo a limpiar.
- *Connection: E*s una conexión con el Sistema Gestor de Base de Datos con el fin de ejecutar la consulta SQL para limpiar el contenido del atributo y actualizar posteriormente la tabla de datos utilizando el método query() de la clase **MyConn**.

El curso normal de los eventos inicia en el momento en que el usuario carga una tabla de datos y selecciona el filtro de transformación **Attribute Clear**, posteriormente selecciona el atributo del cual quiere borrar su contenido y procede a aplicar el filtro. Si durante el proceso hubo un error se cancela la operación y se muestra un mensaje de error de otro modo se actualiza la tabla de datos cargada.

Binarize: Esta clase es la encargada de transformar los atributos seleccionados por parte del analista en una nueva tabla la cual se compone de valores unos (1) y ceros (0) y de una serie de atributos basados en los diferentes valores que se encuentren en los atributos seleccionados. Esta clase se compone del método **MakeIt** el cual recibe como parámetros el nombre de la tabla de datos original, el nombre de la nueva tabla de datos binarizada, un vector con los atributos que se van a binarizar y una conexión al Sistema Gestor de Base de Datos instanciada por el método abrir() de la clase **myConn**.

El curso normal de los eventos se desarrolla cuando el analista carga una tabla de datos, selecciona el filtro de transformación **Binarize,** luego escoge los atributos que desea binarizar, en base a estos se crea la nueva tabla de datos, cuando aplica el filtro, el método **MakeIt** toma los atributos e identifica todos los diferentes valores que estos contienen con el fin de ir creando una estructura de tabla en donde cada valor se convertirá en un nuevo atributo, hecho esto, éste método recorre toda la tabla de datos original fila a fila y compara los valores con la nueva tabla creada, esto significa que, cada valor de la tabla original es buscado como un

atributo de la nueva tabla, si existe alguna coincidencia entonces para ese atributo es insertado el valor uno (1) de otro modo, se inserta el valor cero (0).
Una vez es terminado el proceso, se carga la nueva tabla de datos con todos los valores binarizados. Un ejemplo del funcionamiento se muestra en las tablas 8.5 y 8.6 en donde se aprecia el resultado de binarizar los datos de la tabla 8.5 en la tabla 8.6

Tabla 8.5 Datos de entrada (binarize)

C1	**C2**	**C3**	**C4**	**C5**
A	B	D	B	A
B	C	A	C	D
C	D	A	B	D

Tabla 8.6 Datos de salida (binarize)

C1_A	C1_B	C1_C	C2_B	C2_C	C2_D	C3_D	C3_A	C4_B	C4_C	C5_A	C5_D
1	0	0	1	0	0	1	0	1	0	1	0
0	1	0	0	1	0	0	1	0	1	0	1
0	0	1	0	0	1	0	1	1	0	0	1

Table Encoder/Decoder: Esta clase se encarga de transformar una tabla de datos en una tabla con valores codificados, estos valores son números consecutivos teniendo en cuenta los diferentes valores de la tabla original y son soportados mediante una tabla denominada **Diccionario de Datos** en la cual se describen los diferentes valores encontrados en la tabla original y su correspondiente código asociado a la nueva tabla creada. Además se encarga de transformar una tabla ya codificada a sus valores originales teniendo en cuenta una tabla de **Diccionario de Datos** en especial. Los métodos que hacen parte de esta clase son los siguientes:

- *EncodeAttribute:* Recibe como parámetros un vector con los atributos a codificar, el nombre de la tabla asociada a esos atributos y una conexión. Su funcionamiento se basa en extraer todos los diferentes valores que existen en los atributos seleccionados y asignarle un código único y consecutivo a cada uno. Estos datos (valor, código y atributo) son consignados en un **Diccionario de Datos** que a su vez es una tabla. Posteriormente se reemplazan todos los valores en los atributos seleccionados, por los códigos ya asignados en el **Diccionario de Datos.**
- *DecodeAttribute:* Recibe como parámetros el nombre de la tabla de datos a decodificar, el nombre del **Diccionario de Datos** que tendrá como base la

decodificación y una conexión al Sistema Gestor de Base de Datos. Su funcionamiento se centra en reemplazar los códigos que se encuentran en la tabla codificada con los valores originales que se encuentran almacenados en el diccionario de datos. Un ejemplo del funcionamiento de esta clase se presenta en las tablas 8.7, 8.8 y 8.9 en donde se muestran la tabla original, el diccionario de datos utilizado para la codificación, y la tabla codificada.

Tabla 8.7 Tabla de datos original (encode)

Estado_Civil	Grado_Estudios
Casado	Primaria
Soltero	Secundaria
Casado	Secundaria
Viudo	Primaria

Tabla 8.8 Diccionario de datos codificada (encode)

Codigo	Atributo	Valor
1	Estado_Civil	Casado
2	Estado_Civil	Soltero
3	Estado_Civil	Viudo
4	Grado_Estudios	Primaria
5	Grado_Estudios	Secundaria

Tabla 8.9 Tabla de datos codificada (encode)

Estado_Civil	Grado_Estudios
1	4
2	5
1	5
3	4

8.3.4 Paquete Selection Este paquete cuenta con las siguientes clases y sus respectivos métodos.

MaxMinLength: Esta clase es la encarga de ejecutar los filtros de selección **MaxLength** y **MinLength** mediante el método **MMLength.** Este método es el encargado de generar una tabla de datos temporal que se muestra a través de la clase **StandarGrid** y su método **defineGrid**. Esta tabla temporal contiene todas las

filas que cumplen con la condición que se establece al ser aplicado alguno de los dos filtros. Esta condición viene definida por uno de los parámetros que recibe el método MMLength denominado *type*, en el que se establece si se aplicará el filtro MaxLength o MinLength. Además de este parámetro, el método recibe el nombre del atributo en el cual se aplicará el filtro, el nombre de la tabla asociada a dicho atributo y una conexión que permite la búsqueda y la posterior creación de la tabla de datos temporal.

El curso normal de los eventos para este filtro comienza cuando el analista carga una tabla de datos en la herramienta y procede a seleccionar el filtro de selección **MaxLength** o **MinLength,** en donde se debe especificar el atributo de búsqueda y el tamaño mínimo que deben tener las cadenas de caracteres para ser mostradas dentro de la tabla temporal. Si todo es correcto entonces se generará dicha tabla que será editable, de otro modo se cancela la operación y se muestra un mensaje de error.

JaroWinkler: Esta clase es la encargada de generar una tabla temporal que contiene varios conjuntos de filas, cada conjunto separado por una fila delimitadora. Estos conjuntos son el resultado de todas las posibles coincidencias que pueden existir en base al algoritmo Jaro-Winkler, entre los valores que se encuentran dentro de uno o más atributos seleccionados por el analista. Esta tabla se muestra a través de la clase **StandarGrid** y su método **defineGrid.**

La clase **JaroWinkler** se compone de los métodos: *MakeJaroWinkler, GetDistance y ClearString***.** El primero de ellos es el encargado de realizar todos los procedimientos requeridos para la extracción de cada valor de los atributos seleccionados, luego llama al método *ClearString* que elimina la posible basura que pueda contener cada valor y posteriormente al método *GetDistance* el cual ejecuta internamente el algoritmo de Jaro-Winkler, extrae las distancias y analiza la posible similitud entre dos valores. Si esos dos valores tienen una similitud lo suficientemente alta, entonces son insertados con todos sus respectivos atributos en la tabla temporal, de otro modo, seguirá buscando por toda la tabla de datos original hasta encontrar valores similares.

Además, la tabla de datos temporal generada tiene la posibilidad de ser tratada, es decir, una vez ejecutado este filtro, el analista verifica y toma medidas sobre los datos que realmente son considerados duplicados.

En la figura 8.18 se muestra el pseudocódigo de la implementación de este algoritmo.

Figura 8.18 Pseudocódigo algoritmo Jaro-Winkler

```
1   Array = Lista de Transacciones
2   Fila = Transacción actual
3   ResToShow = Objeto con la lista de resultados
4   Contador = 0
5   while (Fila = ResToShow){
6       Array = Fila
7       for(z = 0; z < Contador; z++){
8           Word = Array[z]
9           Longitud1 = len(Word)
10          Longitud2 = len(Fila)
11          if(Longitud1 > Longitud2){
12              aux = Longitud2     Longitud2 = Longitud1       Longitud1 = aux
13              auxstr = Fila       Fila = Word     Word = auxstr
14          }
15          Lmin = Longitud1
16          Lmax = Longitud2
17          for(i = 0; i < Longitud1; i++){     Fl[i] = false       }
18          for(i = 0; i < Longitud2; i++){     Fl[i] = false       }
19          m = ceil((Lmax/2)-1)
20          common = 0
21          tr = 0
22          a1 = str_split(Word)        a2 = str_split(Fila)
23          for (i = 0; i < Longitud1; i++){
24              if(m >= i){
25                  f = 0
26                  l = i + m
27              }else{
28                  f = i - m
29                  l = i + m
30              }
31              if(l > lmax){   l = lmax    }
32              for (j = f; j < l; j++){
33                  if(a2[j] == a1[i] && f2[j] == false){
34                      common = common + 1
35                      f1[i] = true
36                      f2[j] = true
37                      break
38                  }
39              }
40          }
41          l = 0
42          for(i = 0; i < Longitud1; i++){
43              if(f1[i] == true){
44                  for(j = l; j < Longitud2;j++){
45                      if(f2[$j] == true){
46                          l = j + 1;
47                          if(a1[i] != a2[j]){     tr = tr + 0.5       }
48                          break
49                      }
50                  }
51              }
52          }
53          wcd = 1/3       wrd = 1/3       wtr = 1/3
54          if(common != 0){
55              jaro = wcd * common / l1 + wrd * common / l2 + wtr * (common - tr) / common
56          }else{
57              jaro = 0;
58          }
59      }
60      Contador++
61  }
```

DoubleMetaphone: Esta clase tiene el mismo principio de funcionamiento que tiene la clase **JaroWinkler** la diferencia se enmarca en los métodos que son llamados a ejecución y el procedimiento de ejecución del algoritmo, puesto que los resultados son generados y pueden ser tratados de la misma forma que la clase anteriormente mencionada.

Los métodos que componen a esta clase son: **DMP,** el cual recibe como parámetros una lista con los atributos sobre los cuales se aplicará el filtro, la tabla asociada a ese grupo de atributos y una conexión con el Sistema Gestor de Base de Datos para poder crear e insertar registros en la tabla temporal generada mediante el método *query()* de la clase **MyConn**. Su objetivo es ir extrayendo y cargando en memoria todas las filas de la tabla original en base a los atributos seleccionados por el analista para proceder a ejecutar el método **GetDoubleMetaPhone,** en donde se pasa como parámetro el valor de cada atributo para ser codificado según el algoritmo DoubleMetaphone y posteriormente buscar todas las posibles coincidencias con el nuevo valor generado. Este procedimiento se repite hasta haber extraído todas las filas de la tabla original, haberlas cargado en memoria y analizarlas para su evaluación. En la figura 8.19 se muestra una sección del código para la implementación del algoritmo. En ella se puede apreciar que existen cerca de 100 líneas de código para el caso de que una palabra contenga la letra "C", y también, que se contemplan una gran variedad de casos para posibles variaciones solo de esa letra. Dentro del código completo se estableció que el total de líneas de código para el caso de la letra "C" es un total de 180 y en general para todo el algoritmo aproximadamente 1000. En las que se almacenan una gran cantidad de posibles variaciones para todas las letras del abecedario, contemplando también diferencias de idiomas.

Levenshtein: Al igual que las clases **JaroWinkler** y **DoubleMetaphone** genera una tabla temporal tratable por el analista con el fin de buscar posibles coincidencias entre valores de tipo cadena. Se compone del método **getDistance** el cual analiza toda la tabla de datos original buscando de valores de dos en dos con el fin de encontrar la mayor cantidad posible de coincidencias. Los parámetros de entrada que recibe este método son: un listado de los atributos sobre los cuales el usuario desea aplicar la búsqueda, el nombre de la tabla de datos en la cual se encuentran estos atributos y una conexión con el Sistema Gestor de Base de Datos. El pseudocódigo de este algoritmo se muestra en la figura 8.20.

Figura 8.19 Sección de código del algoritmo DoubleMetaphone

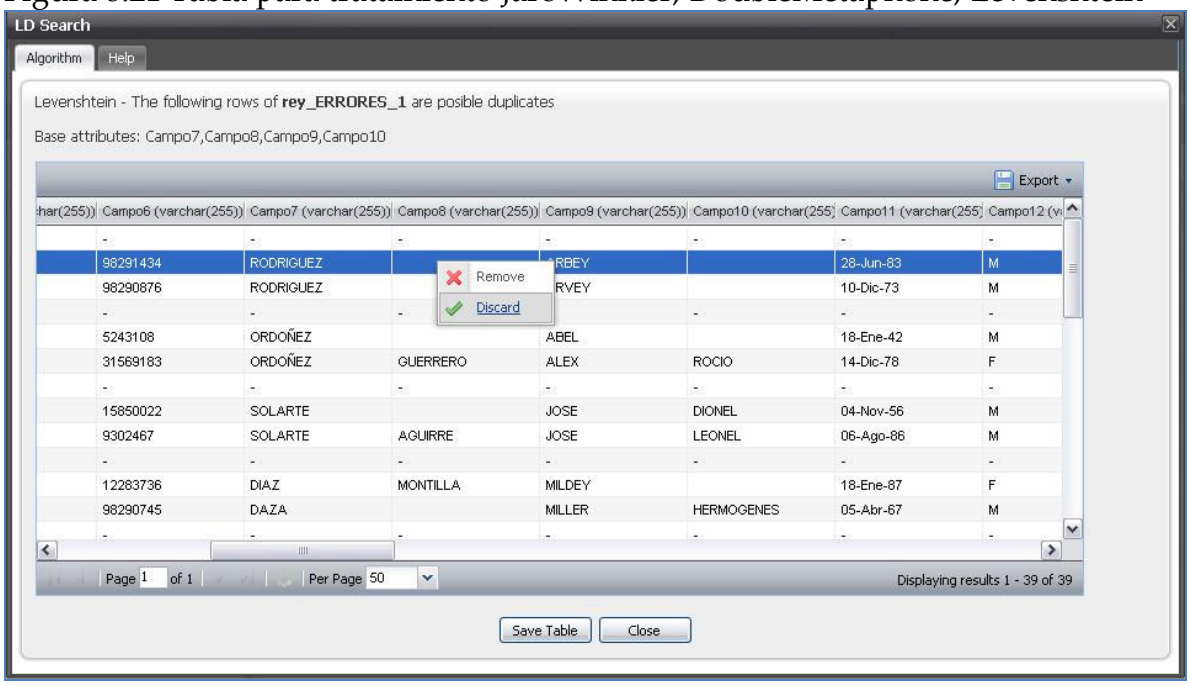
```
185         case 'C':
186           // various gremanic
187           if ((this->current > 1)
188               && !this->IsVowel(this->original, this->current - 2)
189               && this->StringAt(this->original, this->current - 1, 3,
190                         array("ACH"))
191               && ((substr(this->original, this->current + 2, 1) != 'I')
192                   && ((substr(this->original, this->current + 2, 1) != 'E')
193                       || this->StringAt(this->original, this->current - 2, 6,
194                                 array("BACHER", "MACHER"))))) {
195
196             this->primary   .= 'K';
197             this->secondary .= 'K';
198             this->current += 2;
199             break;
200           }
201           // special case 'caesar'
202           if ((this->current == 0)
203               && this->StringAt(this->original, this->current, 6,
204                         array("CAESAR"))) {
205             this->primary   .= 'S';
206             this->secondary .= 'S';
207             this->current += 2;
208             break;
209           }
210           // italian 'chianti'
211           if (this->StringAt(this->original, this->current, 4,
212                         array("CHIA"))) {
213             this->primary   .= 'K';
214             this->secondary .= 'K';
215             this->current += 2;
216             break;
217           }
218           if (this->StringAt(this->original, this->current, 2, array("CH"))) {
219             // find 'michael'
220             if ((this->current > 0)
221                 && this->StringAt(this->original, this->current, 4,
222                         array("CHAE"))) {
223               this->primary   .= 'K';
224               this->secondary .= 'X';
225               this->current += 2;
226               break;
227             }
228             // greek roots e.g. 'chemistry', 'chorus'
229             if ((this->current == 0)
230                 && (this->StringAt(this->original, this->current + 1, 5,
231                         array("HARAC", "HARIS"))
232                     || this->StringAt(this->original, this->current + 1, 3,
233                               array("HOR", "HYM", "HIA", "HEM")))
234                 && !this->StringAt(this->original, 0, 5, array("CHORE"))) {
235               this->primary   .= 'K';
236               this->secondary .= 'K';
237               this->current += 2;
238               break;
239             }
240
241             // germanic, greek, or otherwise 'ch' for 'kh' sound
242             if ((this->StringAt(this->original, 0, 4, array("VAN ", "VON "))
243                  || this->StringAt(this->original, 0, 3, array("SCH")))
244                 // 'architect' but not 'arch', orchestra', 'orchid'
245                 || this->StringAt(this->original, this->current - 2, 6,
246                         array("ORCHES", "ARCHIT", "ORCHID"))
247                 || this->StringAt(this->original, this->current + 2, 1,
248                         array("T", "S"))
249                 || ((this->StringAt(this->original, this->current - 1, 1,
250                         array("A","O","U","E"))
251                      || (this->current == 0))
252                     // e.g. 'wachtler', 'weschsler', but not 'tichner'
253                     && this->StringAt(this->original, this->current + 2, 1,
254                         array("L","R","N","M","B","H","F","V","W"," "))))) {
255               this->primary   .= 'K';
256               this->secondary .= 'K';
257             }
```

Figura 8.20 Pseudocódigo algoritmo Levenshtein Distance

```
int LevenshteinDistance(char s[1..m], char t[1..n])
   // d is a table with m+1 rows and n+1 columns
   declare int d[0..m, 0..n]

   for i from 0 to m
       d[i, 0] := i
   for j from 1 to n
       d[0, j] := j

   for i from 1 to m
       for j from 1 to n
           if s[i] = t[j] then cost := 0
                          else cost := 1
           d[i, j] := minimum(
                                d[i-1, j] + 1,      // deletion
                                d[i, j-1] + 1,      // insertion
                                d[i-1, j-1] + cost   // substitution
                            )

   return d[m, n]
```

Un ejemplo del funcionamiento de las clases **Levenhtein, JaroWinkler y DoubleMetaphone** se muestra en la figura 8.21, donde los resultados pueden ser tratados mediante menús de contexto, guardados o exportados a los formatos de archivo soportados por EXDACLET.

Figura 8.21 Tabla para tratamiento JaroWinkler, DoubleMetaphone, Levenshtein



Soundexx: Esta clase es la encargada de ejecutar el algoritmo para la búsqueda de similitud entre cadenas denominado Soundex, este procedimiento se ejecuta mediante el método **Soundex,** que recibe como parámetros una lista con los atributos sobre los cuales se buscará posibles similitudes, el nombre de la tabla a la cual se encuentran asociados tales atributos y una conexión con el Sistema Gestor de Base de Datos. El objetivo de este método es generar una tabla de datos temporal en donde se encuentren todos los registros que según el algoritmo de Soundex hayan tenido una alta probabilidad de ser iguales, esta tabla contiene además de los atributos seleccionados por el analista, todos los demás atributos que componen a la tabla de datos original. Durante todo el proceso de ejecución de este filtro aparece una alta interacción entre la herramienta y el sistema gestor de base de datos dando pie a que el tiempo de ejecución del mismo resulte ser muy eficiente. En la figura 8.22 se muestra el pseudocódigo de la implementación de este algoritmo.

Figura 8.22 Pseudocódigo algoritmo SOUNDEX

```
1   S = Cadena a codificar
2   SIZE = Tamaño de la cadena codificada
3   x = Convertir la cadena a mayusculas
4   firstLetter = x[0]
5   // convertir letras a codigos numericos
6   for (int i = 0; i < len(x); i++) {
7       switch (x[i]) {
8           case 'B':
9           case 'F':
10          case 'P':
11          case 'V': { x[i] = '1' break }
12          case 'C':
13          case 'G':
14          case 'J':
15          case 'K':
16          case 'Q':
17          case 'S':
18          case 'X':
19          case 'Z': { x[i] = '2' break }
20          case 'D':
21          case 'T': { x[i] = '3' break }
22          case 'L': { x[i] = '4' break }
23          case 'M':
24          case 'N': { x[i] = '5' break }
25          case 'R': { x[i] = '6' break }
26          default:  { x[i] = '0' break }
27      }
28  }
29  // remove duplicates
30  output = firstLetter
31  last = x[0]
32  for (i = 1; i < len(x); i++) {
33      if (x[i] != '0' && x[i] != last) {
34          last = x[i];
35          output:= last;
36      }
37  }
38  // pad with 0's or truncate
39  for (i = len(output); i < SIZE; i++) {
40      output := '0';
41  }
42  output = substr(output,0, SIZE);
```

La tabla temporal que se genera puede ser tratada de la misma manera como se trata en los filtros tales como **Levenshtein, JaroWinkler y DoubleMetaphone** como se muestra en la Figura 8.21.

Duplicates: Esta clase es la encargada de generar una tabla temporal en la cual se encuentran todos los registros duplicados teniendo en cuenta los atributos clave para realizar la búsqueda, esta clase se compone del método **duplicates** el cual recibe como parámetros una lista con los atributos clave de la búsqueda, es decir, con los atributos sobre los que se buscará una coincidencia exacta entre los valores que contengan, el nombre de la tabla de datos a la cual se encuentran asociados dichos atributos y una conexión con el Sistema Gestor de Base de Datos. La tabla temporal que se genera a partir de esta clase puede ser tratada de la misma manera que los anteriores filtros de selección que aplican algoritmos para búsqueda de coincidencias entre cadenas.

El curso normal para la ejecución de esta clase y de igual manera para las clases **Levenshtein, JaroWinkler, DoubleMetaphone y Soundexx** comienza cuando el analista carga en la herramienta una tabla de datos sobre la cual desea aplicar el filtro, posteriormente selecciona cualquiera de los siguientes filtros de selección, **Soundex Search, Duplicates Search, LD Search, Jaro-Winkler Search, DoubleMetaphone Search,** paso siguiente a esta acción el analista selecciona los atributos clave, esto se refiere a escoger sobre qué atributos se realizará la búsqueda, por ejemplo, si son seleccionados 4 atributos, entonces se aplicará la búsqueda que tomará a esos 4 atributos como a uno solo, es decir que, si al hacer un análisis con dos registros, 3 de los cuatro atributos seleccionados pasan dicho análisis, estos dos registros no serán insertados en la tabla temporal. Ahora, si son seleccionados solamente 3 atributos, entonces los dos registros anteriormente mencionados si serán insertados.

Una vez hecho el procedimiento de selección de atributos, la herramienta procede a analizar la tabla de datos original y los atributos seleccionados y ejecuta el filtro mediante el método correspondiente al filtro seleccionado, si todo es correcto, se muestra en pantalla la tabla de datos con los resultados encontrados, si no hubo resultados la tabla de datos estará vacía, de otro modo se muestra un mensaje de error informando del tipo de error que ocurrió durante el proceso.

Details: Esta clase permite generar una tabla temporal, la cual contiene todos los atributos de la tabla original y adiciona uno más, que muestra información referente a la cantidad de registros con los mismos datos en todos sus atributos. Su funcionamiento se da a partir del método **tableDetails** que recibe como parámetros los atributos que se requieren mostrar y el nombre de la tabla de datos asociada a esos atributos.

Union: Esta clase es la encargada de realizar la unión de 2 tablas teniendo en cuenta los atributos que seleccione el analista y funciona mediante los siguientes métodos.

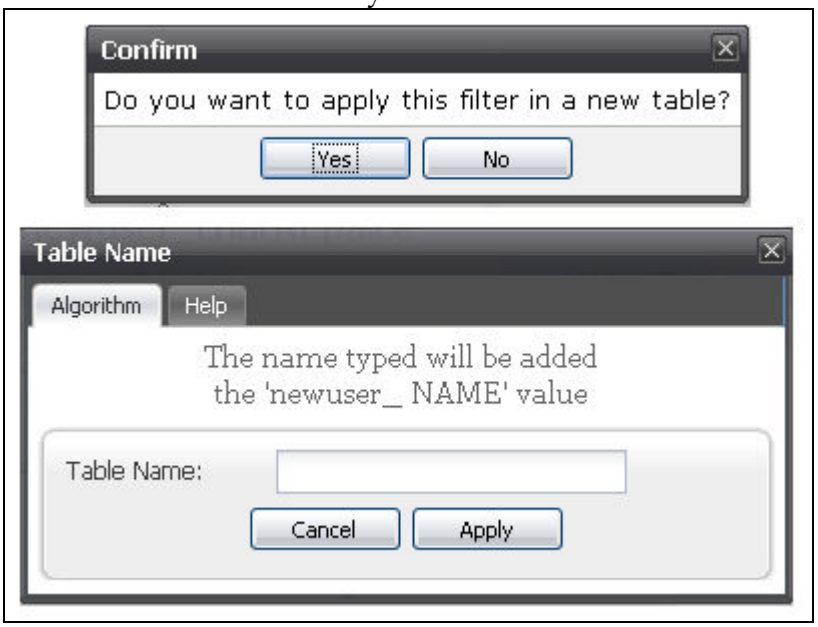
- *initStartUnion:* Es el encargado de generar la interfaz gráfica sobre la cual se desarrolla todo el procedimiento de unión de tablas.
- *StartUnion:* Muestra un listado con todas las tablas de datos que tiene el usuario con el fin de que seleccione dos de ellas.
- *VerifyUnion:* Este método permite al analista escoger los atributos de las dos tablas de datos seleccionadas, los cuales se convertirán en los atributos de la nueva tabla de datos.
- *DoUnion:* Es el encargado de verificar la consistencia de los atributos seleccionados en las dos tablas de datos. Esto hace referencia a que verifica que la cantidad de atributos seleccionados en la primera tabla sea la misma que la de los atributos seleccionados en la segunda tabla, también de que los atributos seleccionados y que serán unidos, cumplan con el mismo tipo de datos. Posteriormente se encarga de preguntar al analista el nombre de la nueva tabla de datos que se generará a partir de las dos primeras.
- *ExecUnion:* Es el método encargado de llevar a cabo el proceso de unión, extrayendo los valores de los atributos seleccionados en las dos tablas de datos, creando la nueva tabla de datos con la misma cantidad de atributos seleccionados y con el nombre suministrado por el analista y finalmente insertando todos los valores en la nueva tabla de datos.

Todas las clases que se encuentran dentro de los paquetes de Cleaning (limpieza), Transform (transformación) y Selection (Selección), tienen un paso intermedio de verificación en donde se pregunta si el filtro será aplicado sobre la misma tabla de datos que se esté utilizando en el momento, o si se aplicará en una copia de la misma con un nombre definido por parte del analista como se muestra en la figura 8.23. Todo esto con el fin de que se mantenga una historia de todos los procedimientos que el analista ha aplicado sobre la tabla desde el momento en que

esta es cargada en la herramienta mediante alguno de los formatos de archivo valido soportado.

Este procedimiento lo realiza el método **Verify** que se encuentra en la clase **Algorithms**, encargada de crear todas las instancias de clases requeridas durante todo el proceso de limpieza y transformación dependiendo de los filtros y procedimientos que el analista desee utilizar. En caso de ser confirmado el evento de aplicación del filtro sobre una nueva tabla en el método **Verify**, se ejecutan los métodos **CreateDup** y **genDup** los cuales se encargan de mostrar la interfaz de usuario para definir el nuevo nombre de la tabla sobre la cual será aplicado un filtro y crear físicamente dentro de la herramienta la tabla de datos definitiva sobre la cual será aplicado el filtro seleccionado por el analista.

Figura 8.23 Interfaces de confirmación y creación de una nueva tabla



8.3.5 Paquete GUI Este paquete se compone de una gran variedad de clases para el manejo de la interfaz de usuario las cuales permiten hacer de EXDACLET una herramienta lo suficientemente amigable para el usuario final al momento de cargar tablas de datos y aplicar sobre ellas procesos de limpieza, transformación y selección.

MainCode: En esta clase se desarrolla la interfaz principal de la herramienta, además interactúa con otras clases de la biblioteca EXTJS mediante las cuales se

logran los efectos y se establece la distribución del espacio de trabajo de la herramienta. Su implementación se muestra en la figura 8.24.

Dentro de la interfaz principal se distinguen tres secciones fundamentales a saber: en la parte izquierda se encuentra el **Selector** en el cual el analista puede seleccionar y cargar una tabla de datos. En la parte central se encuentra el **Espacio De Trabajo** en el cual se carga la tabla de datos seleccionada por el analista a través del **Selector**, también, en esta sección, una vez cargada una tabla de datos, aparece una pestaña que permite la consulta de las estadísticas de la tabla de datos, esto con el fin de que el analista tenga un método gráfico para observar la consistencia de los atributos que la componen. Y finalmente en la parte derecha se encuentra el **Algorithms List** que es un menú desplegable dentro del cual se listan todos los filtros implementados dentro de EXDACLET agrupados según su objetivo en procesos de Pre-carga, Limpieza, Transformación y Selección; aparece también dentro de este menú la opción para salir de la herramienta.

Figura 8.24 Clase MainCode. Interfaz principal de la herramienta

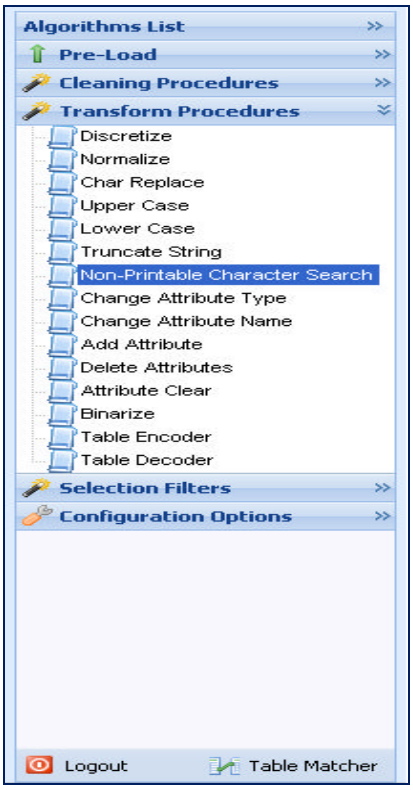
El método **init_app** es el encargado de iniciar y establecer instancias desde el servidor hacia el cliente mediante métodos de la clase **EXTJS** que serán utilizados por la herramienta. Dentro de este método se encuentra el llamado al sub-método **fadeOut** el cual permite que una vez iniciada la sesión para el ingreso a la herramienta, se muestre un efecto amigable de carga sobre la pantalla mientras se

inician todos los componentes necesarios para la presentación de la interfaz principal. Posteriormente es llamado el método **init_container** en el que se generan las tres secciones de la interfaz de la herramienta y las dos pestañas que se muestran dentro del espacio de trabajo referentes a la tabla de datos y a las estadísticas de la misma. Todo esto mediante los sub-métodos de la clase **EXTJS, Ext.BorderLayout** y **Ext.TabPanel.**

Una vez se han definido las tres secciones que componen la herramienta, entonces se procede a llenar las secciones **Selector** mediante el llamado a la clase **TreePHP y Algorithms List** mediante el método **init_accordion** de la clase **MainCode**.

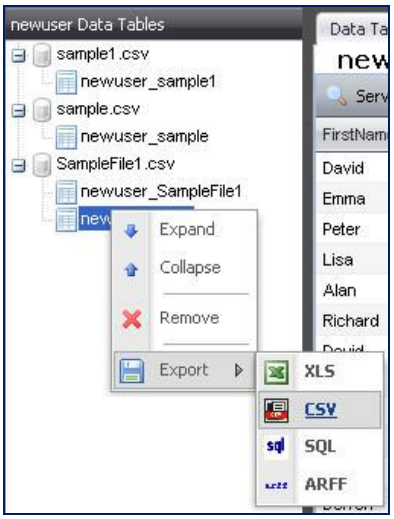
- *Init_accordion():* es el encargado de generar toda la estructura de los menús desplegables y todo el contenido de los mismos. Este método interactúa con el método **Ext.ux.Accordion** de la clase **EXTJS** para generar el acordeón. Internamente llama a los métodos *init_algTree() y fireAlg*() encargados de generar los contenidos de cada sección del acordeón y de ejecutar el filtro que seleccione el analista respectivamente. En la figura 8.25 se muestra el acordeón.

Figura 8.25 Acordeón



TreePHP: Es la clase encargada de generar una árbol compuesto por todos los archivos que el analista ha cargado dentro de la herramienta. Cada archivo (nodo) se subdivide en hojas las cuales hacen referencia a todas las tablas de datos que se han generado tras aplicar procesos de limpieza, selección o transformación. En la figura 8.26 se muestra el árbol y las acciones que se pueden realizar sobre él.

Figura 8.26 Sección Selector



Los métodos mediante los cuales se logra la generación del árbol son:

- *Tree():* Genera todo el código JavaScript que será ejecutado para mostrar el árbol. Características internas dentro de este método también lo son la generación del código para el despliegue de un menú contextual para cada nodo del árbol en donde, si el nodo es hoja, puede ser eliminado siempre y cuando hayan más nodos hoja junto a él, y también puede ser exportado a cualquiera de los formatos de archivo soportados por EXDACLET. Si el nodo es un nodo padre, entonces este puede ser eliminado pero no exportado. Al ser eliminado, se eliminan también todas las hojas asociadas a ese nodo, es decir, eliminar el archivo y todas las tablas que se encuentren asociadas a él.
- *Crear():* Es el encargado de generar toda la estructura interna del árbol, es decir, la creación de la estructura de nodos padre y nodos hijo mediante la interacción con el Sistema Gestor de Base de Datos. Este método es llamado por el método *Tree()* para completar toda la estructura del árbol.
- *Del_F_T():* Es el encargado de ejecutar los procedimientos de eliminación de hojas o nodos del árbol y al igual que el método anterior tiene una alta

interacción con el Sistema Gestor de Base de Datos, de tal forma que no solo se eliminen los datos del árbol sino también físicamente de la herramienta.

Otra de las principales características que tiene la clase **TreePHP** es que a través de ella se cargan las tablas de datos seleccionadas por el analista al hacer doble clic sobre algún nodo hoja del árbol cargado. Cuando este evento ocurre, la sección **Espacio de Trabajo** es cargada con dos pestañas. La primera llamada **DataTable,** en donde se carga la tabla de datos asociada al nodo seleccionado por el analista a través de la clase **GRID** y la segunda, llamada **Statistics**, que contiene un resumen estadístico de la tabla de datos cargada, el cual es presentado mediante un gráfico y una tabla de ponderados, esta sección es trabajada por la clase **Statistics.**

Statistics: Esta clase es la encargada de generar toda la interfaz gráfica para mostrar las estadísticas detalladas de una tabla de datos e interactúa con las clases **Graph** para la generación de la gráfica estadística y con la clase **TableGrid** para la creación de la tabla de resumen estadístico. Esta clase se compone de los siguientes métodos:

- *Statistics():* Es el encargado de la creación del espacio de trabajo dentro del cual serán ubicados, el formulario para la selección de atributos que tendrá la gráfica estadística, la tabla de datos de resumen estadístico de los atributos seleccionados y la gráfica estadística.
- *GenGraph():* Establece una conexión con el Sistema Gestor de Base de Datos mediante una instancia de la clase **MyConn** para extraer información de los atributos seleccionados. Además, es el encargado de la creación de la tabla de datos de resumen estadístico con la información recopilada anteriormente a través de una instancia de la clase **TableGrid.** y finalmente hace un llamado a la clase **Graph** encargada de devolver un mapa de bits que representa la gráfica estadística de la tabla de datos.

TableGrid: Es una clase que extiende de la clase **ext.grid** que a su vez se extiende de la clase **EXTJS,** su objetivo se centra en la creación de una tabla de datos de solo lectura a partir de una tabla HTML básica. Esta tabla HTML es generada a través del método **getInfo** el cual toma todo el contenido de la información extraída por el método *GenGraph()* de la clase **Statistics** y construye la estructura HTML básica requerida para que esta pueda ser interpretada

Graph: Es la clase que crea la gráfica estadística a partir de los parámetros proporcionados por el método *GenGraph()* de la clase **Statistics** y extiende de las clases **jpgraph, jpgraph_bar y jpgraph_line**. Utiliza los siguientes métodos:

- *Graph():* Para iniciar un espacio de trabajo dentro del cual se creará la gráfica y para recolectar los datos que harán parte de la gráfica.
- *SetScale():* Para definir el tamaño de las leyendas.
- *yaxis->scale->SetGrace():* Para establecer una distancia de separación entre el límite superior de la gráfica y el límite superior del espacio de trabajo.
- *xaxis->SetTickLabels():* Inserta las etiquetas o títulos de cada uno de los atributos seleccionados.
- *xaxis->SetLabelAlign():* Define la orientación de las etiquetas.
- *xaxis->SetLabelAngle():* Define el ángulo de inclinación de las etiquetas.
- *xaxis->SetLabelMargin():* Establece un margen entre las etiquetas y el límite inferior del espacio de trabajo.
- *xaxis->SetFont():* Establece el tipo de letra utilizado para las etiquetas.
- *yaxis->SetLabelMargin():*Establece un margen entre las etiquetas y el límite izquierdo del espacio de trabajo.
- *SetShadow():* Asigna una sombra a la gráfica.
- *img->SetMargin():* Define las márgenes con respecto al espacio de trabajo a partir de las cuales será dibujada la gráfica.
- *title->Set():* Establece el título de la gráfica.
- *xaxis->SetTitle():* Establece el título para la coordenada X.
- *yaxis->SetTitle():* Establece el título para la coordenada Y.
- *title->SetFont(),yaxis->title->SetFont() y xaxis->title->SetFont():* Establecen el tipo de letra para los títulos.
- *BarPlot():* Establece que el tipo de gráfica será de barras.
- *LinePlot():*Establece que el tipo de gráfica será de líneas.
- *Add():* Agrega al espacio de trabajo el tipo de gráfica seleccionado.
- *Stroke():* Dibuja la gráfica.

En las figuras 8.27 y 8.28 se muestran el espacio de trabajo con una tabla de datos y el espacio de trabajo con las estadísticas de esa misma tabla de datos.

Figura 8.27 Espacio de trabajo con tabla de datos

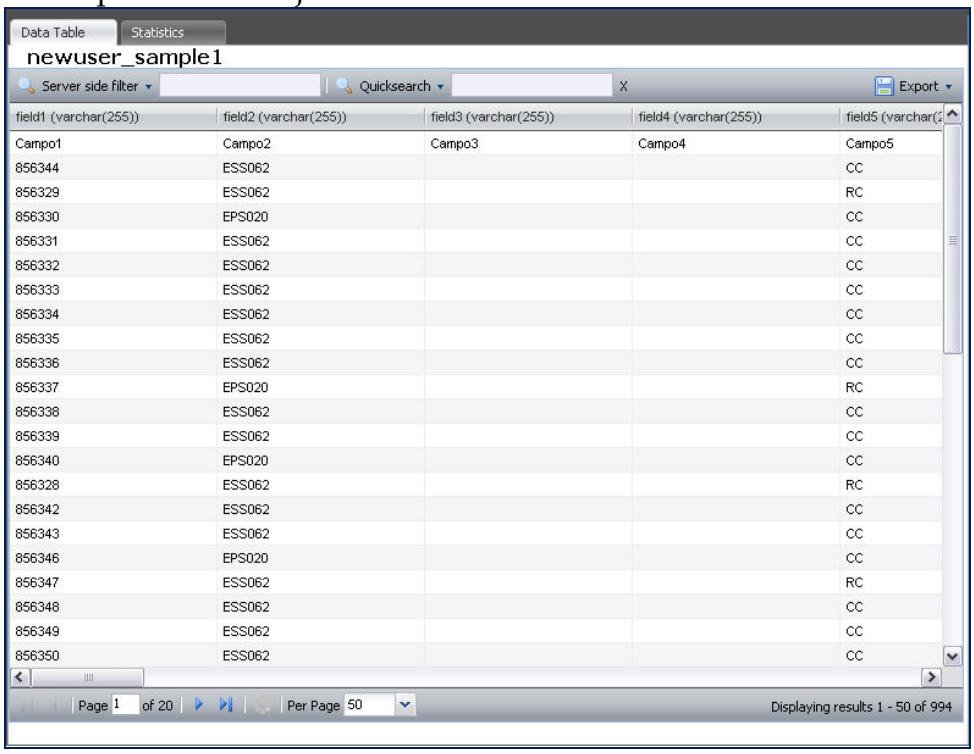
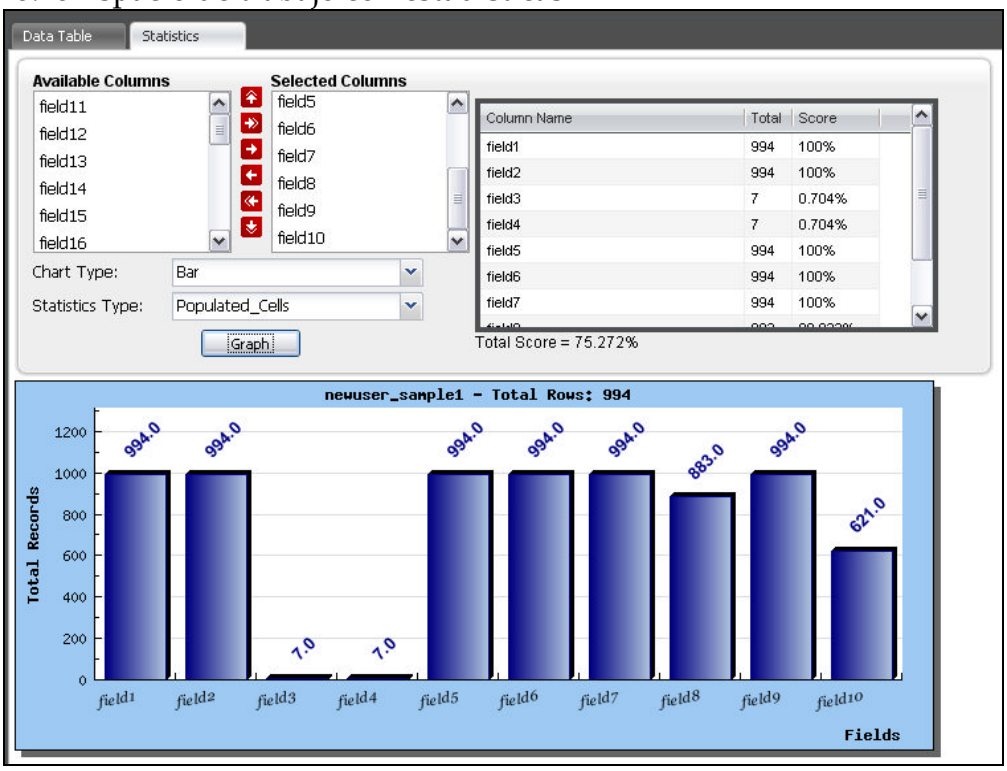


Figura 8.28 Espacio de trabajo con estadísticas



AlgorithmForm: Esta es una de las clases fundamentales dentro de la interfaz gráfica de usuario de EXDACLET, debido a que por ella pasan todos y cada uno de los filtros que el analista desea aplicar sobre las tablas de datos. Se compone de los siguientes métodos.

- *__construct():* Define una instancia del método **Ext.LayoutDialog** de la clase **EXTJS** para mostrar una ventana de dialogo dentro de la cual se construirá todo el formulario. Dentro de esta ventana se generan dos pestañas para mostrar el formulario como tal y para mostrar un contenido de ayuda sobre cómo debe ser manipulado ese formulario.
- *resizeDialog():* Redimensiona el tamaño de la ventana de dialogo mediante el sub-método de la clase **EXTJS, setContentSize**.
- *addCombo():* Permite agregar al formulario una ComboBox y recibe como parámetros un id, la etiqueta, los valores, un mensaje de error y una descripción. Internamente utiliza los sub-métodos de la clase **EXTJS Ext.data.Record.create, Ext.data.SimpleStore y Ext.form.ComboBox** para definir el contenido y la estructura de la ComboBox.
- *setComboColumnStore():*Define un conjunto de registros y un almacenamiento para los valores que tendrá una ComboBox ubicada dentro de una columna mediante los sub-métodos de la clase **EXTJS Ext.data.Record.create, Ext.data.SimpleStore.**
- *setComboColumnCont():* Establece la estructura de la Combobox y asigna el contenido de la misma definido en el método *setComboColumnStore().*
- *setTextFieldColumn():* Permite crear una caja de texto ubicada dentro de una columna y recibe como parámetros un id, una etiqueta, una expresión regular si lo necesita para validación, un mensaje de error en caso de no cumplir con la expresión regular, una bandera para definir si se permiten campos vacios, un listado con los valores que no se permiten, un mensaje de error para esta última situación, la cantidad de caracteres permitidos y una descripción. Internamente utiliza al sub-método de la clase **EXTJS Ext.form.TextField**.
- *setColumn():* Define una columna dentro del formulario, en ella se cargarán las cajas de texto o ComboBox definidas por los métodos *setComboColumnCont(). y setTextFieldColumn().*
- *addNumberField():* Permite agregar a un formulario una caja de texto que solo acepta valores numéricos, recibe como parámetros un id, una etiqueta, un valor a mostrar por defecto si lo tiene, una expresión regular si lo necesita para validación, un mensaje de error en caso de no cumplir con la expresión regular, una bandera para definir si se permiten campos vacios, una bandera para definir si el valor escrito en su interior aceptará o no decimales, la cantidad de caracteres permitidos y una descripción. Internamente utiliza al sub-método de

la clase **EXTJS Ext.form.NumberField** para definir la estructura de la caja de texto.

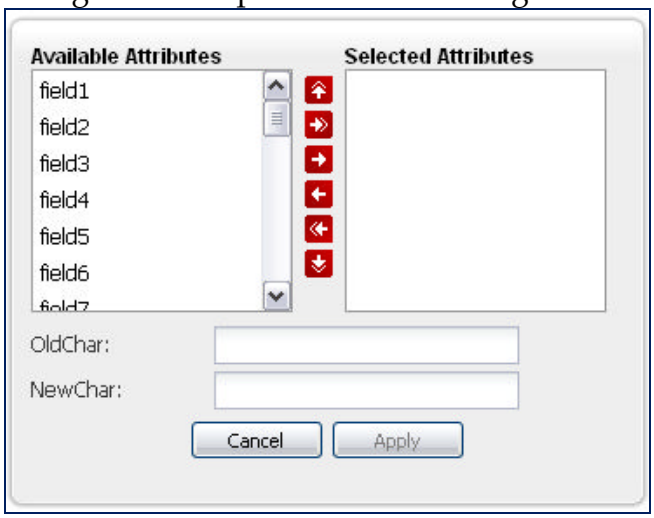
- *addTextField():* Permite agregar una caja de texto a un formulario. Recibe como parámetros los mismos valores que recibe el método *setTextFieldColum.*
- *addEditor():* Permite agregar una TextArea utilizada por el filtro **Expert Rule Editor**, internamente utiliza el sub-método de la clase **EXTJS Ext.form.TextArea**.
- *addMultiselect():* Permite agregar dos listas de selección dentro de las cuales pueden ser movidos ítems de una a otra. Recibe como parámetros una etiqueta general, una etiqueta para la lista izquierda, una etiqueta para la lista derecha, los valores de la lista izquierda, los valores de la lista derecha, el ancho, el alto y una descripción. Internamente interactúa con la clase **Ext.ux.Multiselect** que a su vez se deriva de la clase **EXTJS** y utiliza el sub-método **Ext.ux.ItemSelector** para establecer la estructura y definir el contenido de las mismas.
- *addMsgDiv():* Permite agregar un espacio al final del formulario para la generación de mensajes de error que puedan ocurrir durante la interacción con el mismo.
- *setContainer():* Permite definir un espacio dentro del formulario sobre el cual se creará una cantidad determinada de Campos de texto, número, Selectbox o ComboBox.
- *ComboEvent():* Permite recargar el contenido de una ComboBox dependiendo del valor que se encuentre seleccionado en otra ComboBox mediante la manipulación de sus contenidos. Interactúa con el sub-método **findField** de la clase **EXTJS** para establecer la ComboBox base y la ComboBox destino.
- *getUploadForm():* Permite generar un formulario para la carga de archivos, incluyendo en su interior un campo de carga en el cual se mostrará la ruta del archivo a cargar y dos botones para seleccionar el archivo origen e importarlo respectivamente. Internamente interactúa con la clase **writeFile** encargada de verificar la consistencia del archivo y trasladarlo de la máquina cliente al servidor para su posterior proceso de importación a la herramienta mediante el Sistema Gestor de Base de Datos e interactúa también con los sub-métodos **Ext.form.Form, Ext.form.TextField, addButton, render, submitForm y checkMessage** de la clase **EXTJS** para validar y garantizar el correcto funcionamiento y desempeño del proceso de pre-carga de archivos.
- *getLoginForm():*Permite generar un formulario para el registro e ingreso de un usuario a la herramienta. Utiliza internamente los métodos **Ext.form.Form, Ext.form.TextField y addButton**.
- *getForm():* Este método es el encargado de generar todo el código JavaScript necesario para la construcción de un formulario. Para poder utilizar este método siempre debe haber al menos un previo llamado a cualquiera de los métodos *addNumberField(), addTextField(), addCombo() o addMultiselect().* Utiliza

internamente los sub-métodos de la clase **EXTJS Ext.form.Form, addButton y render.**

- *getFormRule():* Permite construir el formulario para el método *addEditor().* Utiliza internamente los mismos métodos de la clase **EXTJS** mencionados en el método getForm().
- *getJS():* Es el método encargado de devolver el código JavaScript a ser ejecutado para construir la ventana de dialogo en la cual estará inmerso el formulario.
- *setFormWidthGen():* Permite establecer el ancho del formulario.
- *getFormStruct():* Retorna el código HTML encargado de construir la interfaz del formulario.
- *getFormStructTwo():*Retorna el código HTML encargado de armar la interfaz del formulario.
- *getHtmlHelp():* Retorna el código HTML para la definición de la estructura del contenido de ayuda del formulario.
- *setHtmlHelp():* Establece las características para la estructura del contenido de ayuda del formulario.
- *setDescription():* Extrae todas las descripciones definidas en cada elemento del formulario y las inserta dentro del contenido de ayuda.

En la figura 8.29 se observa un ejemplo de un formulario generado a través de esta clase.

Figura 8.29 Formulario generado a partir de la clase AlgorithmForm



GRID: Esta clase es la encargada de generar la tabla de datos de un archivo seleccionado por el analista dentro del **Espacio de Trabajo.** Esta tabla de datos es una tabla editable y sobre ella se pueden aplicar filtros de búsqueda local y remota. Una búsqueda local se refiere a buscar valores dentro de uno o más atributos de los resultados que se estén presentando en pantalla mientras que una búsqueda remota hace referencia a buscar un valor dentro de un atributo determinado en toda la tabla de datos. Los métodos que componen esta clase son los siguientes:

- *grid():* Este método es el encargado de generar toda la estructura de la tabla de datos que será cargada. Esta estructura viene definida por los términos **jsonreader, columnmodel y datastore** los cuales son la base para la creación de dicha tabla. Internamente crea una instancia de la clase **DATA** encargada de la construcción del código JSON que contiene todos los valores que componen la tabla de datos (filas y columnas). Además, este método interactúa con los métodos de la clase **EXTJS, Ext.menu.CheckItem**, **layout.getRegion, Ext.data.Store, Ext.data.ScriptTagProxy, Ext.data.JsonReader, Ext.grid.GridEditor, Ext.form, Ext.grid.ColumnModel, Ext.grid.EditorGrid, Ext.grid.RowSelectionModel, Ext.menu.Menu, Ext.PagingToolbar, Ext.data.SimpleStore, Ext.form.ComboBox, grid.getView, y grid.getHeaderPanel.** Todos ellos encargados de construir físicamente la tabla de datos en el cliente.
- *Grid_Update():* Este es el método encargado de actualizar la tabla de datos una vez ha sido editada, tanto para el cliente como para el servidor interactuando también con el Sistema Gestor de Base de Datos.
- *Grid_DeleteGroup():* Este método es el encargado de eliminar un conjunto de filas de la tabla de datos y al igual que el método *Grid_Update()* actualiza la tabla de datos para el cliente y para el servidor.
- *Grid_Delete() y Grid_Delete_B():* Elimina una fila de la tabla de datos y actualiza para el cliente y el servidor.

StandardGrid: Esta es la clase encargada de generar todas las tablas temporales que se utilizan en los filtros de selección y en algunos de limpieza y transformación y se compone de los siguientes métodos:

- *__construct():* Inicializa una instancia para la creación de la tabla de datos temporal.
- *Define_grid():* Es el método encargado de construir toda la estructura y el contenido de la tabla de datos. Dependiendo del tipo de filtro que sea utilizado,

este método incluye o excluye ciertas opciones que pueden ser adheridas a la tabla. Algunas de estas opciones son: Si los valores de todos o algunos atributos pueden ser editables o no, Si aparece dentro de un menú contextual utilizado en la tabla opciones de eliminación u omisión de registros, si se agregan opciones para el reemplazo múltiple de valores. Internamente utiliza los mismos métodos de la clase **EXTJS** que se ejecutan en el método *grid()* de la clase **GRID**.

- *setForm():* Crea un formulario para realizar tareas de análisis o ejecutar un procedimiento para guardar la tabla de datos temporales o simplemente cerrar la ventana de dialogo y por ende la tabla de datos generada en su interior.
- *getForm():* Retorna todo el código JavaScript generado por el método *setForm()* y lo ejecuta.
- *addReplace():* Permite agregar a la tabla de datos las opciones de reemplazo individual o múltiple de valores que se encuentran en la misma.
- *UseCorrections():* Es un método utilizado por el filtro **Email Cleaner** en el cual se actualiza un atributo de la tabla de datos original en base a la tabla de datos temporal.
- *Grid_ReplaceCol():* Método utilizado para transformar una serie de valores, es ejecutado cuando en la tabla de datos temporal es agregada la opción de reemplazo múltiple.
- *Dup_tmp_tab():* Método utilizado a la hora de guardar la tabla de datos temporal. Es el encargado de generar la interfaz gráfica en la cual se solicita el nuevo nombre de la tabla que será guardada.
- *Create_dup_tmp():* Es el paso siguiente a la ejecución del método *Dup_tmp_tab(),* en este método se procede a la creación de la nueva tabla de datos mediante la interacción con el Sistema Gestor de Base de Datos.

Un ejemplo de la creación de una tabla de datos a partir de la clase **StandardGrid** se muestra en la figura 8.22 **(**Tabla para tratamiento JaroWinkler, DoubleMetaphone, Levenshtein**).**

AlgDialog: Esta clase funciona en conjunto con la clase **StandardGrid** puesto que es la encargada de crear la ventana de dialogo y la estructura gráfica dentro de la cual será generada la tabla de datos temporal. Los métodos que la componen son los siguientes:

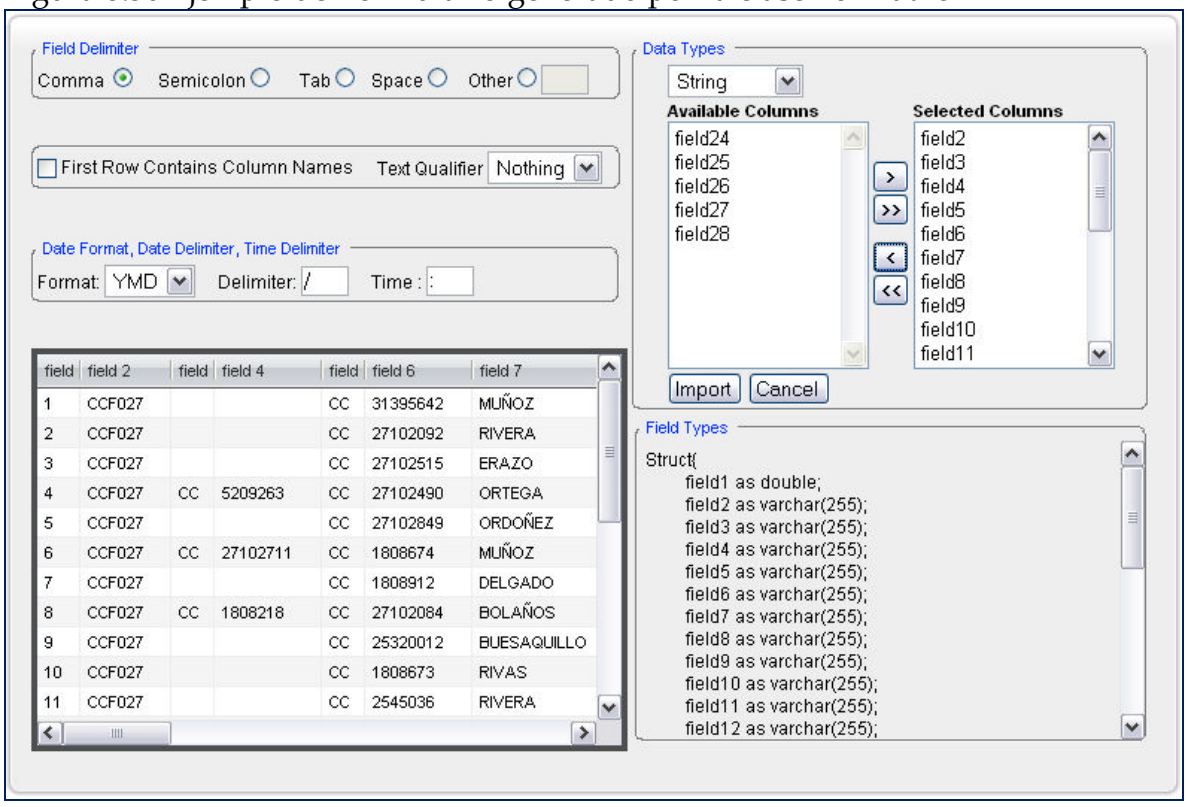
- *__construct():* Define la ventana de dialogo que será mostrada, interactúa con el método **Ext.LayoutDialog** de la clase **EXTJS** para lograr este objetivo.
- *getDLG():* Ejecuta el código generado en el método *__construct().*
- *getFormStruct():* Establece el espacio dentro del cual será cargada la tabla de datos temporal.
- *setHtmlHelpT():* Define la estructura del contenido de ayuda para la tabla temporal y los procesos que pueden ser realizados sobre ella.
- *getHtmlHelpT():* Ejecuta el código HTML generado por el método *setHtmlHelpT().*

FormatForm: Es la clase que genera toda la estructura del formulario para los procesos de pre-carga de archivos de tipo CSV y XLS. Los métodos utilizados son:

- *__construct():* Define la ventana de dialogo sobre la cual se generará el formulario.
- *setForm():* Crea la estructura del formulario mediante código HTML.
- *getForm():* Ejecuta el código generado por setForm().
- *setRadioCsv():* Establece las opciones de radio button que se tendrán en cuenta dependiendo del tipo de formato de archivo que se esté utilizando.
- *setTextQualifier():* Estable la opción para la selección del tipo de delimitador de cadena que caracteriza al archivo.
- *setDecimalSymbol():* permite crear la estructura para los delimitadores del símbolo decimal.
- *setTableResume():* Permite generar un espacio dentro del formulario para la carga de un contenido de resumen de la tabla de datos que se asocia al archivo que se esté importando y al cual se le este definiendo su estructura.
- *setFieldResume():* Permite generar un espacio dentro del formulario para la carga de un contenido en el cual se muestra la estructura de los tipos de datos que se están asociando a cada atributo de la tabla de datos.
- *getJS():* Ejecuta el código generado por el método *__construct()* para la creación de la ventana de dialogo.
- *getFormStruct():*Establece el espacio dentro del cual será cargada la tabla de datos temporal.
- *setHtmlHelpF():* Define la estructura del contenido de ayuda para la tabla temporal y los procesos que pueden ser realizados sobre ella.
- *getHtmlHelpF():* Ejecuta el código HTML generado por el método *setHtmlHelpF().*

En la figura 8.30 se muestra un ejemplo del formulario generado mediante la clase **FormatForm**, en ella se muestran las opciones generadas y requeridas al momento de importar un archivo de tipo CSV.

Figura 8.30 Ejemplo del formulario generado por la clase FormatForm



Dentro del paquete GUI es importante la interacción constante con la biblioteca EXTJS a través de su clase **EXTJS** y utilizada para la generación de las estadísticas de cada tabla de datos.

8.3.6 Paquete Connection Este paquete es uno de los pilares fundamentales dentro de EXDACLET, debido a que absolutamente todos los filtros que reúne la herramienta y todos los procedimientos que ellos ejecutan tienen que interactuar constantemente con el Sistema Gestor de Base de Datos. Se compone de la siguiente clase con sus métodos y descripción de los mismos.

MyConn: esta clase es la encargada de establecer todas las conexiones entre la herramienta y el Sistema Gestor de Base de Datos MYSQL, es la más importante debido a que todas las acciones que realice un usuario tienen que ver con el acceso, selección y transformación de diferentes tablas de datos que deben ser almacenadas de una forma persistente, si esta clase no cumple su objetivo toda la herramienta se vuelve obsoleta. Los métodos que llevan a cabo todos los procesos de conexión entre la herramienta y el Sistema Gestor de Base de Datos además de los procesos de ejecución de consultas para selección, modificación o eliminación de tablas y registros se muestran a continuación:

- *abrir():* Es el método encargado de establecer la conexión con el Sistema Gestor de Base de Datos mediante parámetros como el **host, nombre de usuario, contraseña y base de datos** destino de la conexión.
- *Cerrar():* Es el método encargado de dar por terminada la conexión con el Sistema Gestor de Base de Datos. Este método debe ser llamado tantas veces como sea llamado el método abrir(). Después de haberse ejecutado las consultas necesarias.
- *Query():* Es el método que ejecuta la consulta que le es pasada como parámetro, devuelve los resultados que genere la misma. Como exigencia se encuentra que debe estar establecida una conexión con el Sistema Gestor de Base de Datos.
- *Fetch():* Es el método encargado de extraer una a una las filas resultado de la consulta hecha mediante el método *Query*() y al igual que él, es necesario tener establecida una conexión con el Sistema Gestor de Base de Datos.
- *TotalRows():* Este método devuelve el número de filas resultantes al ser aplicada una consulta mediante el método *Query*(). Requiere tener establecida una conexión con el Sistema Gestor de Base de Datos.

En conjunto con los anteriores paquetes mencionados, cabe destacar la existencia de un paquete el cual no se encuentra directamente asociado con la herramienta pero que es el encargado de la administración de usuarios que pueden acceder a la misma. Este paquete es llamado **UserAdmin** y se compone de las siguientes clases:

UserTree: Esta clase es la encargada de generar el árbol mediante el cual se muestran todos los usuarios registrados en la herramienta en conjunto con todos los archivos que cada uno tiene cargado y las tablas asociadas a cada archivo. La principal característica que tiene esta clase es la generación de un menú contextual

en el cual se pueden eliminar tablas, archivos o usuarios según sea la necesidad del administrador de la herramienta. Esta clase se compone de los siguientes métodos:

- *userTree():* Dentro del cual se crea toda la estructura y se inserta el contenido del árbol que será cargado.
- *Crear():* Encargado de generar todo el contenido del árbol que será insertado por el método *userTree().*
- *Del_F_T():* Es el encargado de actualizar el árbol cuando un nodo u hoja del mismo han sido eliminados e interactúa con el Sistema Gestor de Base de Datos para ejecutar los procedimientos requeridos al ser eliminada una tabla, un archivo o un usuario.
- *Usermodtree():* Este método se encarga de crear el árbol utilizado al momento de modificar usuarios.

Users: Esta clase es la encargada de generar todas las interfaces de usuario y procedimientos requeridos para la administración básica de usuarios. Se compone de los siguientes métodos con sus descripciones:

- *App():* Crea un efecto de carga antes de ingresar a la interfaz gráfica de administración.
- *usersInit():* Genera y ejecuta el código HTML para crear la distribución física de las opciones del proceso de administración.
- *initAddUser():* Define el espacio dentro del cual será cargado el formulario para la creación de un usuario.
- *createAccount():* Es el método encargado de ejecutar la consulta para la creación de un usuario que pueda ingresar a trabajar con la herramienta. Esto se logra mediante la interacción con el Sistema Gestor de Base de Datos y la clase **BD**.
- *initModUser():* Este método permite la generación del espacio en el cual se cargará el formulario para la modificación de usuarios.
- *modifyAccount():* Es el método encargado de ejecutar la consulta para la actualización de los datos de un usuario que por alguna razón olvidó su contraseña o nombre de usuario. Esto se logra mediante la interacción con el Sistema Gestor de Base de Datos y la clase **BD**.

UserForm: Esta es la clase que permite la creación de los formularios para la inserción y modificación de usuarios. Contiene los siguientes métodos:

- *__construct():* Inicializa los valores para la creación del formulario.
- *addUserForm():* Genera todo el código JavaScript a ser ejecutado que contiene todo el formulario para la creación de usuarios.
- *modUserForm():* Genera todo el código JavaScript a ser ejecutado que contiene todo el formulario para la modificación y/o actualización de usuarios.
- *setFormWidthGen():* Establece el ancho que tendrá el formulario generado.
- *getFormStruct():*Retorna el código HTML encargado de construir la interfaz del formulario.

En las figuras 8.31 y 8.32 se muestran los formularios generados para la adición y modificación de usuarios respectivamente.

Figura 8.31 Formulario para la adición de usuarios

Figura 8.32 Formulario para la modificación de usuarios

9. PRUEBAS Y RESULTADOS

Las pruebas y evaluación del rendimiento de los principales algoritmos implementados en EXDACLET, se realizaron en un computador con un procesador DualCore Intel Core 2 Duo, a 2000 MHz, con una memoria RAM de 3Gb DDR2 SDRAM, con disco duro SATA-II de 250 GB y 7200 RPM.

Para la realización de las pruebas se utilizaron conjuntos de datos reales pertenecientes a los usuarios beneficiarios del régimen subsidiado en salud del departamento de Nariño que hacen parte de la BDUA (Base de Datos Única de Afiliados) del sistema de salud.de Colombia, y a los estudiantes de la Universidad de Nariño. Además se utilizaron repositorios de datos sintéticos generados de la página http://www.datgen.com/, este es un sitio web en el cual reside una aplicación encargada de generar conjuntos de datos para ayudar con el análisis empírico de otros programas y tiene un enfoque especial en los procesos KDD. Las características de los conjuntos de datos reales son las siguientes:

- idsn26112005.csv. Conjunto de datos con las características personales, características familiares, fecha de ingreso del sistema, datos SISBEN, tipo de contratación, fecha del contrato y tipo del servicio de salud, de 817692 usuarios del SISBEN con 28 atributos.
- Udenar.csv. Contiene datos relacionados con la información familiar y académica de 20388 estudiantes de la Universidad de Nariño de todas las sedes del departamento de Nariño (Colombia) con 23 atributos.

Para generar los conjuntos de datos sintéticos se tuvo en cuenta el número de atributos, el número de reglas, el dominio de variación de los valores que contendrá cada atributo y el número de registros a generar (1-1000000). La nomenclatura utilizada tanto para el conjunto de datos reales como para el conjunto de datos sintéticos es la siguiente:

DB9TC99.

En donde

- Db: Identificador del conjunto de datos.
- 9: Cantidad en miles de registros.
- TC: Identificador del número de atributos.
- 99: Cantidad de Atributos.

En la tabla 9.1 se muestran los conjuntos de datos extraídos de los archivos idsn26112005.csv, udenar.csv y los conjuntos de datos generados de la página http://www.datgen.com aplicados durante los procesos de prueba.

Tabla 9.1 Conjuntos de datos de prueba

ARCHIVO ORIGEN	NOMENCLATURA	FILAS	COLUMNAS
idsn26112005.csv	DB1TC28	1000	28
	DB2TC28	2000	28
	DB3TC28	3000	28
	DB4TC28	4000	28
	DB5TC28	5000	28
	DB10TC28	10000	28
	DB15TC28	15000	28
	DB20TC28	20000	28
	DB30TC28	30000	28
	DB40TC28	40000	28
	DB50TC28	50000	28
	DB80TC28	80000	28
	DB100TC28	100000	28
	DB150TC28	150000	28
	DB200TC28	200000	28
	DB250TC28	250000	28
UDENAR.CSV	DB20388TC23	20388	23
http://www.datgen.com	DB1TC11	1000	11
	DB10TC11	10000	11
	DB100TC11	100000	11
	DB1000TC11	1000000	11
	DB1TC21	1000	21
	DB10TC21	10000	21
SAMPLEFILE20000.CSV	DB20TC12	19461	12
SAMPLEFILE8000.CSV	DB8TC12	8000	12

9.1 PRUEBAS CON FILTROS DE SELECCIÓN

Las pruebas realizadas con la herramienta EXDACLET, consisten en medir el tiempo de ejecución de un determinado filtro de selección utilizando el repositorio idsn26112005.csv.

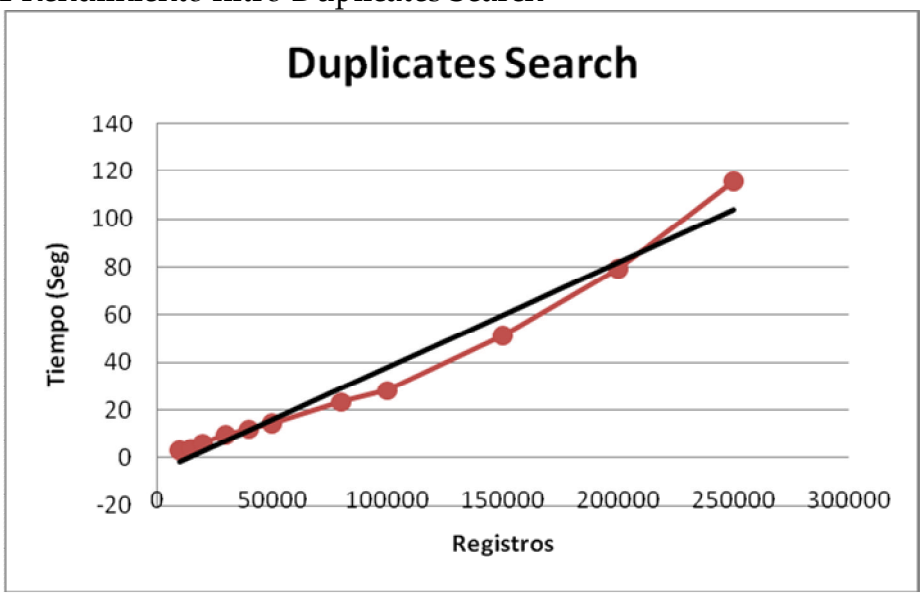
9.1.1 Filtro Duplicates Search Se evaluó el rendimiento del algoritmo Duplicates Search aplicando el filtro sobre los repositorios derivados del repositorio idsn26112005.csv, generando cada uno una cantidad de duplicados diferentes. Teniendo en cuenta que el objetivo del filtro es la búsqueda de registros duplicados, fueron seleccionados para la búsqueda 4 atributos que se componen de nombres y apellidos.

Los resultados de la aplicación del filtro se pueden apreciar en la tabla 9.2, y en la figura 9.1 se muestra el comportamiento del algoritmo con respecto al tiempo y a la cantidad de registros de cada repositorio.

Tabla 9.2 Resultados evaluación del filtro Duplicates Search

Repositorio	Registros	Tiempo (seg)	Duplicados Encontrados
DB10TC28	10000	3.13	0
DB15TC28	15000	4.06	4
DB20TC28	20000	6.2	4
DB30TC28	30000	9.63	10
DB40TC28	40000	11.98	16
DB50TC28	50000	14.45	32
DB80TC28	80000	23.24	123
DB100TC28	100000	28.12	167
DB150TC28	150000	51.4	438
DB200TC28	200000	79.33	666
DB250TC28	250000	115.65	1013

Figura 9.1 Rendimiento filtro Duplicates Search



Teniendo en cuenta los resultados generados y que la tendencia de los valores de la gráfica 9.1 es lineal, se puede afirmar que el tiempo de ejecución de este filtro es proporcional a la cantidad de registros evaluados y por lo tanto es un filtro escalable. También es importante destacar dos situaciones cuando este filtro es aplicado:

- La cantidad de duplicados encontrados puede variar dependiendo de cómo esté conformada la tabla de datos y de la cantidad de atributos que se seleccionen.
- El tiempo de ejecución sobre una misma tabla puede variar dependiendo de la cantidad de atributos que se seleccionen.

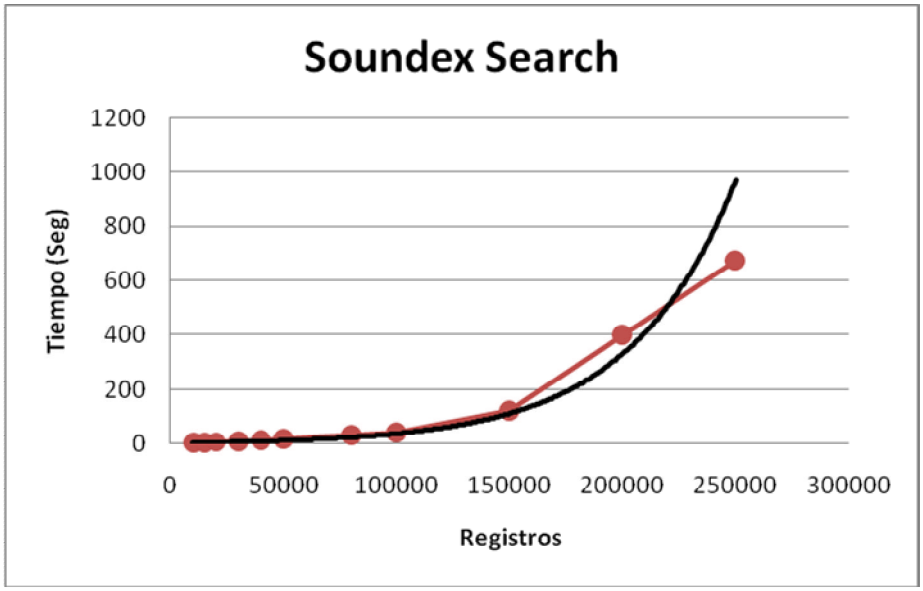
9.1.2 Filtro Soundex Search Este filtro tiene como objetivo la búsqueda de posibles duplicados en base a su pronunciación.

Para la búsqueda de duplicados se tuvieron como referencia los mismos atributos utilizados en el filtro Duplicates Search. En la tabla 9.3 se presenta el rendimiento del algoritmo en base al tiempo y la cantidad de posibles duplicados encontrados en diferentes repositorios, y en la figura 9.2 se muestra la interpretación gráfica de dichos resultados.

Tabla 9.3 Resultados evaluación del filtro Soundex Search

Repositorio	Registros	Tiempo (seg)	Duplicados Encontrados
DB10TC28	10000	3.43	2
DB15TC28	15000	5.02	10
DB20TC28	20000	6.32	20
DB30TC28	30000	9.54	38
DB40TC28	40000	13.22	72
DB50TC28	50000	17.01	110
DB80TC28	80000	31.58	329
DB100TC28	100000	43.36	484
DB150TC28	150000	122.45	1104
DB200TC28	200000	397.88	1827
DB250TC28	250000	672.89	2852

Figura 9.2 Rendimiento filtro Soundex Search



Teniendo en cuenta los resultados que se muestran en la tabla 9.3 y la línea de tendencia que acompaña los datos de la figura 9.2, es posible afirmar que el comportamiento del algoritmo Soundex es exponencial, por esta razón, se sugiere la aplicación del filtro sobre conjuntos de datos que no superen los 200000 registros, ya que hasta ese punto se puede decir que el comportamiento del filtro con respecto al tiempo es estable, como se detalla en el rango que va desde 0 hasta 150000 registros. Y al igual que en el filtro Duplicates Search, el tiempo de ejecución de este filtro depende completamente de la cantidad de atributos

seleccionados y de los valores que estos contengan, mientras que la cantidad de posibles duplicados generados depende solamente de los valores que contengan los atributos seleccionados. Teniendo en cuenta los resultados anteriores:

- El filtro Duplicates Search es eficaz y eficiente a la hora de la "*búsqueda de duplicados*" frente al filtro Soundex Search, dado que este ultimo realiza un proceso intermedio de estandarización fonética de las palabras lo cual implica una utilización mayor de recursos tanto de procesamiento como de tiempo.
- Si se trata de la búsqueda de "*duplicados y posibles duplicados*", entonces el filtro Soundex Search es realmente el más eficiente puesto que es ese proceso intermedio de estandarización fonética el que acopla las palabras a un campo de búsqueda mucho más reducido, ayudando así a encontrar muchas posibles variaciones con respecto a la estructura de una o más palabras.

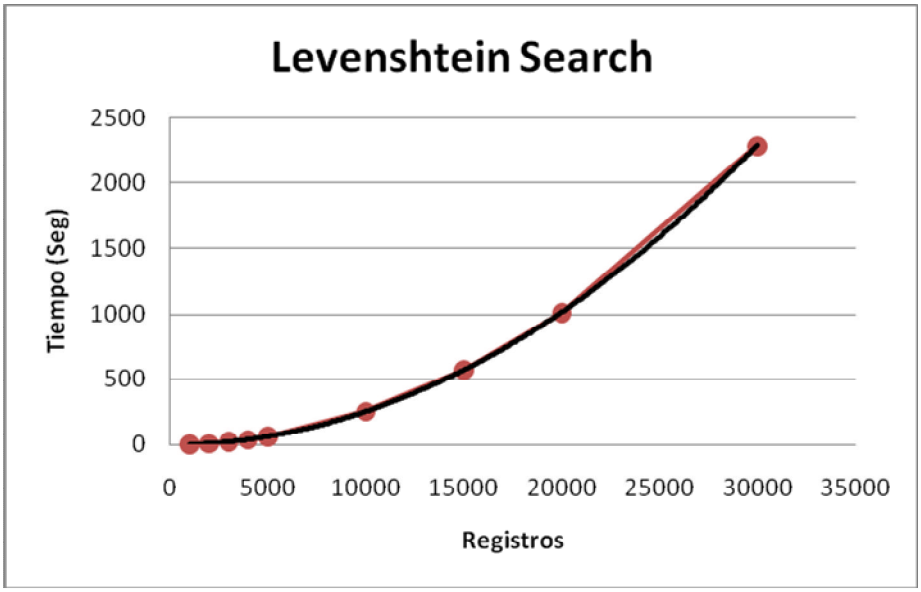
9.1.3 Filtro Levenshtein Search Este filtro se enfoca en la búsqueda de posibles duplicados teniendo en cuenta la métrica de las palabras.

Para la búsqueda de posibles duplicados se tuvieron como referencia los mismos atributos utilizados en los filtros Duplicates Search y Soundex Search, en la tabla 9.4 se presenta el rendimiento del algoritmo en base al tiempo y la cantidad de posibles duplicados encontrados en diferentes repositorios, y en la figura 9.3 se presentan gráficamente los resultados de la tabla 9.4.

Tabla 9.4 Resultados evaluación filtro Levenshtein

Repositorio	Registros	Tiempo (seg)	Posibles Duplicados
DB30TC28	30000	2287.43	7024
DB20TC28	20000	1012.93	3274
DB15TC28	15000	572.19	2011
DB10TC28	10000	255.38	898
DB5TC28	5000	64.64	260
DB4TC28	4000	40.65	164
DB3TC28	3000	21.91	107
DB2TC28	2000	9.9	49
DB1TC28	1000	2.53	3

Figura 9.3 Rendimiento filtro Levenshtein Search



En la figura 9.3 se puede observar que la línea de tendencia tiene un comportamiento potencial y se acopla perfectamente con los datos obtenidos en las pruebas. Teniendo en cuenta dichos datos, se observa que a medida que se aumenta la cantidad de registros, el comportamiento del filtro con respecto al tiempo tiende a volverlo ineficiente, sin afectar en ningún momento la calidad de los resultados. Por esta razón se sugiere aplicar el filtro sobre conjuntos de datos que no superen los 25000 registros puesto que de hacer lo contrario, se vería afectado el rendimiento de otros filtros que podrían ser aplicados en el mismo momento por otros usuarios.

Este filtro a pesar de su bajo rendimiento con conjuntos de datos muy grandes, tiene un nivel de precisión mucho mayor que el filtro Soundex Search al momento de buscar posibles duplicados. Otro de los puntos importantes que se deben tener en cuenta y que pueden afectar la ejecución de este filtro es la cantidad de atributos que se tengan en cuenta y los valores que estos contienen.

9.2 PRUEBAS CON FILTROS DE TRANSFORMACIÓN

Entre algunos de los principales filtros de transformación implementados en EXDACLET y cuyo objetivo se centra en preparar los datos para que estos sean coherentes, completos y ayuden a agilizar posteriores aplicaciones de algoritmos

de Minería de Datos, se encuentran Non-Printable Character Search, Char Replace, Binarize y Encode/Decode.

Sobre estos filtros se realizaron pruebas que permiten verificar su funcionamiento y a la vez demostrar que son acoplables a los requerimientos que se tengan por parte del usuario. Los resultados obtenidos fueron los siguientes:

9.2.1 Filtro Non-Printable Character Search En la figura 9.4 se aprecia una tabla de datos de prueba constituida por algunos caracteres considerados no imprimibles por no hacer parte del grupo de caracteres alfa-numérico.

Durante el proceso de aplicación de este filtro, el usuario selecciona los valores por los que serán reemplazados tales caracteres como se muestra en la figura 9.5 para finalmente mostrar el resultado de la transformación como se observa en la figura 9.6.

Figura 9.4 Tabla de datos con caracteres no imprimibles

field1 (varchar(255))	field2 (varchar(255))	field3 (varchar(255))	field4 (varchar(255))	field5 (varchar
user¥1	admiün	goa°l	Rende§R	mondÄy
Rende§R	admiün	user¥1	mondÄy	?DD
user¥1	admiün	goa°l	Rende§R	mondÄy
Rende§R	admiün	user¥1	mondÄy	?DD
user¥1	admiün	goa°l	Rende§R	mondÄy
Rende§R	admiün	user¥1	mondÄy	?DD

Figura 9.5 Tabla de corrección de caracteres no imprimibles

Caracteres no Imprimibles	Reemplazar Por
¥	Ñ
§	S
ü	U
Ä	A

Figura 9.6 Resultado de la aplicación del filtro non-printable character search

field1 (varchar(255))	field2 (varchar(255))	field3 (varchar(255)	field4 (varchar(255))	field5 (varchar
userÑ1	admiUn	goa°l	RendeSR	mondAy
RendeSR	admiUn	userÑ1	mondAy	?DD
userÑ1	admiUn	goa°l	RendeSR	mondAy
RendeSR	admiUn	userÑ1	mondAy	?DD
userÑ1	admiUn	goa°l	RendeSR	mondAy
RendeSR	admiUn	userÑ1	mondAy	?DD

En la figura 9.6 se observa que aún hay caracteres que se podrían considerar como no imprimibles, pero estos caracteres se encuentran dentro del rango de caracteres alfanuméricos, por lo tanto para su tratamiento se hace necesaria la aplicación del filtro Char Replace como se muestra en la figura 9.7 para tratar estos valores.

Figura 9.7 Tratamiento de algunos valores que podrían ser considerados no imprimibles

field1 (varchar(255))	field2 (varchar(255))	field3 (varchar(255))	field4 (varchar(255))	field5 (varchar
userÑ1	admiUn	goaOl	RendeSR	mondAy
RendeSR	admiUn	userÑ1	mondAy	_DD
userÑ1	admiUn	goaOl	RendeSR	mondAy
RendeSR	admiUn	userÑ1	mondAy	_DD
userÑ1	admiUn	goaOl	RendeSR	mondAy
RendeSR	admiUn	userÑ1	mondAy	_DD

En la figura 9.7 se observa que los caracteres considerados no imprimibles como "º" y "?" fueron reemplazados por los valores "O" y "_" respectivamente. De esta manera se demuestra que tanto el filtro Non-Printable Character Search como el filtro Char Replace son muy eficaces y eficientes en lo que respecta al proceso de transformación y preparación de los datos; además pueden ser aplicados sobre conjuntos de datos de cualquier tamaño.

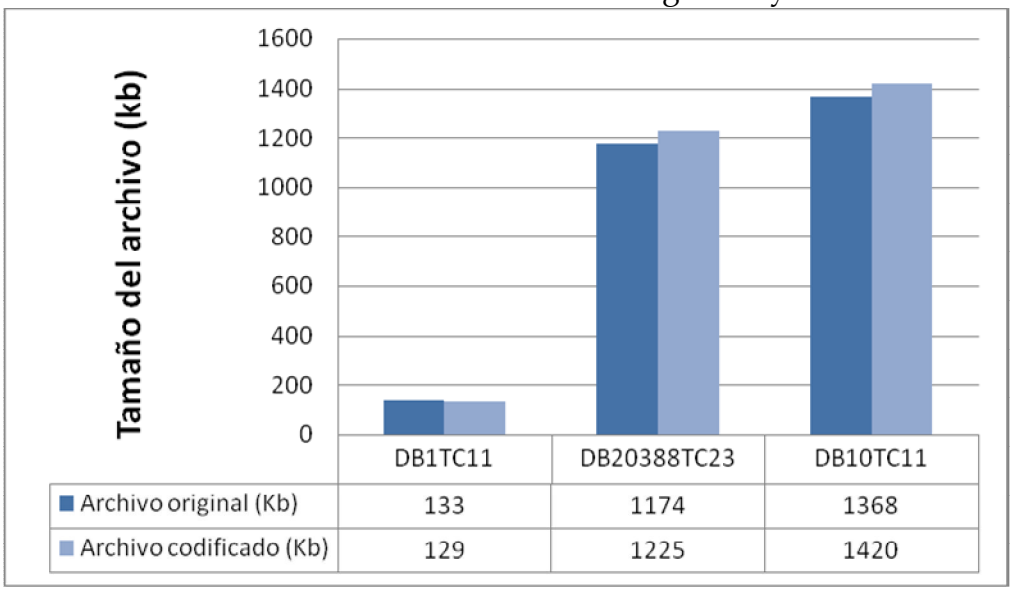
9.2.2 Filtro Encode/Decode Para este filtro fueron utilizados los repositorios sintéticos generados a partir de la página Web http://www.datgen.com. Los resultados se muestran en la tabla 9.5

Tabla 9.5 Detalle de los resultados entre un archivo original y un codificado

Repositorio	Atributos codificados	Archivo original (Kb)	Archivo codificado (Kb)	Valores Diferentes	Tiempo (Seg)
DB1TC11	28	133	129	4610	148.83
DB20388TC23	23	1174	1225	84	345.64
DB10TC11	28	1368	1420	21375	1620.12

En la figura 9.8 se presenta una comparación entre los archivos originales y sus respectivos archivos codificados.

Figura 9.8 Relación entre los tamaños de archivos originales y codificados



De la gráfica anterior es posible deducir que:

La codificación y decodificación son un proceso lento puesto que a pesar de que la construcción y carga de la tabla de diccionario de datos asociada a cada tabla codificada tiene un tiempo de creación supremamente pequeño, el proceso de reemplazo de cada uno de los distintos valores que componen la tabla de datos por el nuevo código asignado, consume una gran cantidad de recursos y por lo tanto una gran cantidad de tiempo. Pero si este procedimiento fuese comparado con una codificación manual, entonces fácilmente se puede deducir que es un proceso muy

veloz, confiable y claro puesto que el usuario estará soportado por un diccionario de datos completo y de fácil acceso ayudándolo a interpretar los valores que ahí se encuentren o incluso, puede decodificar los valores que se encuentran almacenados en la tabla de datos aplicando el proceso inverso.

El tiempo de ejecución de este filtro puede variar dependiendo de la carga de transacciones que se realicen sobre el servidor, sin afectar el resultado que este pueda generar.

Las diferencias entre el tamaño en Bytes del archivo de origen y el archivo codificado, son relativos a los valores que estos contengan, por lo tanto no siempre un archivo codificado es de igual o de menor tamaño que su respectivo archivo original. La diferencia, se centra en el hecho de que al ser aplicados procesos de minería de datos, estos presentan un mejor desempeño trabajando con valores numéricos que con valores de tipo cadena ya que los valores numéricos ocupan mucho menos espacio.

9.2.3 Filtro Binarize Para este filtro fueron utilizados los repositorios sintéticos generados a partir de la página Web http://www.datgen.com los cuales una vez aplicado el filtro generaron los resultados que se muestran en las tablas 9.6, 9.7 y 9.8. En las figuras 9.9, 9.10 y 9.11 se muestra la interpretación gráfica de los tamaños de los archivos originales y sus respectivos archivos binarizados.

Tabla 9.6 Detalle de archivos originales frente a archivos binarizados(1)

Repositorio	Atributos Binarizados	Archivo original (Kb)	Archivo binarizado (Kb)	Columnas Tabla Binarizada	Tiempo (Seg)
DB20388TC23	23	1174	3355	85	9.14
DB1TC11	11	24	392	85	0.69
DB10TC11	11	226	2888	199	15.467

Figura 9.9 Relación entre los tamaños de archivos originales y binarizados(1)

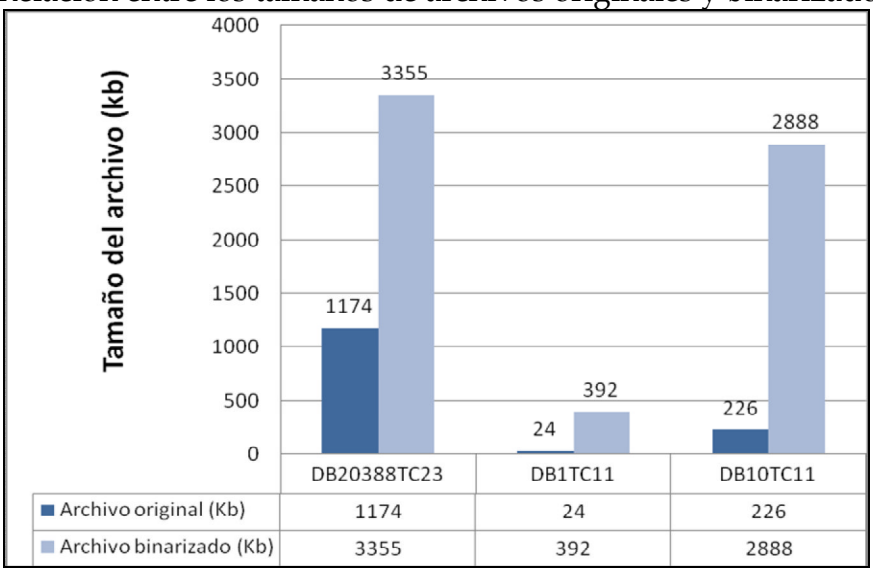


Tabla 9.7 Detalle de archivos originales frente a archivos binarizados(2)

Repositorio	Atributos Binarizados	Archivo original (Kb)	Archivo binarizado (Kb)	Columnas Tabla Binarizada	Tiempo (Seg)
DB100TC11	11	2256	39063	200	166.1
DB1000TC11	11	16370	286288	202	1210.366

Figura 9.10 Relación entre los tamaños de archivos originales y binarizados(2)

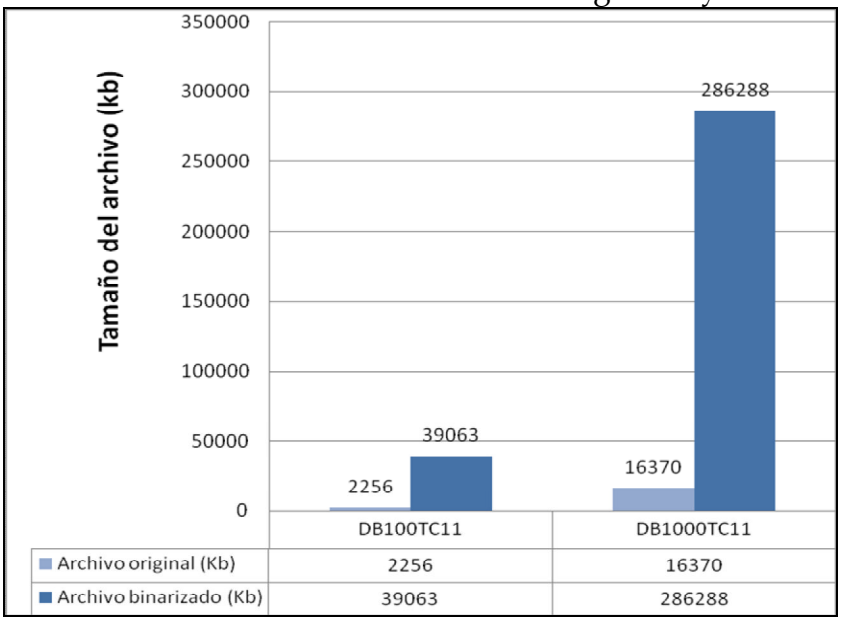
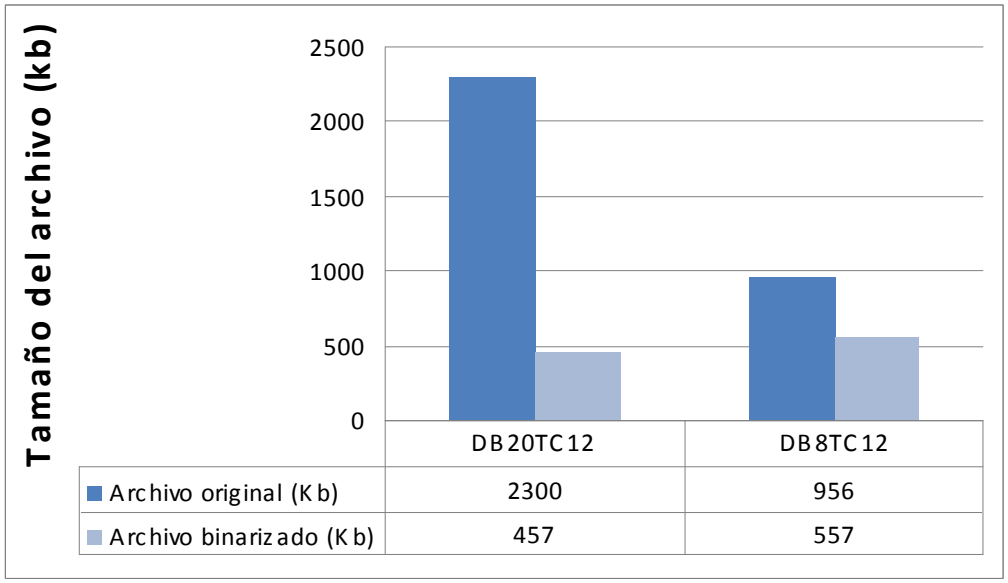


Tabla 9.8 Detalle de archivos originales frente a archivos binarizados(3)

Repositorio	Atributos Binarizados	Archivo original (Kb)	Archivo binarizado (Kb)	Columnas Tabla Binarizada	Tiempo (Seg)
DB20TC12	12	2300	457	12	9.43
DB8TC12	12	956	557	36	3.57

Figura 9.11 Relación entre los tamaños de archivos originales y binarizados(3)



De las anteriores pruebas se puede deducir que:

Si la cantidad de atributos resultantes en el archivo binarizado (Total de valores diferentes en el archivo original) es mucho mayor que la cantidad de atributos del archivo original, entonces el tamaño en Bytes del archivo binarizado también será mucho mayor con respeto al archivo original. De la misma manera se puede deducir la situación contraria, si la cantidad de atributos resultantes es menor o igual a la del archivo original entonces el tamaño en Bytes del archivo binarizado será mucho menor independientemente de la longitud en caracteres que tenga cada valor. Pero si la diferencia entre la cantidad de atributos resultantes con los originales no es muy grande, entonces el tamaño del archivo binarizado puede ser mayor o menor que el original dependiendo esta vez, de que los valores tengan una longitud mayor a un caracter.

La velocidad de ejecución de este filtro también depende completamente de la cantidad de filas que tenga el archivo y de la cantidad de valores diferentes que en él se encuentren.

9.3 PRUEBAS CON FILTROS DE LIMPIEZA

9.3.1 Filtro Email Cleaner Para la aplicación de este filtro se tomo como base de pruebas el siguiente ejemplo:

En la figura 9.12 se presenta una tabla de datos compuesta por un atributo de tipo varchar(255) denominada *"Email"* sobre la cual se aplicará este filtro. En ella se puede detallar que existen algunos correos electrónicos inválidos que tras la aplicación de este filtro procederán a una etapa de sugerencias y corrección.

Figura 9.12 Tabla de datos con algunas direcciones de Correo Electrónico (Email)

Town (varchar(255))	Email (varchar(255))	DateofBirth (date)
Bracknell	emma@writeway.co.uk	1972-07-12
Leeds	alan#bakers.com	1981-09-04
Bournmouth	alan_powell@bakery.com	1981-05-02
Gillingham	lisabell@flowershop.com"	
Bracknelll	richardevans@treats	1974-09-19
	dave.smith@.comp.co.uk	1980-09-20
Birmingham	alan.knight@compshop.com	1968-08-09
Birmingham	p_taylor@compshop.com	
Birmingham	k.hunt@compshop.com	

En la figura 9.13 se muestra una tabla de datos que contiene todas las direcciones de correo electrónico que presentaron errores tras el análisis y además contiene una columna en la cual se presentan las mejores sugerencias para cada uno de los campos considerados como inválidos. Si el campo considerado dirección de correo electrónico se aleja demasiado del formato de una dirección de correo electrónico, entonces, la sugerencia será un valor nulo.

Una vez se han verificado las direcciones de correo entonces se procede a aplicar el uso de las correcciones devolviendo como resultado la tabla de datos original pero

con los cambios efectuados sobre el campo "Email" como se muestra en la figura 9.14.

Figura 9.13 Tabla de datos con las sugerencias de Correros Electrónicos.

Email (varchar(255))	Suggest_Email (varchar(255))
emma@writeway.co.uk	emma@writeway.com.uk
alan#bakers.com	alan@bakers.com
richardevans@treats	
dave.smith@.comp.co.uk	dave.smith@comp.com.uk
lisabell@flowershop.com'	lisabell@flowershop.com

Figura 9.14 Resultado final filtro Email Cleaner

Town (varchar(255))	Email (varchar(255))	DateofBirth (date)
Bracknell	emma@writeway.com.uk	1972-07-12
Leeds	alan@bakers.com	1981-09-04
Bournmouth	alan_powell@bakery.com	1981-05-02
Gillingham	lisabell@flowershop.com	
Bracknelll		1974-09-19
	dave.smith@comp.com.uk	1980-09-20
Birmingham	alan.knight@compshop.com	1968-08-09
Birmingham	p_taylor@compshop.com	
Birmingham	k.hunt@compshop.com	

Teniendo en cuenta el análisis y las sugerencias de corrección que presenta el filtro Email Cleaner, es posible afirmar que ayuda en gran forma a tener una lista de direcciones de correo electrónico veraces de modo tal que se facilite el envío de mensajes a grandes grupos de direcciones de correo.

Es importante destacar que este filtro se compone de dos algoritmos que utilizan la programación dinámica, el primero enfocado en inspeccionar completamente un campo para verificar si cumple o no con la estructura de un correo electrónico y el segundo, enfocado en analizar los campos que no cumplieron con la estructura, transformándolos y adaptándolos a una posible estructura.

CONCLUSIONES

EXDACLET es la primera aplicación Web enfocada en el proceso de Limpieza de Datos, el cual hace parte activa del proceso de investigación en Descubrimiento de Conocimiento en Bases de Datos del grupo de investigación GRIAS KDD del programa de Ingeniería de Sistemas de la Universidad de Nariño.

EXDACLET es una herramienta de licencia GNU-LGPL que funciona en distintos Sistemas Operativos, bajo distintos tipos de red y puede ser acoplada a cualquier Servidor Web que utilice Apache 2.x, Php 5.x y Mysql 5.x.

EXDACLET es una herramienta completamente amigable, de fácil manejo y que presenta lo más detallado posible la información de las tablas de datos cargadas por el usuario a partir de archivos planos y los respectivos filtros aplicados sobre ellas.

EXDACLET, al ser una herramienta orientada a la Web, es completamente interactiva y no consume recursos locales exceptuando lo que requiera el navegador Web sobre el cual esta haya sido cargada.

EXDACLET es una herramienta flexible que cuenta con un total de 51 filtros distribuidos en módulos para precarga, limpieza, transformación y selección de datos, dándole así una mayor aplicabilidad sobre conjuntos de datos multidisciplinarios de tal forma que pueda ser utilizada bajo distintos ámbitos.

EXDACLET cuenta con algunos filtros de selección que a pesar de su alto costo de procesamiento y tiempo, generan resultados valiosos que difícilmente pueden ser obtenidos con otro tipo de algoritmos; además, estos algoritmos no han sido implementados en herramientas que se dedican a la Limpieza de Datos, aumentado así su aplicabilidad.

EXDACLET proporciona filtros capaces de mejorar la calidad de los datos y a su vez de codificarlos para que estos puedan ser utilizados en procesos de minería de datos, ayudando así a que estos procesos generen información real y coherente y que además puedan terminar satisfactoriamente su análisis dado que es posible que algunos algoritmos de minería de datos colapsen porque los datos pueden estar incorrectos.

El desarrollo de herramientas orientadas a la Web bajo Software libre permite y facilita a herramientas como EXDACLET su masificación y por lo tanto su acceso a las PYMES que difícilmente pueden adquirir herramientas dedicadas a procesos específicos por sus altos costos.

Los resultados generados y el tiempo de ejecución de todos los filtros de selección, limpieza y tranformación aplicados en EXDACLET dependen únicamente de la cantidad de registros que compongan una tabla de datos y de la estructura de los valores que estén contenidos en ella.

Todos los filtros de Selección que utilizan algoritmos de similitud (JARO-WINKLER, METAPHONE, DOUBLE-METAPHONE, LEVENSHTEIN) cumplen con la expresión: "***si A similar a B y B similar a C, entonces, no siempre A es similar a C***", es decir, que si una cadena tiene una alta relación con otras dos cadenas no siempre esas dos cadenas tienen una alta relación entre sí, siendo así que esas dos cadenas son excluyentes. Por esta razón, son los filtros que mayor cantidad de recursos y tiempo consumen en el momento de su aplicación.

Finalmente, el desarrollo de este trabajo y de esta herramienta ha permitido que apliquemos todo el conocimiento adquirido en el programa de Ingeniería de Sistemas, en especial de las electivas de bases de datos y las materias de Seminario de Computación e Informática, así como nuestro trabajo y aprendizaje dentro del Grupo de Investigación GRIAS Línea KDD.

RECOMENDACIONES

Como punto de partida para trabajos futuros se han establecido una serie de recomendaciones:

Implementar nuevos algoritmos que permitan aumentar la aplicabilidad de EXDACLET a diversos esquemas de bases de datos, que ayuden de forma eficaz y eficiente en los procesos de preparación de datos para su posterior tratamiento.

Realizar más pruebas sobre la totalidad de los filtros para detectar posibles errores y sugerir el mejoramiento de los mismos garantizando así la continuidad del proyecto.

Mantener en constante actualización la herramienta de acuerdo a las nuevas versiones de las bibliotecas de desarrollo bajo las cuales EXDACLET está soportada.

Disponer de EXDACLET como material de apoyo a las electivas de base de datos dentro del programa de Ingeniería de Sistemas y también como parte importante del proceso de investigación del grupo GRIAS KDD.

REFERENCIAS BIBLIOGRAFICAS

[1] Waikato ML Group. The waikato environment for knowledge analisys. http://www.cs.waikato.ac.nz/ml/weka.

[2] Faculty of Computer and Slovenia Information Science, University of Liubliana. Orange, fruitful and fun. http://www.ailab.si/orange, 2006.

[3] EBusiness Technology Institute. Ebusiness technology institute, the university of hong kong. http://www.eti.hku.hk, 2005.

[4] Pure ListCleaner Pro. http://www.winpure.com/

[5] MatchIT Suite. http://helpit.com/folders/software_solutions/deduplication/

[6] ListWare 6.0. http://www.melissadata.com/listware/listware.htm

[7] MerlinMerge® SpeedPro. http://www.merge-purge-software.com/

[8] Melissa Data. http://www.melissadata.com/software.htm

[9] Tsang J., Rudychev I. Bayser Consulting. Endowing Reasoning Capabilities to a Matching/Scrubbing Algorithm, August 16, 2002, Journal of Data Warehousing, October 2002.

[10] Molina J, Garcia J. "Técnica de análisis de datos". Universidad Carlos III, Madrid, 2005.

[11] EBusiness Technology Institute. The university of hong kong. “Alphaminer, An open source data mining platform”, http://www.eti.hku.hk/alphaminer/.

[12] Timaran R, Calderón A. Ramirez I, Guevara F, and Alvarado J. “TariyKDD, una herramienta de Mineria de Datos débilmente acoplada a un SGBD”. Jornadas Iberoamericanas de Ingeniería de Software e Ingeniería del Conocimiento – JIISIC 2008, Guayaquil, Ecuador, 2008.

[13] Repici J, Burton D. The comma separated value (CSV) file format, 2007, http://www.creativyst.com/Doc/Articles/CSV/CSV01.htm

[14] Waikato ML Group. Attributerelation file format (arff). http://www.cs.waikato.ac.nz/ml/weka/arff.html/.

[15] Murray R. Speigel. ESTADISTICA. Teoría y Problemas. McGraw-Hill, 1969.

[16] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. Doklady Akademii Nauk SSSR. 1965 (Russian). English translation in Soviet Physics Doklady. 1966.

[17] Lawrence Philips Software Engineer at Verity, Inc. ("The leading provider of knowledge retrieval solutions for the enterprise and the Internet"), Sunnyvale, CA. http://www.ddj.com

[18] Monge, A., and Elkan, C. 1996. The field-matching problem: algorithm and applications. In proceedings of the Second International Conference on Knowledge Discovery and Data Mining.

[19] Monge, A., and Elkan, C. 1997. An efficient domain-independent algorithm for detecting approximately duplicate database records. In the proceedings of the SIGMOD 1997 workshop on data mining and knowledge discovery.

[20] William W. Cohen, Pradeep Ravikumar, Stephen E. Fienberg. 2003. A Comparison of String Distance Metrics for Name-Matching Tasks

[21] Jaro, M. A. (1989). "Advances in record linking methodology as applied to the 1985 census of Tampa Florida". *Journal of the American Statistical Society* 64: 1183-1210.

[22] Jaro, M. A. (1995). "Probabilistic linkage of large public health data file". *Statistics in Medicine* 14: 491-498.

[23] Winkler, W. E. (1999). "The state of record linkage and current research problems". *Statistics of Income Division, Internal Revenue Service Publication R99/04*.

[24] Hernández J, Ramirez Mª J, Ferri C. 2005. "Introducción a la Mineria de Datos", PEARSON, PRENTICE HALL,

[25] U. Fayyad, G. Piatestky-Shaphiro, P. Smyth and R. Uthurusamy. From Data Mining to Knowledge Discovery: An Overview, Advances in Knowledge Discovery and Data Mining, AAAI/MI Press 1996.

[26] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery: An overview in knowledge discovery and data mining. In AAAI Pres / The MIT Press, 1998.

[27] U. Fayyad, D. Haussler, P. Stolorz. KDD For Sciense Data Analysis: Issues and Examples, 1996.

[28] Han, J., Kamber, M. Data Mining: Concepts and Techniques, Morgan Kaufmann, 2001

[29] Tim O'Reilly. What Is Web 2.0. O'Reilly Network. 2006.

[30] Tim O'Reilly. Web 2.0 Compact Definition: Trying Again. 2007.

[31] PHP, http://www.php.net

[32] Powell, Thomas A.; Schneider, Fritz. *JavaScript: The Complete Reference*. McGraw-Hill Companies. 2001

[33] McDuffie, Tina S. *JavaScript Concepts & Techniques: Programming Interactive Web Sites*. Franklin, Beedle & Associates. 2003.

[34] Flanagan, D. *JavaScript: The Definitive Guide*, 5th Edition, O'Reilly & Associates. 2006.

[35] Garrett J. "Ajax: A New Approach to Web Applications", Adaptive Path, 2005.

[36] J. White., Wilson M. XAJAX. 2005 - 2007, http://www.xajaxproject.org.

[37] Meyer E.: Cascading Style Sheets: The Definitive Guide, Third Edition.

[38] Meyer E.: Cascading Style Sheets 2.0 Programmer's Reference, McGraw-Hill Osborne Media.

[39] EXTJS, Jack Slocum. http://www.extjs.com

[40] JSON. http://www.json.org

[41] WAMP, http://www.wampserver.com

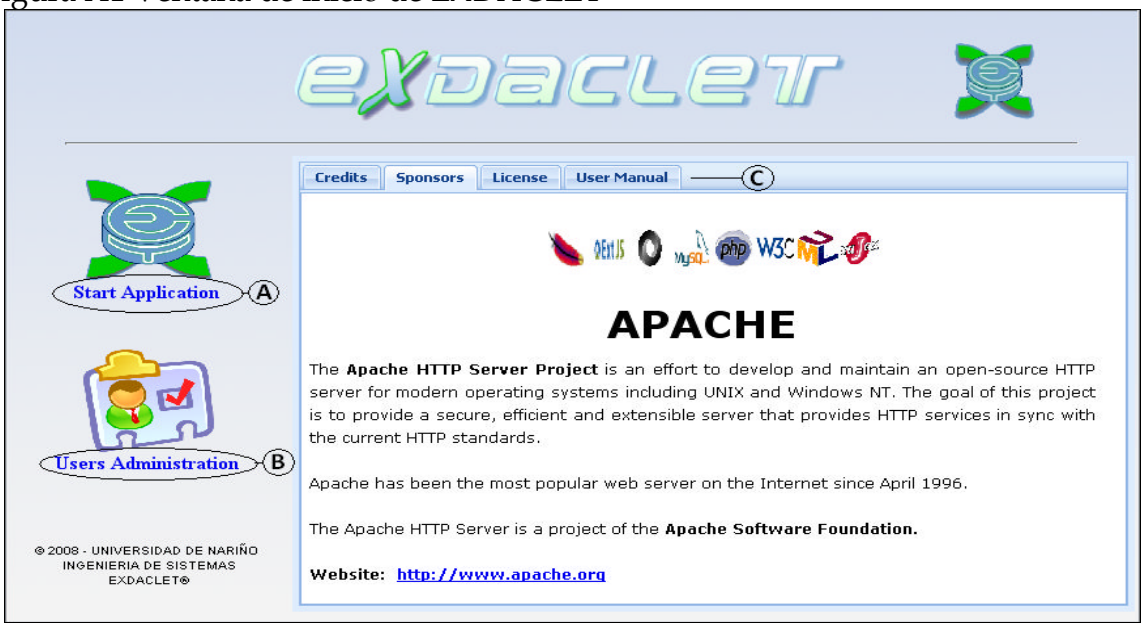
[42] APACHE, http://www.apache.org.

[43] MySQL, http://www.mysql.com

ANEXOS

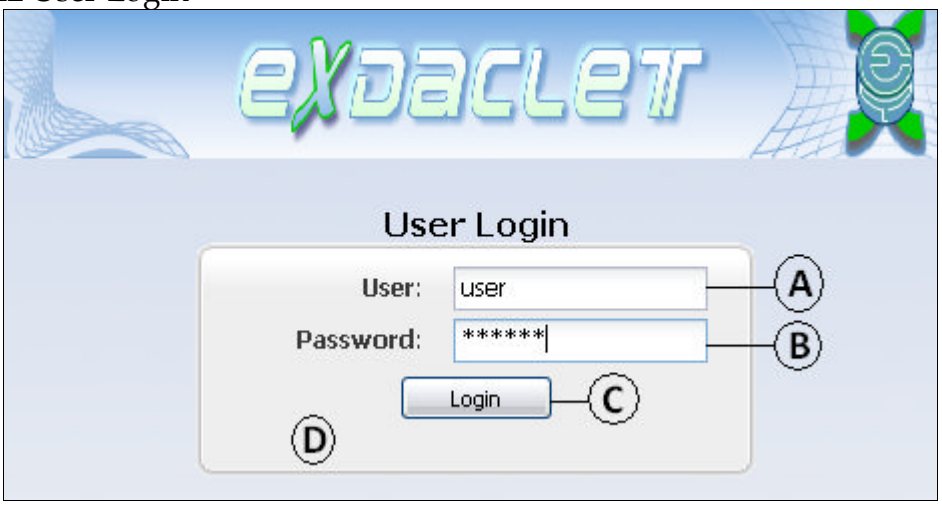
A1. MANUAL DE USUARIO

Figura A1 Ventana de inicio de EXDACLET



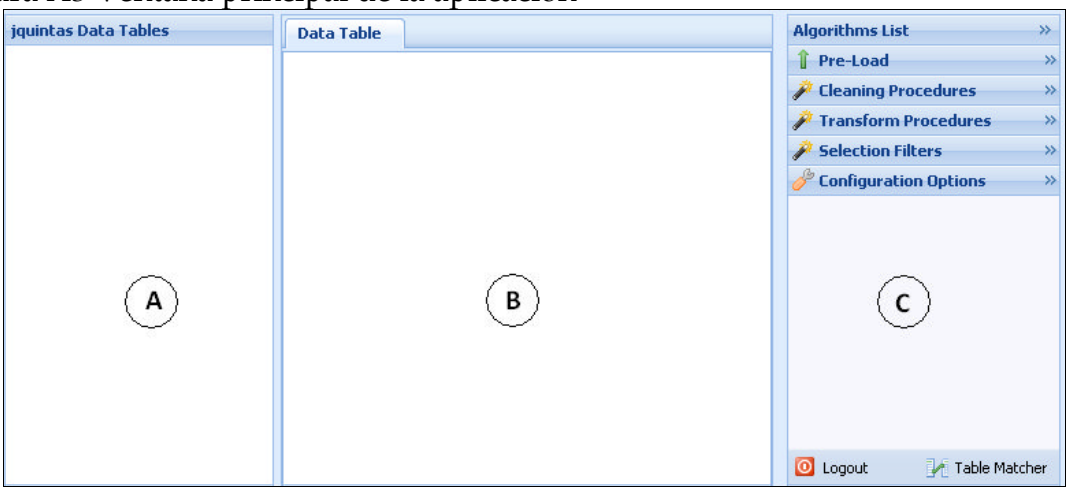
Acción del Usuario	Respuesta del Sistema
	1. Aparece la ventana de inicio de la aplicación.
2. El usuario hace clic en el botón Start Application (A)	3. Se despliega la ventana User Login donde el usuario se registra en la aplicación.
4. El usuario hace clic en el botón Users Administration (B).	5. Se despliega la ventana User Admin en donde el usuario administrador se registra en la aplicación para ingresar al módulo de administración.
6. El usuario hace clic en los botones (C): Credits, Sponsors, License, User Manual.	7. Se muestra un texto con la información correspondiente a cada botón.

Figura A2 User Login



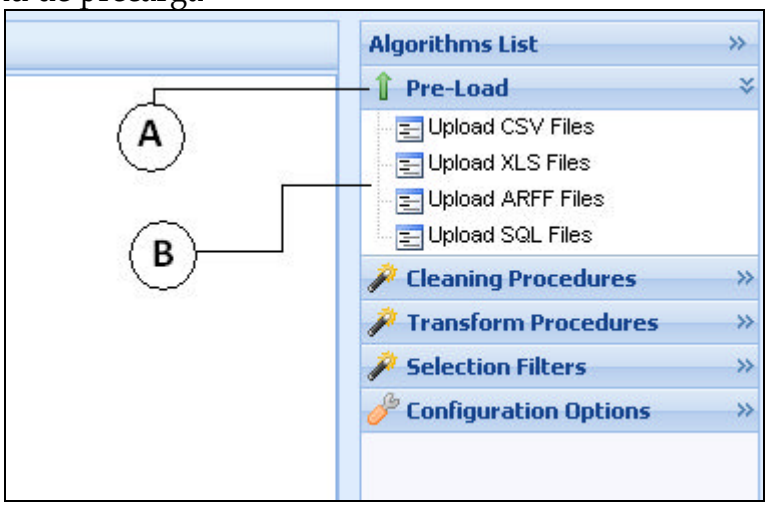
Acción del Usuario	Respuesta del Sistema
	1. Se muestra la ventana User Login.
2. El usuario escribe su login en el campo User (A). 3. El usuario escribe su contraseña en el campo Password (B)	
4. El usuario hace clic en el botón Login (C).	5. Si la verificación de los datos Login y Password es correcta, entonces se muestra la ventana principal de la aplicación, de lo contrario se muestra un mensaje de error en el campo (D).

Figura A3 Ventana principal de la aplicación



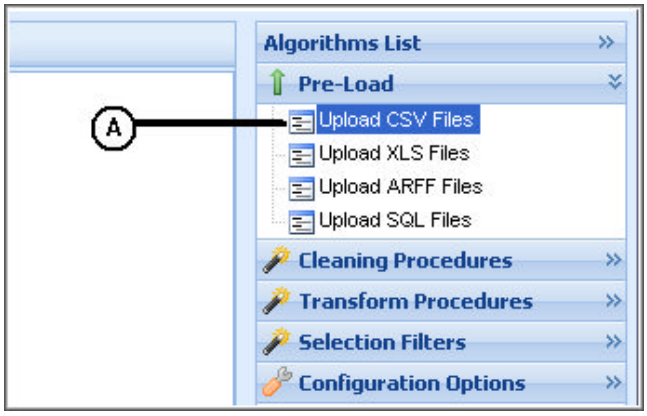
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa a la aplicación.	2. Aparece la interfaz gráfica de la aplicación, como se muestra en la figura A3. (A): Árbol de tablas, en esta sección se encuentran todos los datos que el usuario ha trabajado en forma de tablas. (B): Área de trabajo, donde el usuario observa tanto las tablas como los resultados de los filtros que se aplicaron sobre ciertas tablas. (C): Área de Algorithms, donde se encuentran los diferentes tipos de filtros que se pueden aplicar a los datos de una tabla, entre los filtros se encuentran la precarga, los filtros de limpieza, los filtros de transformación, los filtros de selección y algunas opciones de configuración. Por defecto, todas las pestañas de los diferentes filtros aparecen cerradas.

Figura A4 Menú de precarga



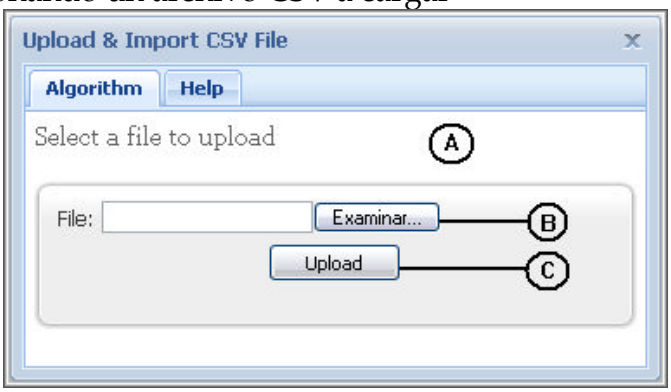
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en la Pestaña Pre-Load (A).	2. Se despliega una lista de formatos de archivo que el sistema puede interpretar (B).

Figura A5 Importando un archivo CSV



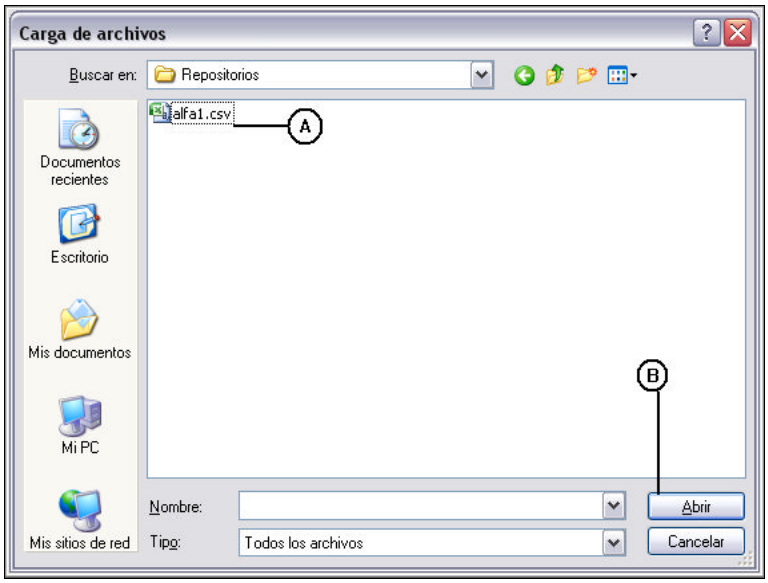
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el icono Upload CSV Files (A).	2. Se despliega un cuadro de dialogo que lleva por título Upload & Import CSV Files.

Figura A6 Seleccionando un archivo CSV a cargar



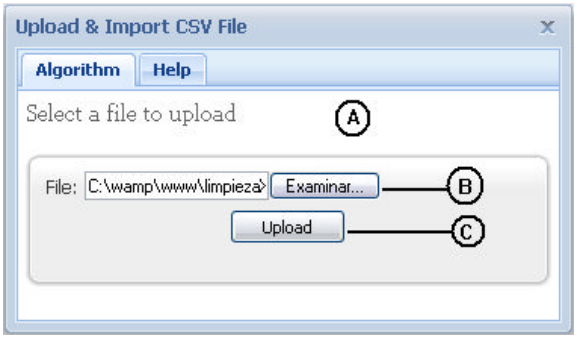
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Examinar… (B).	2. Se despliega una ventana en la que se puede buscar y seleccionar un archivo.

Figura A7 Ventana de selección de archivos



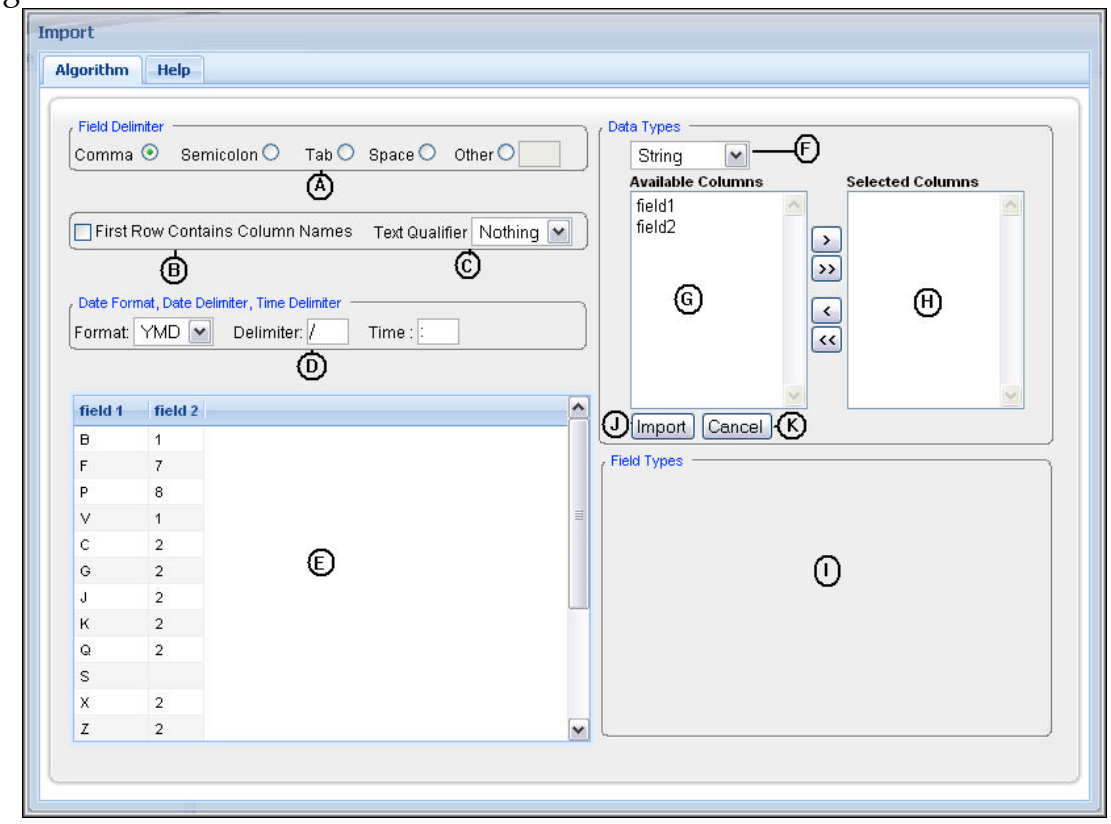
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona un archivo acorde con el formato que escogió (A) y hace clic en el botón Abrir (B).	2. Se retorna al cuadro de dialogo Upload & Import CSV Files, ahora este tiene la dirección del archivo seleccionado por el usuario previamente.

Figura A8 Subiendo un archivo al servidor



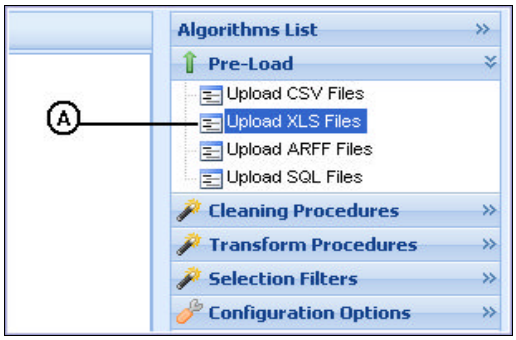
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Upload (C).	2. Se despliega una ventana que permite establecer los parámetros adecuados para importar el archivo.

Figura A9 Parámetros del archivo CSV



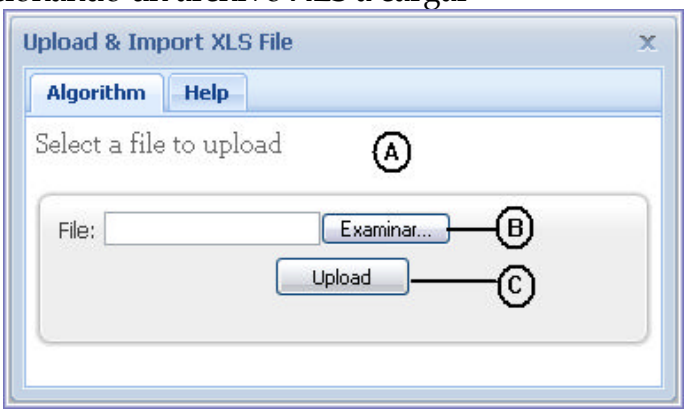
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el delimitador de campos (A).	2. Se actualiza la tabla de vista preliminar (E).
3. El usuario establece si el archivo contiene en la primera fila el nombre de cada uno de los campos que conforman este archivo (B).	4. Se actualiza la tabla de vista preliminar (E).
5. El usuario determina cual es el delimitador de cadenas de caracteres en la tabla (C).	6. Se actualiza la tabla de vista preliminar (E).
7. El usuario establece el formato de fecha, sus delimitadores y los delimitadores de tiempo (D).	8. Se actualiza la tabla de vista preliminar (E).
9. El usuario debe asignar un tipo a cada uno de los atributos que componen la tabla, el usuario selecciona el tipo (F).	10. Se cargan en el campo Selected Columns (H) los atributos que han sido asignados para el tipo de dato seleccionado.
11. El usuario transfiere los atributos deseados desde el campo Available Columns (G) hacia el campo Selected Columns (H).	12. Se cargan los campos transferidos por el usuario y se muestra la estructura total de los datos en Field Types (I).
13. El usuario hace clic en el botón Importar (J)	14. El sistema importa los datos teniendo en cuenta los parámetros suministrados por el usuario y se carga en el área de trabajo la tabla importada, además se agrega al árbol de tablas en la parte izquierda.
15. Si el usuario presiona el botón Cancel (K)	16. El sistema sale del módulo de importación.

Figura A10 Importando un archivo XLS



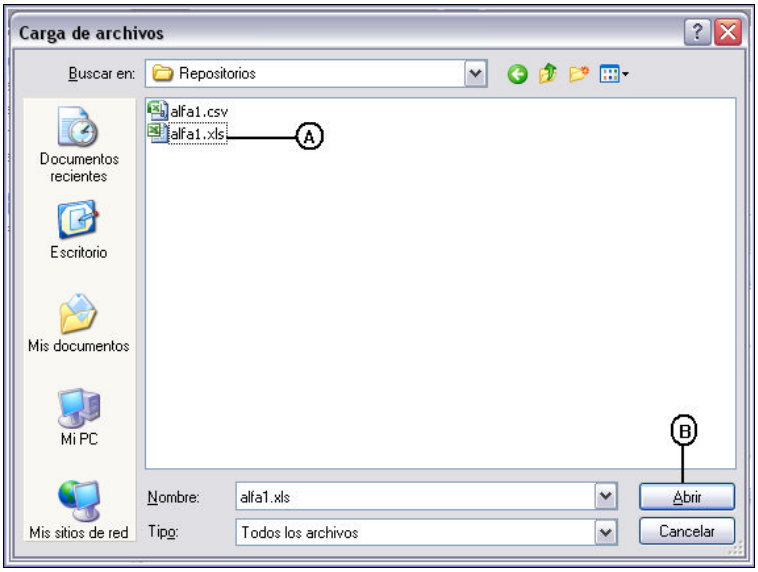
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el icono Upload XLS Files (A).	2. Se despliega un cuadro de dialogo que lleva por título Upload & Import XLS Files.

Figura A11 Seleccionando un archivo XLS a cargar



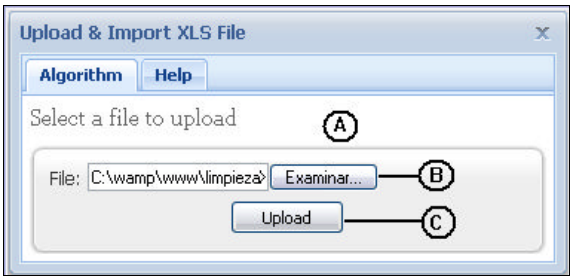
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Examinar… (B).	2. Se despliega una ventana en la que se puede buscar y seleccionar un archivo.

Figura A12 Ventana de selección de archivos



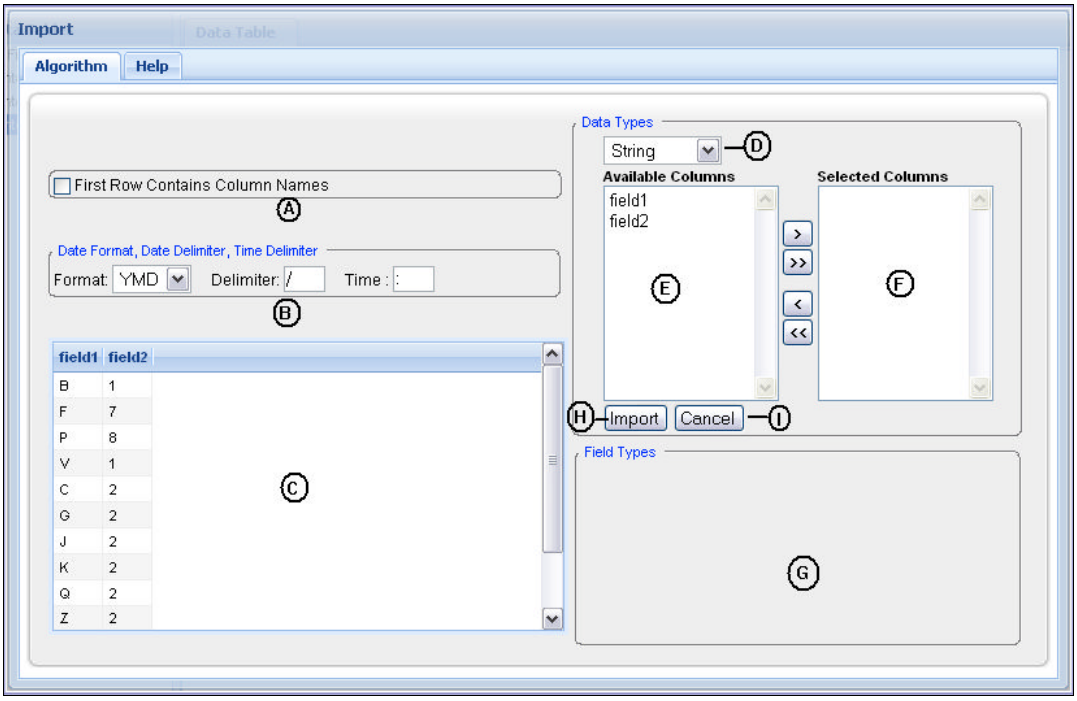
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona un archivo acorde con el formato que escogió (A) y hace clic en el botón Abrir (B).	2. Se retorna al cuadro de dialogo Upload & Import XLS Files, ahora este tiene la dirección del archivo seleccionado por el usuario previamente.

Figura A13 Subiendo un archivo al servidor



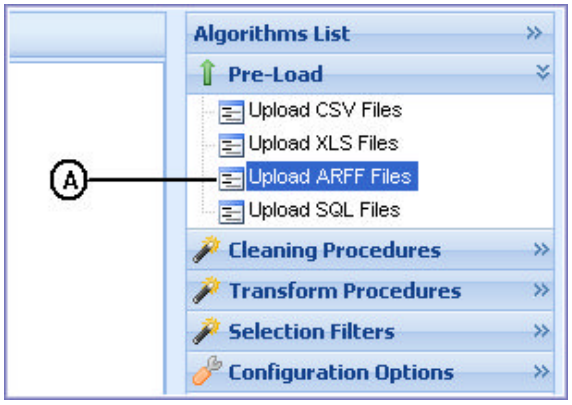
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Upload (C).	2. Se despliega una ventana que permite establecer los parámetros adecuados para importar el archivo.

Figura A14 Parámetros del archivo XLS



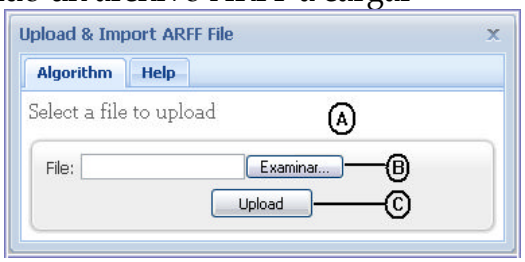
Acción del Usuario	Respuesta del Sistema
1. El usuario establece si el archivo contiene en la primera fila el nombre de cada uno de los campos que conforman este archivo (A).	2. Se actualiza la tabla de vista preliminar (C).
3. El usuario establece el formato de fecha y los diferentes delimitadores para esta, además de los delimitadores de tiempo (B).	4. Se actualiza la tabla de vista preliminar (C).
5. El usuario debe asignar un tipo a cada uno de los atributos que componen la tabla, el usuario selecciona el tipo (D).	6. Se cargan en el campo Selected Columns (F) los atributos que han sido asignados para el tipo de dato seleccionado.
7. El usuario transfiere los atributos deseados desde el campo Available Columns (E) hacia el campo Selected Columns (F).	8. Se cargan los campos transferidos por el usuario y se muestra la estructura total de los datos en Field Types (G).
9. El usuario hace clic en el botón Importar (H)	10. El sistema importa los datos teniendo en cuenta los parámetros suministrados por el usuario y se carga la tabla importada en el área de trabajo, además se agrega al árbol de tablas en la parte izquierda.
11. Si el usuario presiona el botón Cancel (I)	12. El sistema sale del módulo de importación.

Figura A15 Importando un archivo ARFF



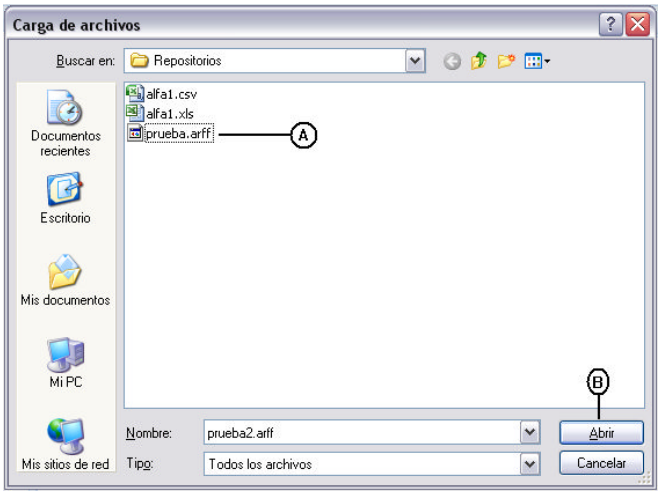
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el icono Upload ARFF Files (A).	2. Se despliega un cuadro de dialogo que lleva por título Upload & Import ARFF Files.

Figura A16 Seleccionando un archivo ARFF a cargar



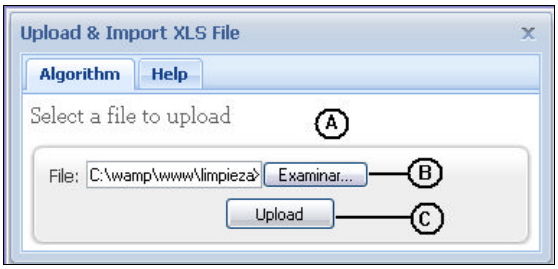
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Examinar… (B).	2. Se despliega una ventana en la que se puede buscar y seleccionar un archivo.

Figura A17 Ventana de selección de archivos



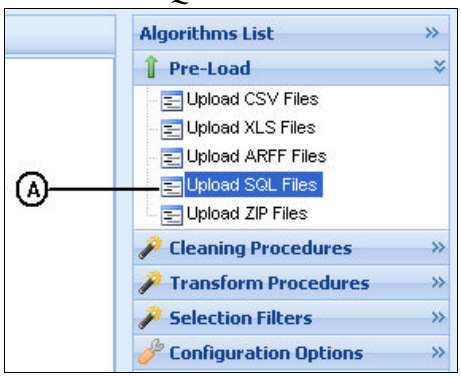
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona un archivo acorde con el formato que escogió (A) y hace clic en el botón Abrir (B).	2. Se retorna al cuadro de dialogo Upload & Import ARFF Files, ahora este tiene la dirección del archivo seleccionado por el usuario previamente.

Figura A18 Subiendo un archivo al servidor



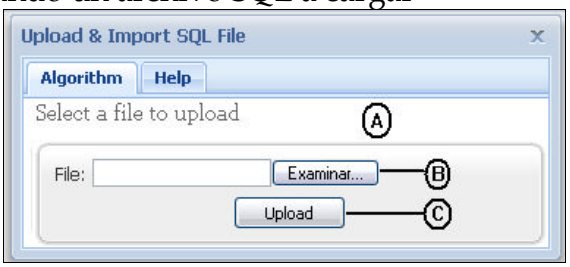
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Upload (C).	2. Debido a que el formato Arff es muy completo, no se solicitan parámetros y el archivo se importa y se muestra en el área de trabajo, además se agrega al árbol de tablas en la parte izquierda.

Figura A19 Importando un archivo SQL



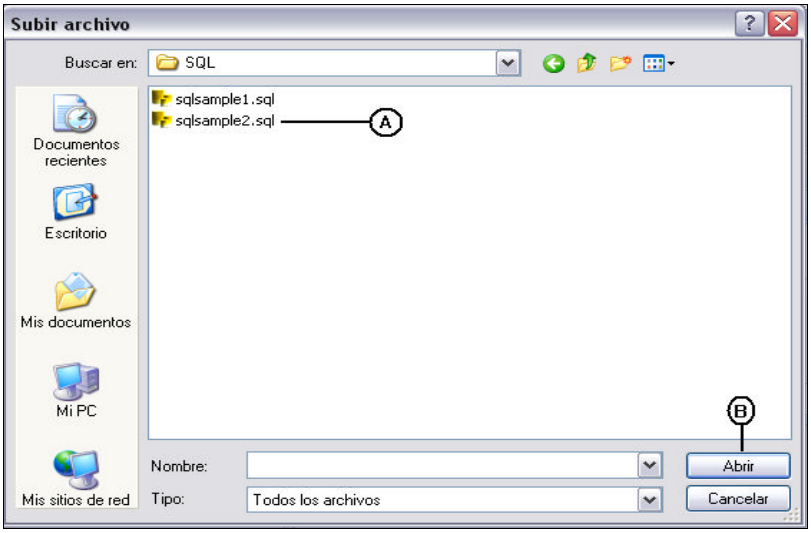
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el icono Upload SQL Files (A).	2. Se despliega un cuadro de dialogo que lleva por título Upload & Import SQL Files.

Figura A20 Seleccionando un archivo SQL a cargar



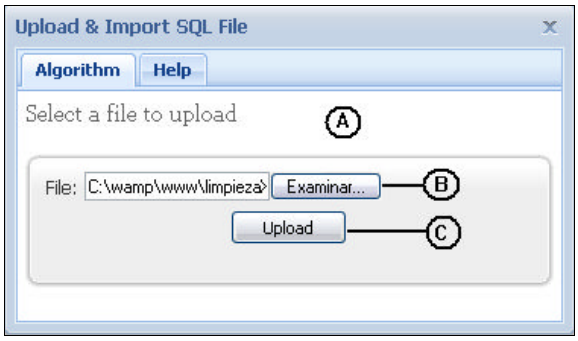
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Examinar… (B).	2. Se despliega una ventana en la que se puede buscar y seleccionar un archivo.

Figura A21 Ventana de selección de archivos



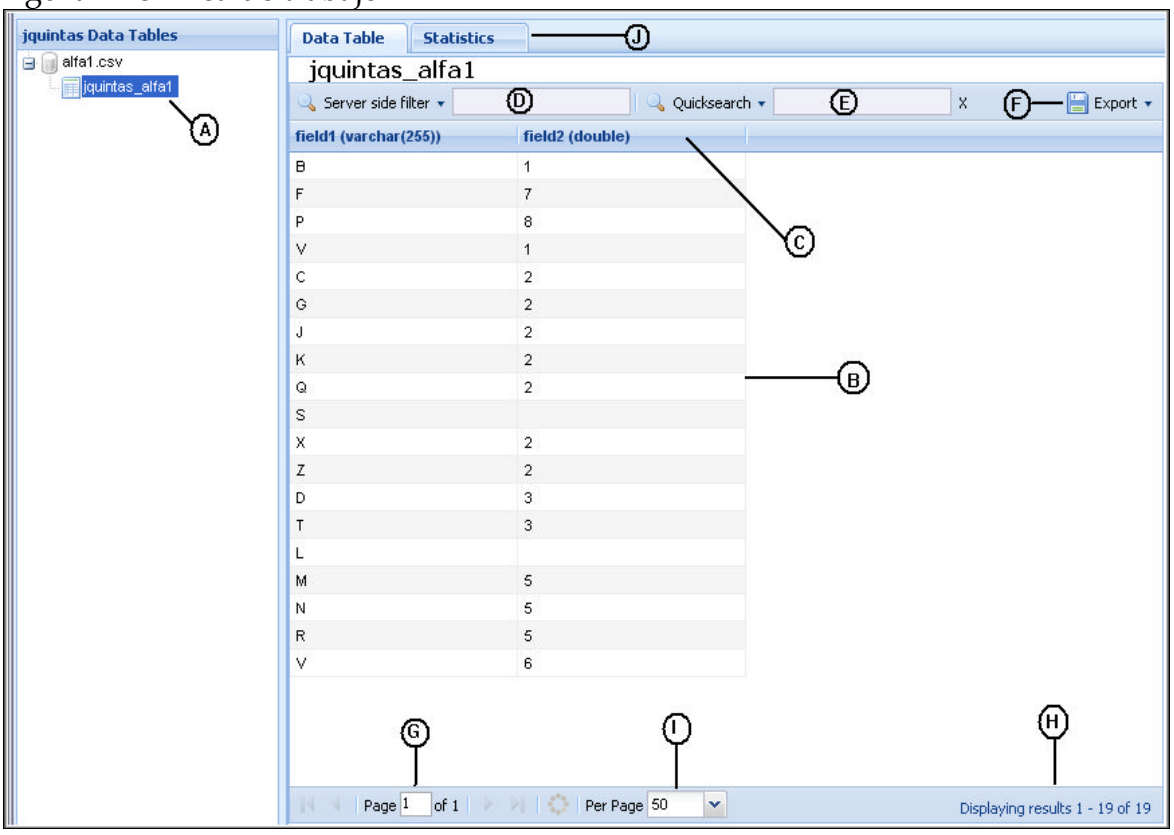
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona un archivo acorde con el formato que escogió (A) y hace clic en el botón Abrir (B).	2. Se retorna al cuadro de dialogo Upload & Import SQL Files, ahora este tiene la dirección del archivo seleccionado por el usuario previamente.

Figura A22 Subiendo un archivo al servidor



Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Upload (C).	2. Debido a que el formato SQL es muy completo, no se solicitan parámetros y el archivo se importa y se muestra en el área de trabajo, además se agrega al árbol de tablas en la parte izquierda.

Figura A23 Área de trabajo

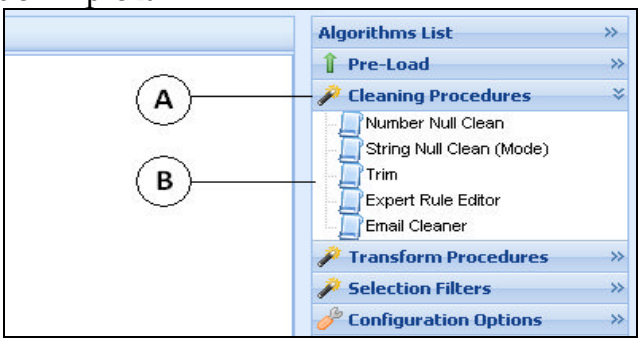


Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona con doble clic la tabla que desea cargar (A).	2. Se cargan los datos de la tabla en el área de trabajo como se indica en la figura A23. (B): Datos contenidos en la tabla. (C): Nombre y tipo del Campo. (D): Server Side Filter, filtro de búsqueda en toda la base de datos, en este se pueden seleccionar los campos de búsqueda. (E): QuickSearch, filtro de búsqueda en los datos que se encuentran dentro de la paginación actual. (F): Export, despliega una lista de formatos en los que se puede exportar la tabla. (G): Indica el número de la página del total de páginas que contienen los datos cargados. (H): Indica la cantidad de registros existentes en la tabla de datos. (I): Per Page, Indica la cantidad de registros a mostrar por pagina. (j): La pestaña statistics muestra un resumen estadístico de los datos actualmente cargados.

Figura A24 Estadísticas

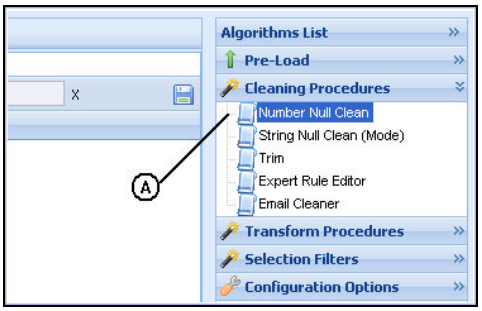
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona la pestaña Statistics (A).	2. Se carga la interfaz de Statistics, que permite generar gráficas y visualizar las estadísticas de los campos deseados como se observa en la figura A24. El usuario puede definir la cantidad de campos y el tipo de gráficos a generar con los campos: (B): Available Columns, se transfieren a este campo los atributos que no intervienen en la gráfica y estadísticas. (C): Selected Columns, en este campo se encuentran los atributos que intervienen directamente en la generación de estadísticas. (D): Char Type, permite seleccionar el tipo de gráficos, estos pueden ser de barras o de líneas. (E): Statistics Type, permite seleccionar el tipo de estadística, esta puede ser de campos llenos (Populated_Cells) o campos vacios (Empty_Cells). (F): Graph, actualiza las gráficas con los datos seleccionados en el campo (C). (G): Tabla estadística generada teniendo en cuenta los parámetros establecidos por el usuario. (H): Gráfica de las estadísticas.

Figura A25 Filtros de limpieza



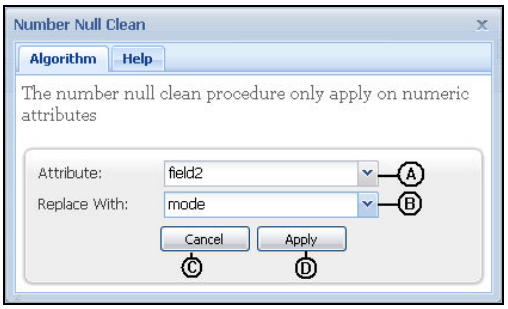
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en la pestaña Cleaning Procedures (A).	2. Se despliega una lista de filtros de limpieza (B).

Figura A26 Filtro Number Null Clean



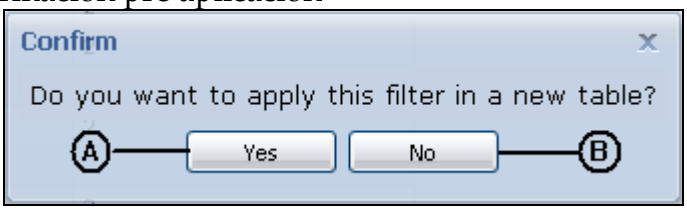
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Number Null Clean (A).	2. Se despliega la ventana Number Null Clean donde se introducen los parámetros del algoritmo; para ejecutar un algoritmo siempre debe estar una tabla cargada en el área de trabajo.

Figura A27 Parámetros del filtro Number Null Clean



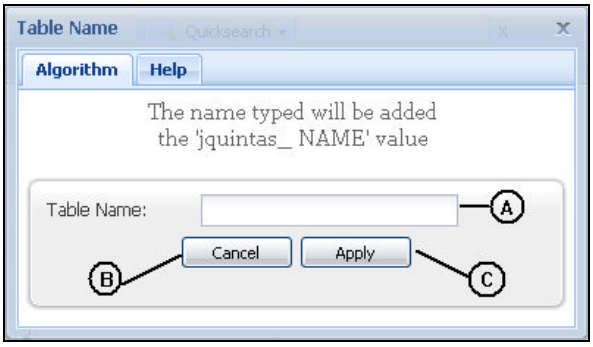
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el atributo de la tabla cargada objeto del filtro (A). 2. El usuario selecciona el valor por el cual desea reemplazar los valores nulos (B). 3. El usuario hace clic en el botón Apply (D). 5. El usuario hace clic en el botón Cancel (C)	4. Se despliega una ventana de confirmación. 6. Se cancela la aplicación del filtro.

Figura A28 Confirmación pre aplicación



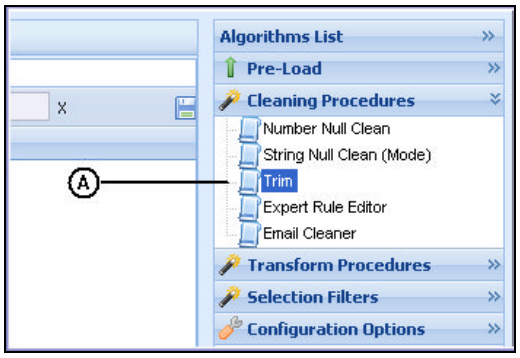
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A).	2. Se despliega la ventana Table Name
3. El usuario hace clic en el botón No (B).	4. El filtro se aplica sobre la misma tabla.

Figura A29 Table Name



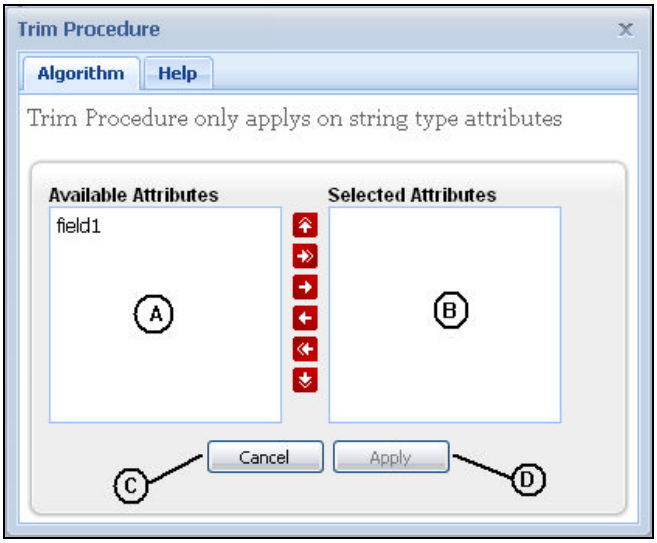
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A).	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error.
3. El usuario hace clic en el botón Apply (C).	4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas.
5. El usuario hace clic en el botón Cancel (B)	6. Se cancela la aplicación del filtro.

Figura A30 Filtro Trim



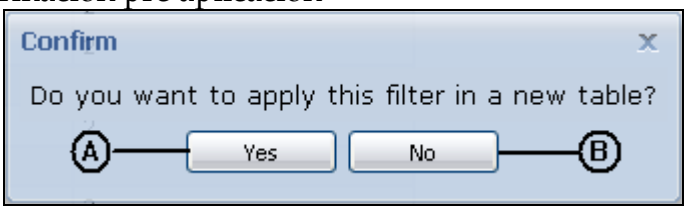
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Trim (A).	2. Se despliega la ventana Trim procedure.

Figura A31 Parámetros del filtro Trim



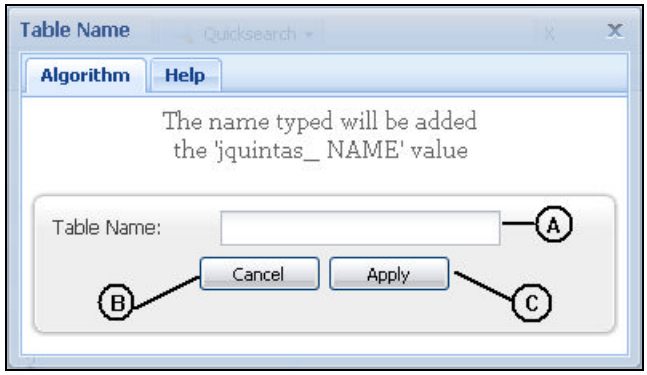
Acción del Usuario	Respuesta del Sistema
1. El usuario transfiere los atributos que serán objeto del filtro del campo Available Attributes (A) hacia el campo Selected Attributes (B). 3. El usuario hace clic en el botón Apply (D).	2. Se activa el botón Apply (D). 4. Se despliega una ventana de confirmación.

Figura A32 Confirmación pre aplicación



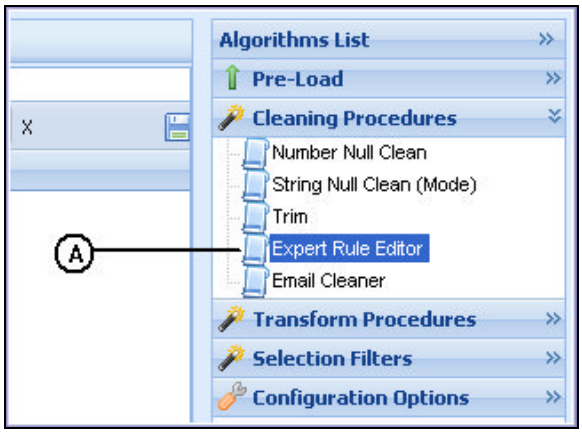
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A). 3. El usuario hace clic en el botón No (B).	2. Se despliega la ventana Table Name 4. El filtro se aplica sobre la misma tabla.

Figura A33 Table Name



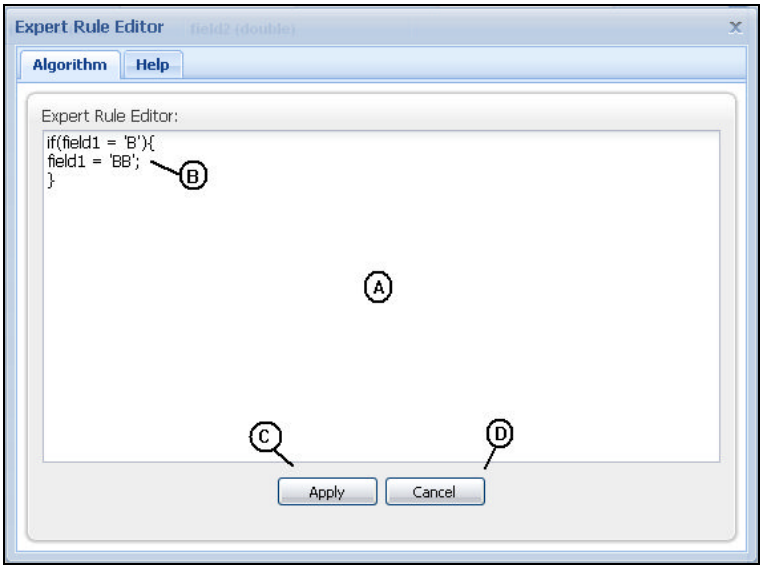
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón Cancel (B)	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas. 6. Se cancela la aplicación del filtro.

Figura A34 Filtro Expert Rule Editor



Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Expert Rule Editor (A).	2. Se despliega la ventana expert Rule Editor.

Figura A35 Editor



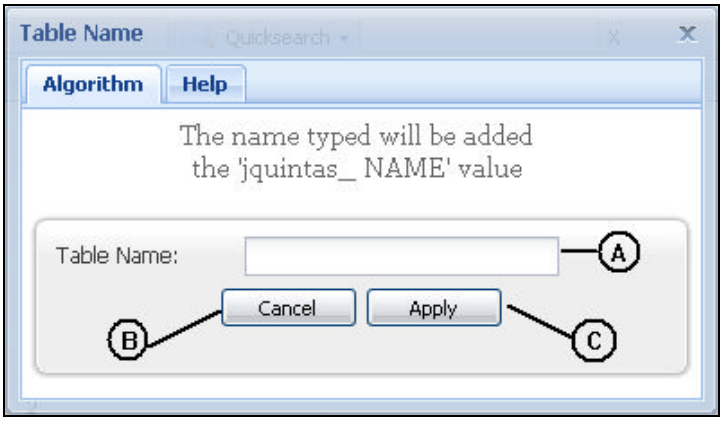
Acción del Usuario	Respuesta del Sistema
1. El usuario escribe las reglas que desea aplicar, las reglas se escriben en el formato clásico del if como lo señala el ejemplo (B).	

Acción del Usuario	Respuesta del Sistema
2. El usuario hace clic en el botón Apply (D).	3. Se indica en la parte baja de la ventana y con letras de color rojo si dentro de las reglas escritas existe algún error sintáctico, de lo contrario pasa a la ventana de verificación para aplicar las reglas.
4. El usuario hace clic en el botón Cancel (D).	5. Se cancela la aplicación del filtro.

Figura A36 Confirmación pre aplicación

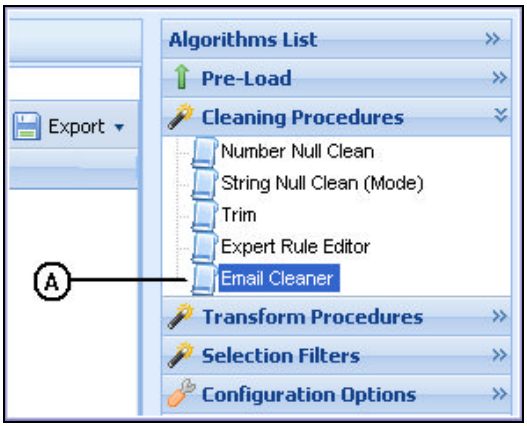
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A). 3. El usuario hace clic en el botón No (B).	2. Se despliega la ventana Table Name 4. El filtro se aplica sobre la misma tabla.

Figura A37 Table Name



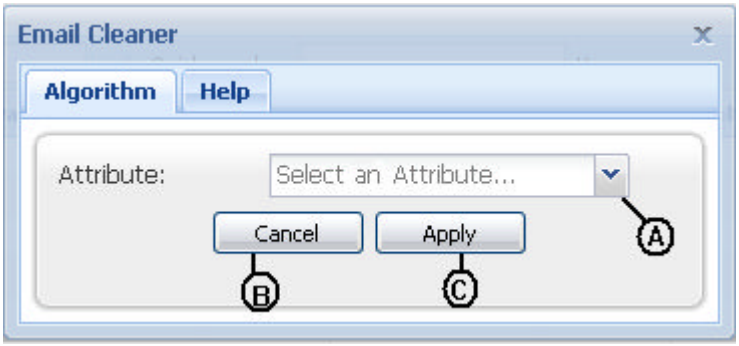
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón Cancel (B)	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas. 6. Se cancela la aplicación del filtro.

Figura A38 Filtro Email Cleaner



Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Email Cleaner (A).	2. Se despliega la ventana Parámetros Email Cleaner.

Figura A39 Parámetros Email Cleaner

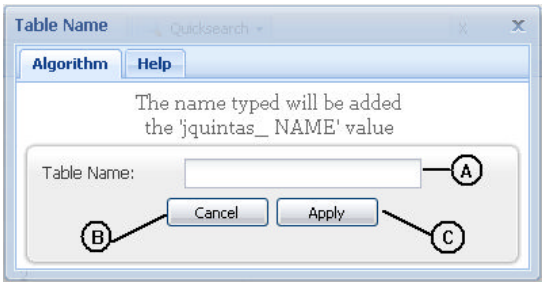


Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el atributo que contiene las direcciones de correo electrónico (A). 2. El usuario hace clic en el botón Apply (C).	3. Se despliega la ventana de verificación de confirmación para aplicar el filtro.

Figura A40 Confirmación pre aplicación

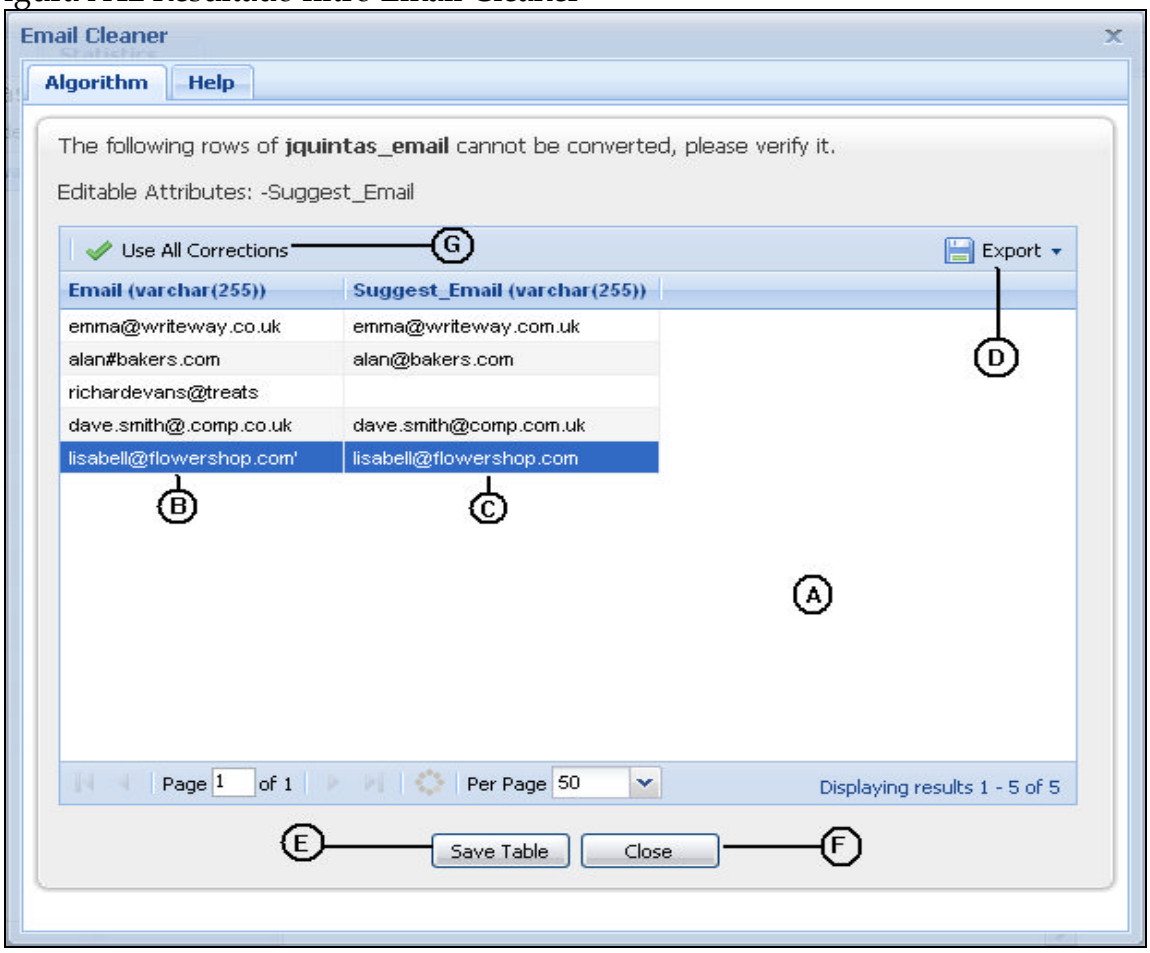
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A). 3. El usuario hace clic en el botón No (B).	2. Se despliega la ventana Table Name 4. El filtro se aplica sobre la misma tabla.

Figura A41 Table Name



Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón Cancel (B)	2. Si el nombre es incorrecto o no está disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas. 6. Se cancela la aplicación del filtro.

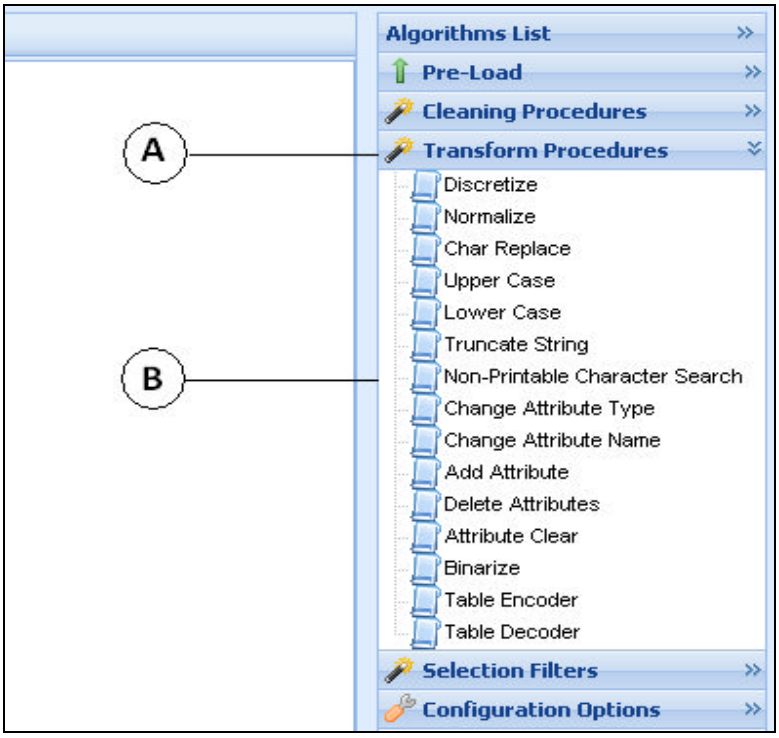
Figura A42 Resultado filtro Email Cleaner



Acción del Usuario	Respuesta del Sistema
1. El usuario aplica el filtro sobre el campo de una tabla seleccionado previamente.	2. Se despliega la ventana Email Cleaner (A) que contiene: (B): El atributo seleccionado como email. (C): La sugerencia que hace el sistema si el email no cuenta con la estructura propia de una dirección de correo, si el campo está en blanco, es porque el sistema no pudo generar una sugerencia, pero implica que el correo no tenga la estructura adecuada. (D): El botón export, que permite exportar la tabla visualizada a los formatos XLS, CSV, SQL, ARRF.

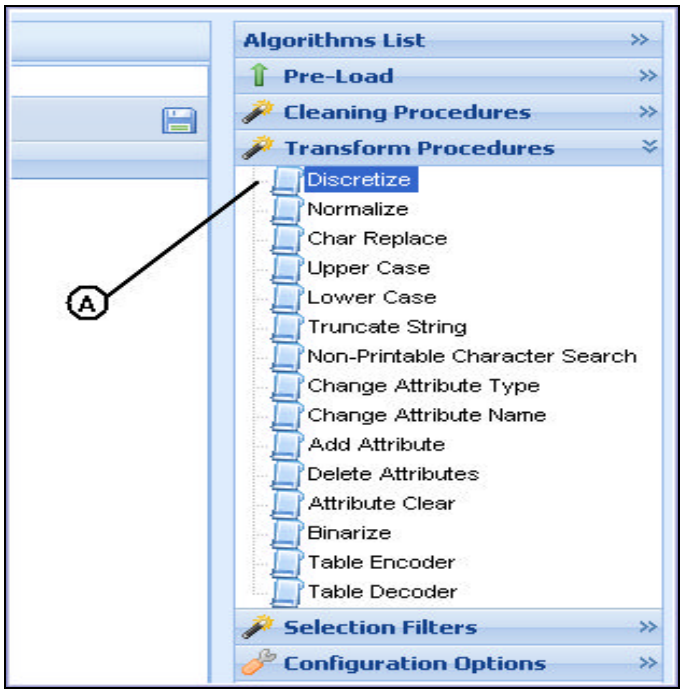
Acción del Usuario	Respuesta del Sistema
	(E): Save Table, permite guardar la tabla generada como una tabla del sistema. (F): Close, se cancela la aplicación del filtro. (G): Use All Corrections, permite reemplazar las direcciones de correo erróneas por las sugerencias del sistema.
3. El usuario hace clic en el botón Use All Corrections (G).	4. Se cambian las direcciones de correo electrónico erróneas, por las sugerencias del sistema.
5. El usuario hace clic en el botón Save Table (E)	6. El sistema solicita el nombre de la nueva tabla con el formulario Table Name.
7. El usuario hace doble clic sobre la sugerencia	8. El campo de sugerencia pasa a modo de edición, y el usuario puede modificar la sugerencia.

Figura A43 Filtros de Transformación



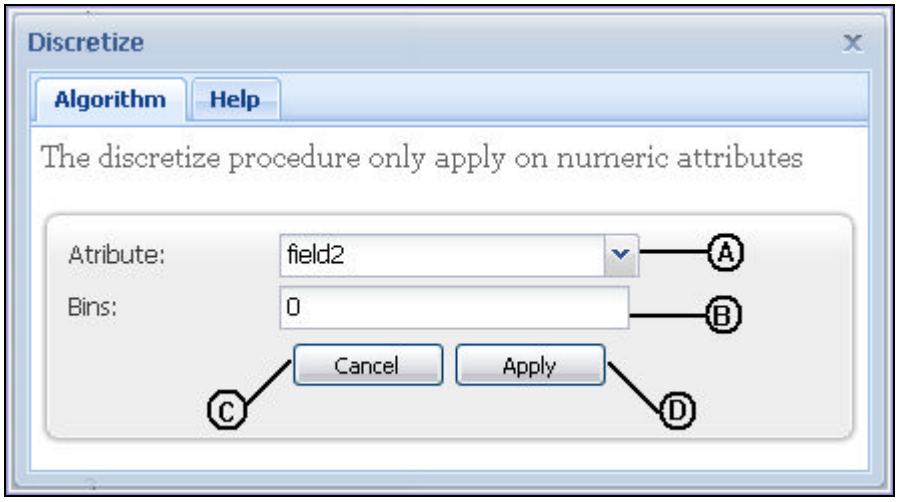
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en la pestaña Transform Procedures (A).	2. Se despliega una lista de filtros de transformación (B).

Figura A44 Filtro Discretize



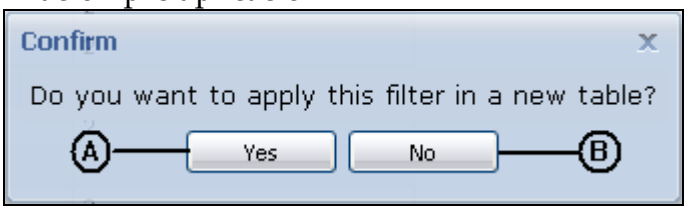
Acción del Usuario	**Respuesta del Sistema**
1. El usuario selecciona el filtro Discretize (A).	2. Se despliega la ventana Discretize.

Figura A45 Parámetros filtro Discretize



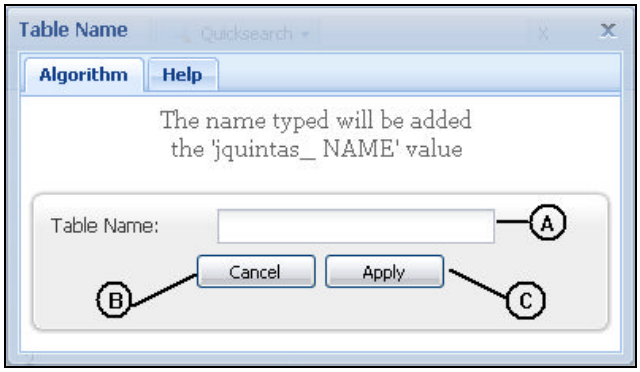
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el campo a discretizar haciendo clic en el botón (A). 2. El usuario escribe la cantidad de bins o intervalos a generar para la descretización (B), si el usuario no escribe intervalos, el sistema busca la cantidad apropiada de intervalos por fórmula. 3. El usuario hace clic en el botón Cancel (C). 5. El usuario hace clic en el botón Apply (D).	4. Se cancela la aplicación del filtro. 6. Se pasa a la ventana de confirmación para aplicar el filtro.

Figura A46 Confirmación pre aplicación



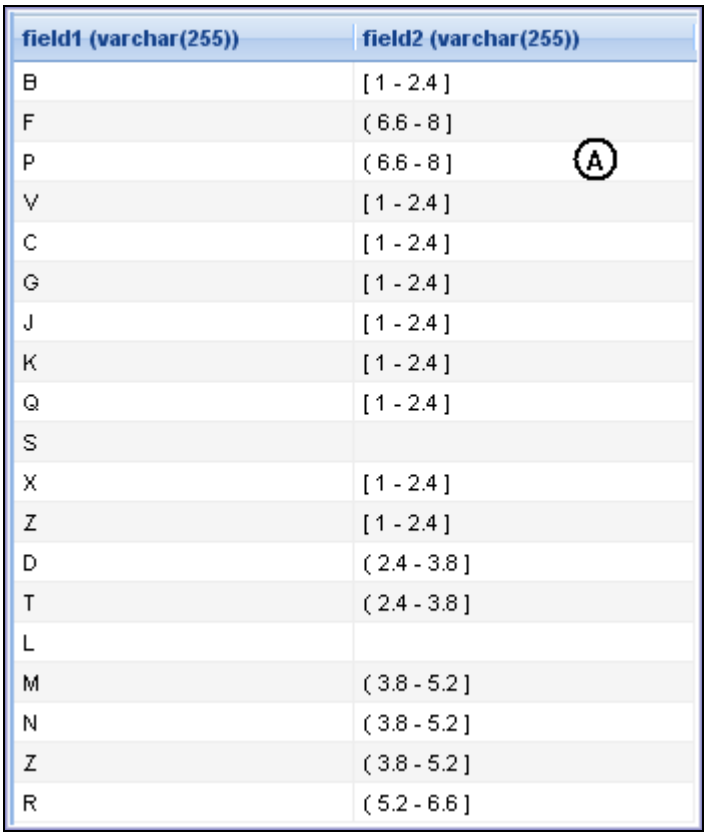
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A). 3. El usuario hace clic en el botón No (B).	2. Se despliega la ventana Table Name 4. El filtro se aplica sobre la misma tabla.

Figura A47 Table Name



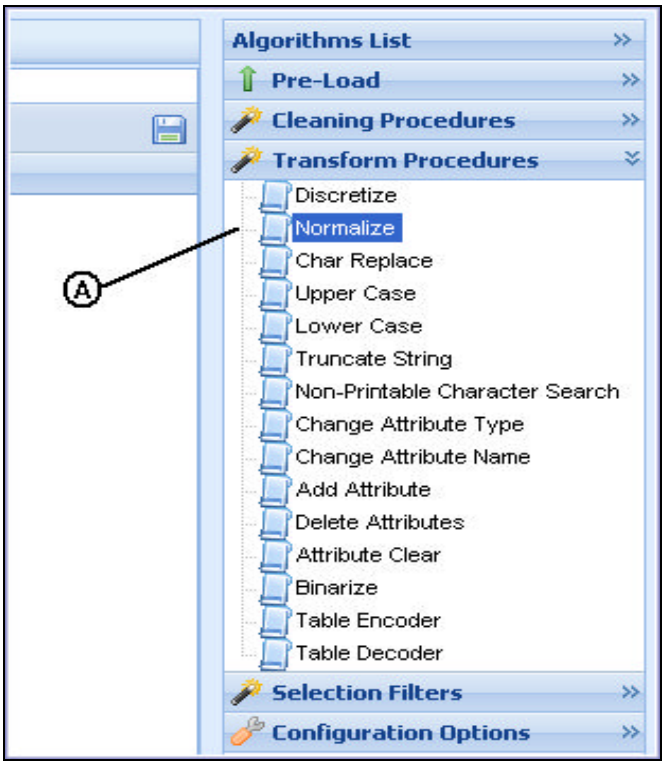
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón Cancel (B)	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas. 6. Se cancela la aplicación del filtro.

Figura A48 Aplicación del filtro Discretize



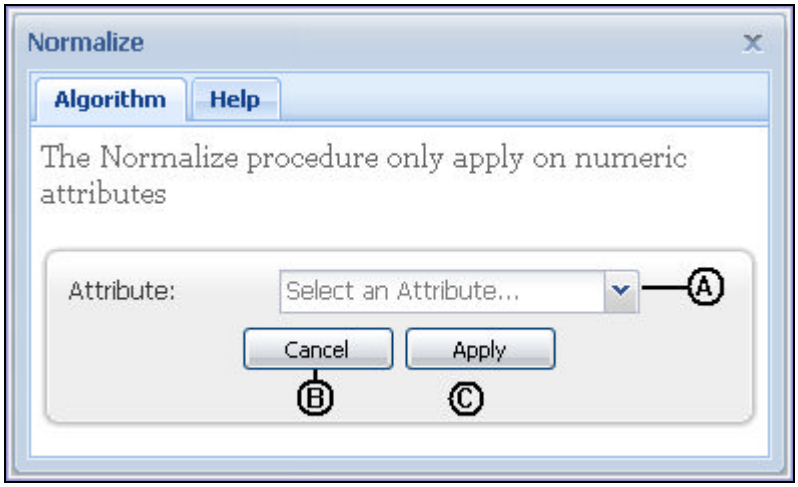
Acción del Usuario	Respuesta del Sistema
	1. Se despliega en el área de trabajo la tabla con el campo seleccionado ahora discretizado (A).

Figura A49 Filtro Normalize



Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Normalize (A).	2. Se despliega la ventana Normalize.

Figura A50 Parámetros filtro Normalize

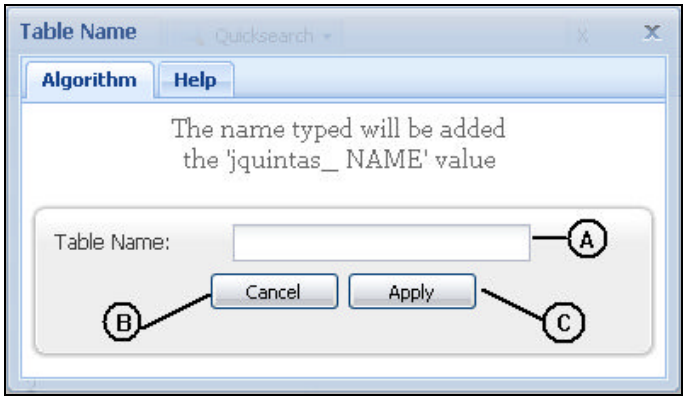


Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el campo a normalizar haciendo clic en el botón (A).	
2. El usuario hace clic en el botón Cancel (C).	3. Se cancela la aplicación del filtro.
4. El usuario hace clic en el botón Apply (D).	5. Se pasa a la ventana de confirmación para aplicar el filtro.

Figura A51 Confirmación pre aplicación

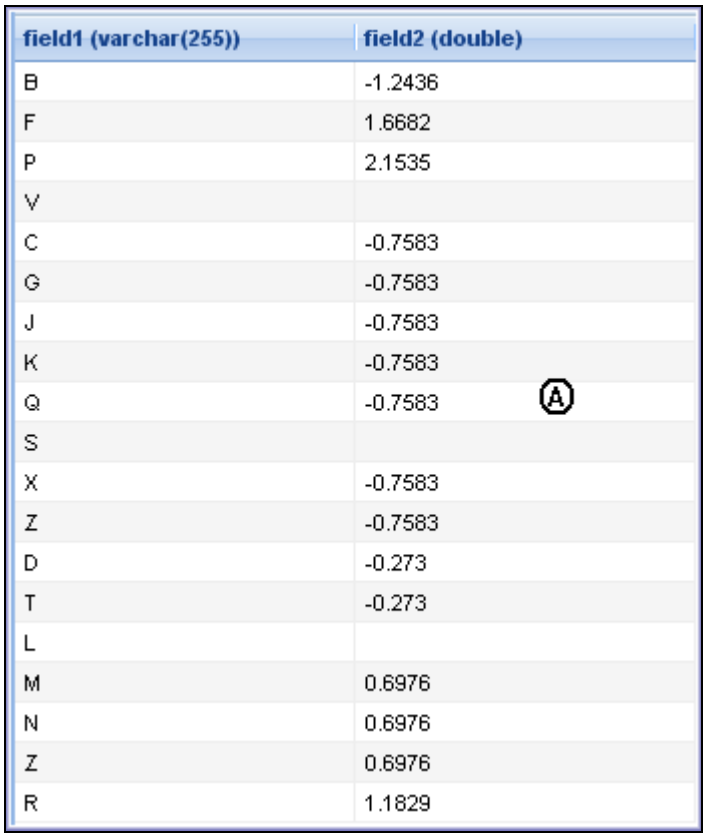
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A). 3. El usuario hace clic en el botón No (B).	2. Se despliega la ventana Table Name 4. El filtro se aplica sobre la misma tabla.

Figura A52 Table Name



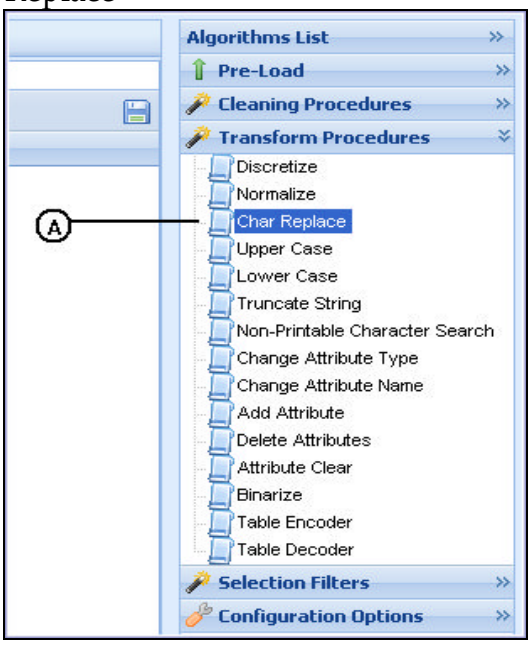
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón Cancel (B)	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas. 6. Se cancela la aplicación del filtro.

Figura A53 Aplicación del filtro Normalize



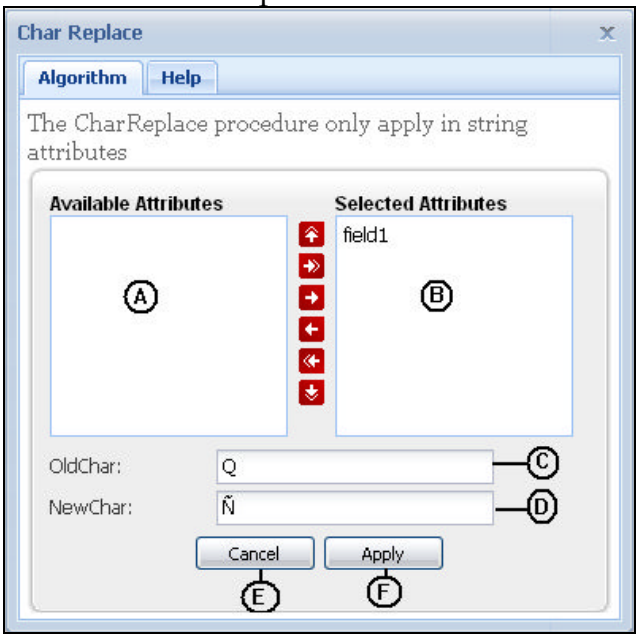
Acción del Usuario	Respuesta del Sistema
	1. Se despliega en el área de trabajo la tabla con el campo seleccionado ahora normalizado (A).

Figura A54 Filtro Char Replace



Acción del Usuario	**Respuesta del Sistema**
1. El usuario selecciona el filtro Char Replace (A).	2. Se despliega la ventana Char Replace en donde se introducen los parámetros del filtro.

Figura A55 Parámetros filtro Char Replace

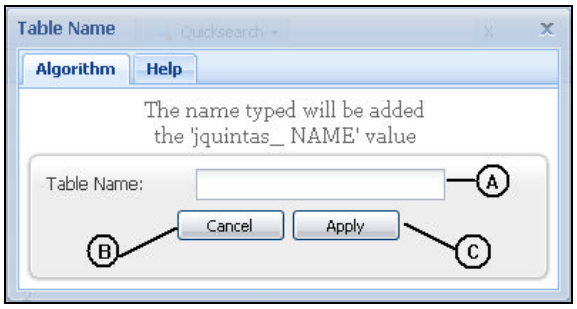


Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el o los campos objeto del filtro, haciendo clic en el campo (A) y transfiriendo los atributos al campo Selected Attributes (B)	
2. El usuario escribe el caracter o cadena de caracteres a reemplazar en el campo Old Char (C).	
3. El usuario escribe el caracter o cadena de caracteres por los cuales se va a reemplazar la cadena anterior (D).	
4. El usuario hace clic en el botón Cancel (E).	5. Se cancela la aplicación del filtro.
6. El usuario hace clic en el botón Apply (F).	7. Se pasa a la ventana de confirmación para aplicar el filtro.

Figura A56 Confirmación pre aplicación

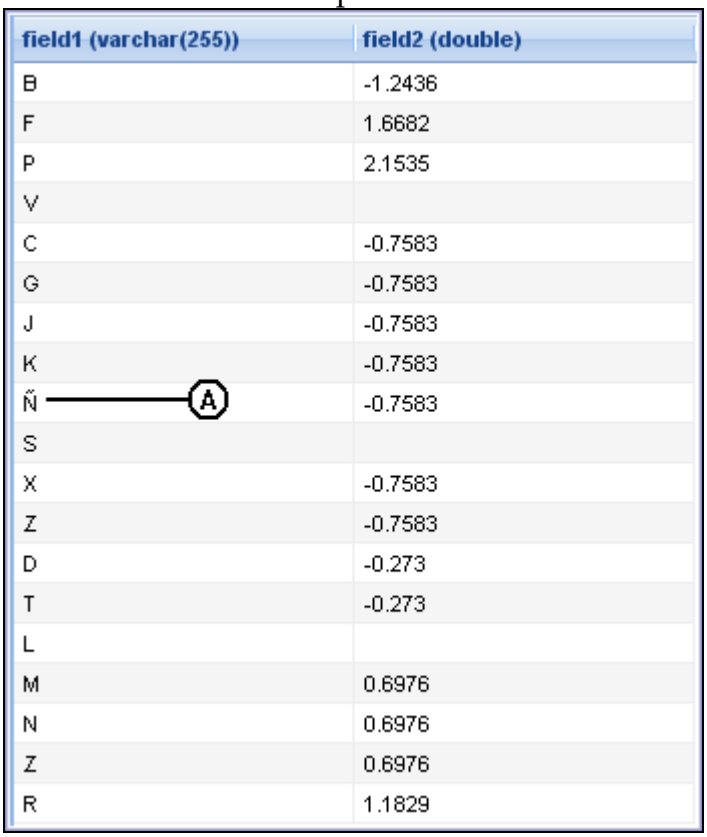
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A). 3. El usuario hace clic en el botón No (B).	2. Se despliega la ventana Table Name 4. El filtro se aplica sobre la misma tabla.

Figura A57 Table Name



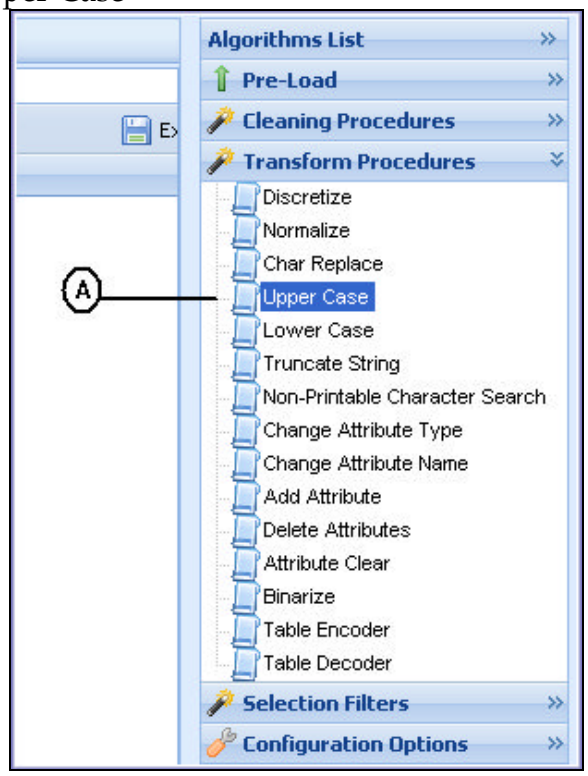
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. Hacer clic en el botón Cancel (B)	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas. 6. Se cancela la aplicación del filtro.

Figura A58 Aplicación del filtro Char Replace



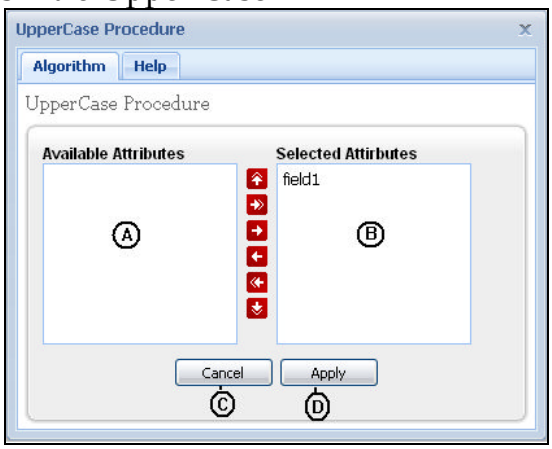
Acción del Usuario	Respuesta del Sistema
	1. Se despliega en el área de trabajo la tabla con el campo seleccionado ahora con las nuevas cadenas de caracteres como en la gráfica (A).

Figura A59 Filtro Upper Case



Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Upper Case (A).	2. Se despliega la ventana Upper Case Procedure donde se seleccionan los atributos objeto del filtro.

Figura A60 Parámetros filtro Upper Case

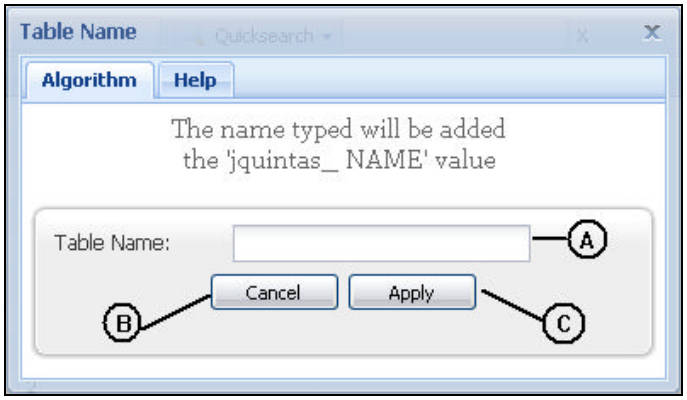


Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el o los campos objeto del filtro, haciendo clic en el campo Available Attibutes(A) y transfiriendo los atributos al campo Selected Attributes (B)	
2. El usuario hace clic en el botón Cancel (C).	3. Se cancela la aplicación del filtro.
4. El usuario hace clic en el botón Apply (D).	5. Se pasa a la ventana de confirmación para aplicar el filtro.

Figura A61 Confirmación pre aplicación

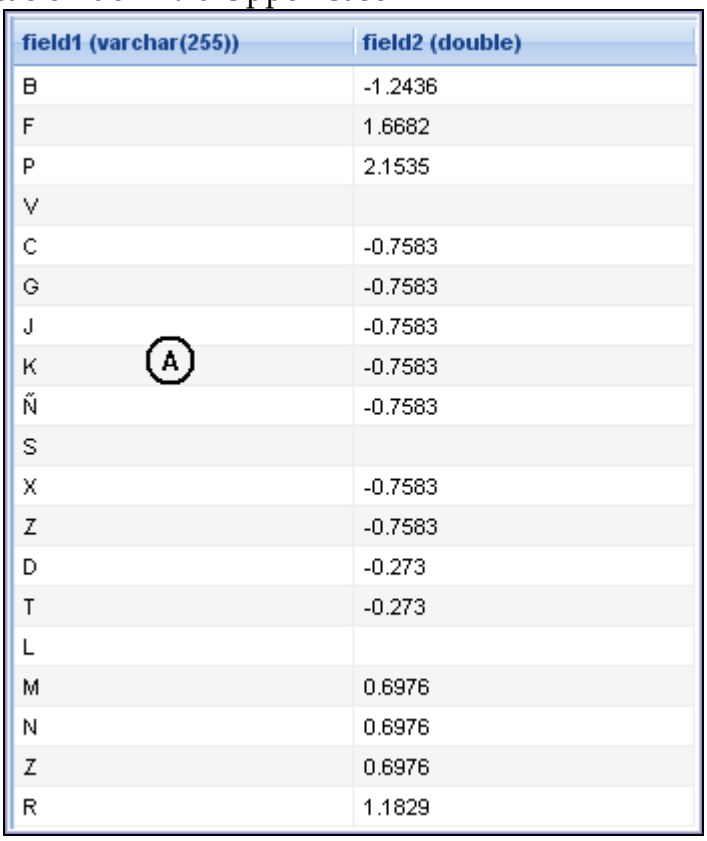
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A). 3. El usuario hace clic en el botón No (B).	2. Se despliega la ventana Table Name 4. El filtro se aplica sobre la misma tabla.

Figura A62 Table Name



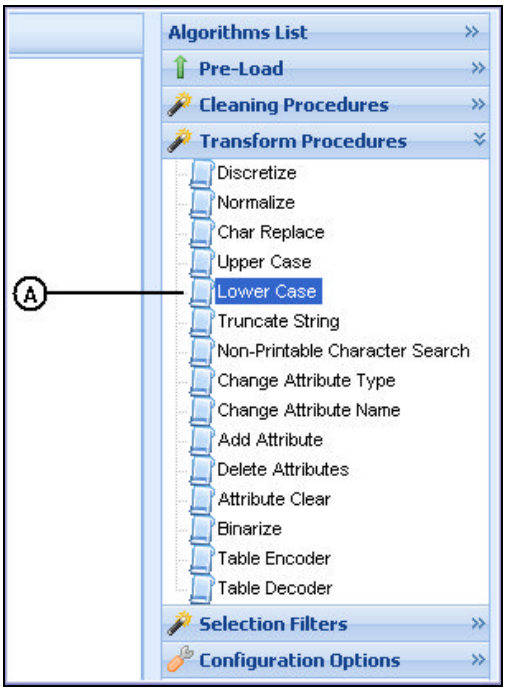
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón Cancel (B)	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas. 6. Se cancela la aplicación del filtro.

Figura A63 Aplicación del filtro Upper Case



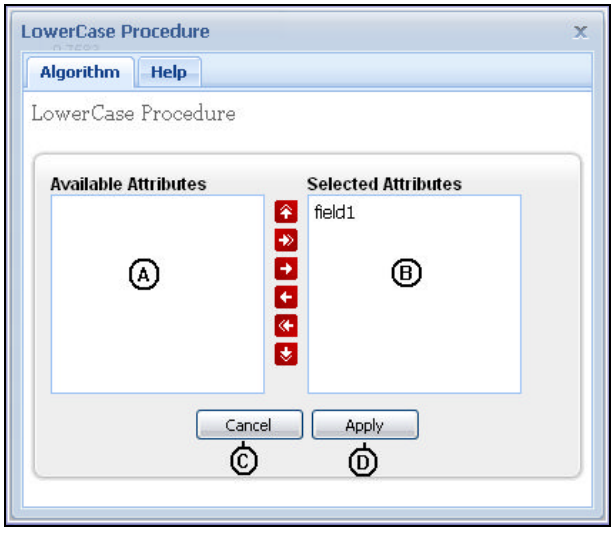
Acción del Usuario	Respuesta del Sistema
	1. Se despliega en el área de trabajo la tabla con el campo seleccionado ahora con todos los caracteres en mayúsculas.

Figura A64 Filtro Lower Case



Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Lower Case (A).	2. Se despliega la ventana Lower Case Procedure donde se seleccionan los atributos objeto del filtro.

Figura A65 Parámetros filtro Lower Case

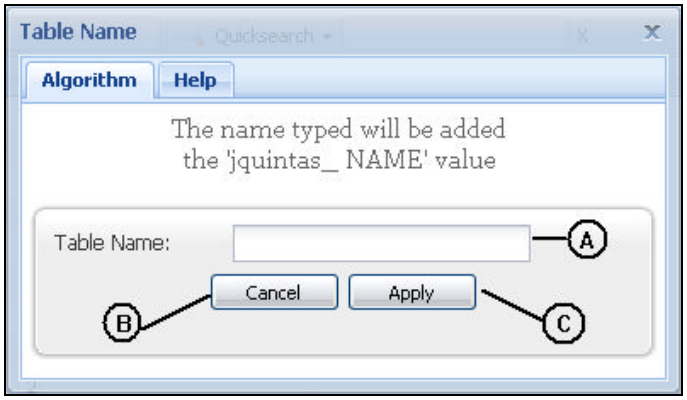


Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el o los campos objetos del filtro, haciendo clic en el campo (A) y transfiriendo los atributos al campo Selected Attributes (B)	
2. El usuario hace clic en el botón Cancel (C).	3. Se cancela la aplicación del filtro.
4. El usuario hace clic en el botón Apply (D).	5. Se pasa a la ventana de confirmación para aplicar el filtro.

Figura A66 Confirmación pre aplicación

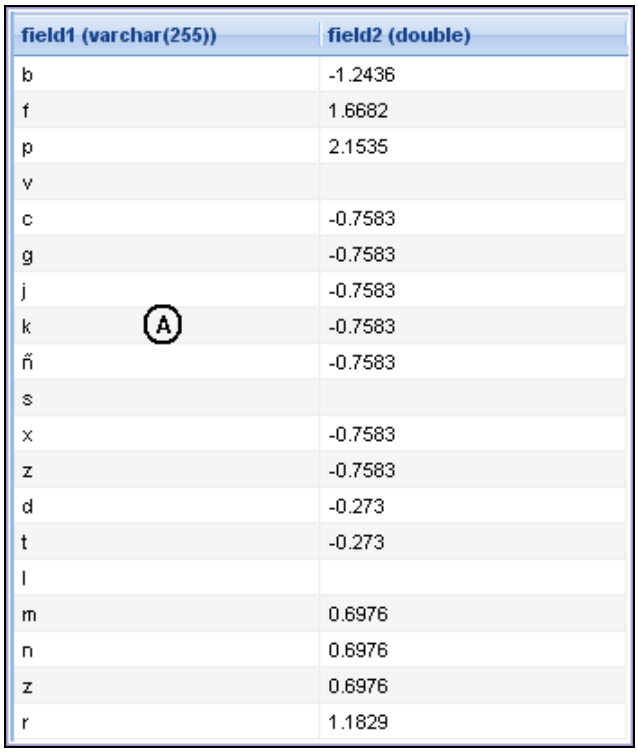
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A). 3. El usuario hace clic en el botón No (B).	2. Se despliega la ventana Table Name 4. El filtro se aplica sobre la misma tabla.

Figura A67 Table Name



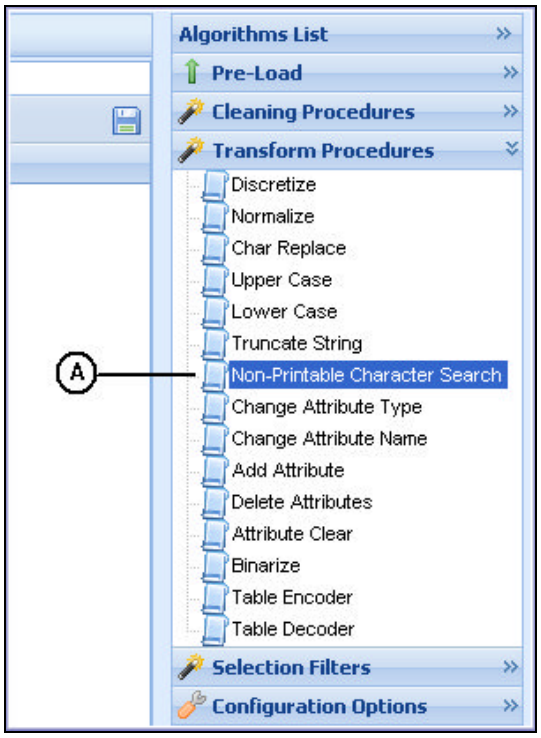
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón Cancel (B)	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas. 6. Se cancela la aplicación del filtro.

Figura A68 Aplicación del filtro Lower Case



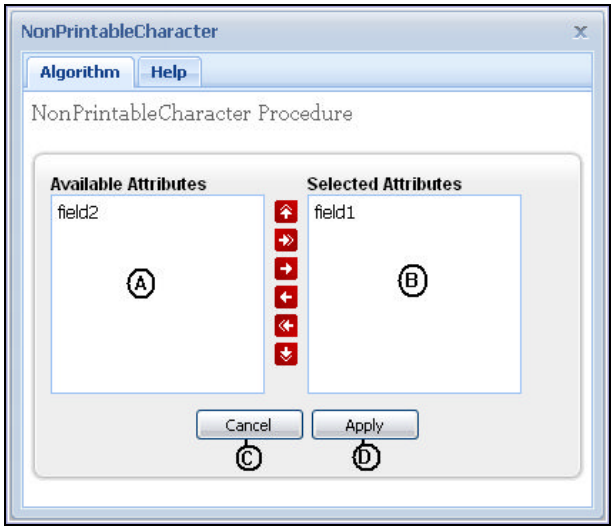
Acción del Usuario	Respuesta del Sistema
	1. Se despliega en el área de trabajo la tabla con el campo seleccionado ahora con todos los caracteres en minúsculas.

Figura A69 Filtro Non-Printable Character Search



Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Non-Printable Character Search (A).	2. Se despliega la ventana Non-Printable Character donde se seleccionan los atributos objeto del filtro.

Figura A70 Parámetros del filtro Non-Printable Character Search

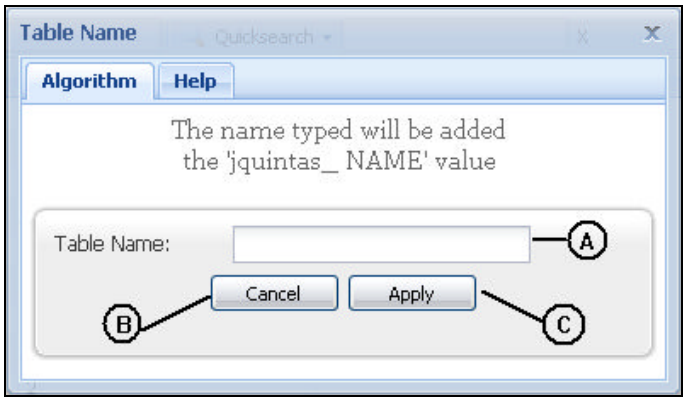


Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona los campos objetos del filtro, haciendo clic en el campo (A) y transfiriendo los atributos al campo Selected Attributes (B)	
2. El usuario hace clic en el botón Cancel (C).	3. Se cancela la aplicación del filtro.
4. El usuario hace clic en el botón Apply (D).	5. Se pasa a la ventana de confirmación para aplicar el filtro.

Figura A71 Confirmación pre aplicación

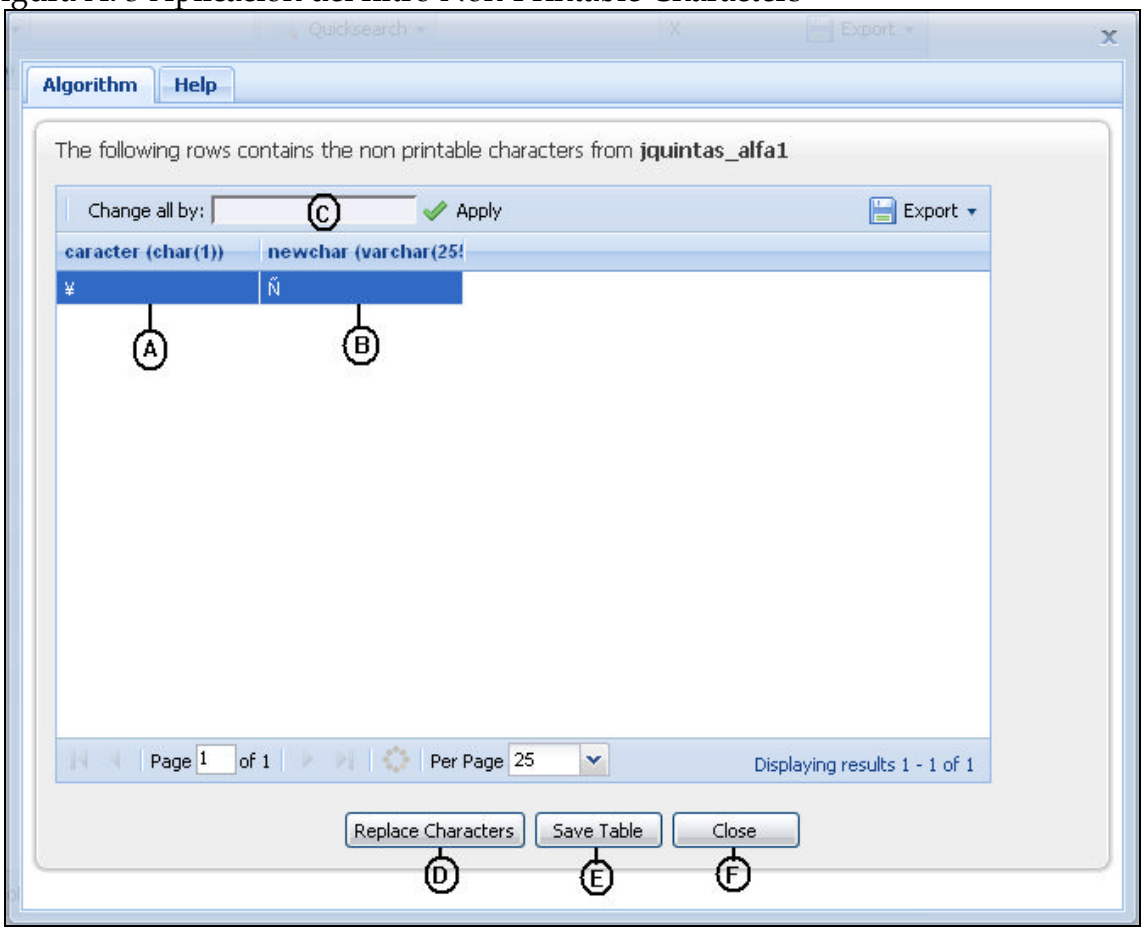
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A). 3. El usuario hace clic en el botón No (B).	2. Se despliega la ventana Table Name 4. El filtro se aplica sobre la misma tabla.

Figura A72 Table Name



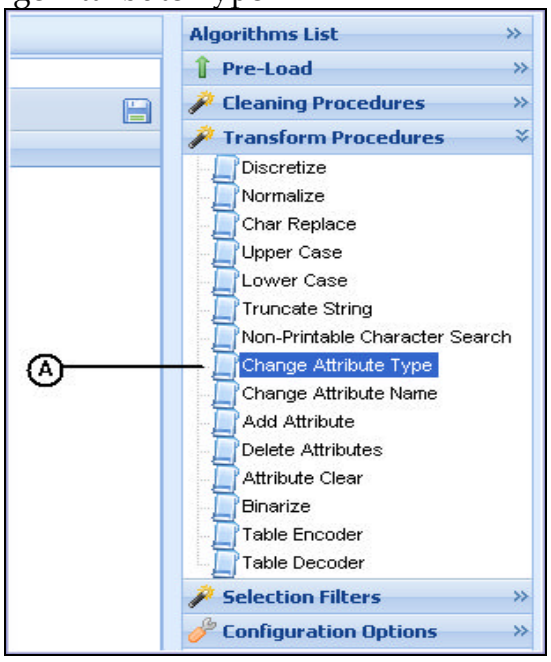
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón Cancel (B)	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas. 6. Se cancela la aplicación del filtro.

Figura A73 Aplicación del filtro Non-Printable Characters



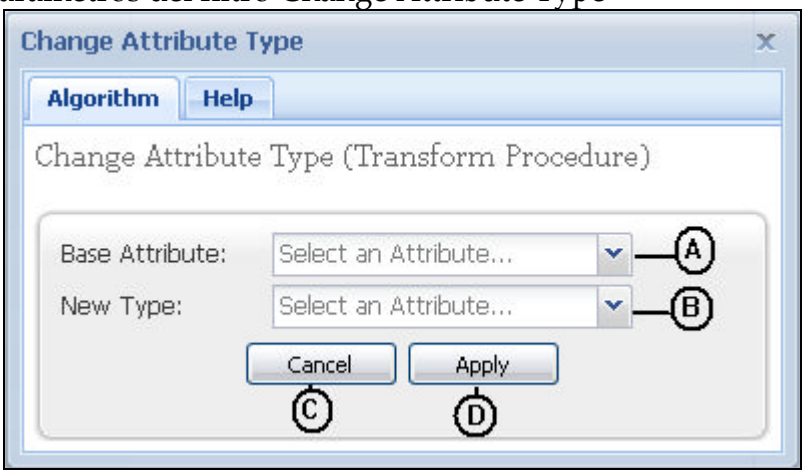
Acción del Usuario	Respuesta del Sistema
	1. Se despliega si se encuentran caracteres no imprimibles, una lista de los caracteres no imprimibles encontrados en el atributo (A), frente a cada caracter no imprimible, se encuentra un campo vacio que sirve para escribir una cadena de caracteres por la que se pueden reemplazar los caracteres encontrados (B).
2. Si el usuario hace clic en el botón Change all by (C).	3. Los caracteres se reemplazarán por la cadena escrita en el campo (C).
4. Si el usuario hace clic en el botón Replace Characters (D).	5. Se reemplazan lo caracteres por la cadena de caracteres escrita por el usuario en el campo (B).
6. Si el usuario hace clic en el botón Save Table (E).	7. El sistema solicita un nombre para la tabla y la guarda como tabla del sistema, pasa a la ventana Table Name.
8. Si el usuario hace clic en el botón Cancel (F).	9. Se cancela la aplicación del filtro.

Figura A74 Filtro Change Attribute Type



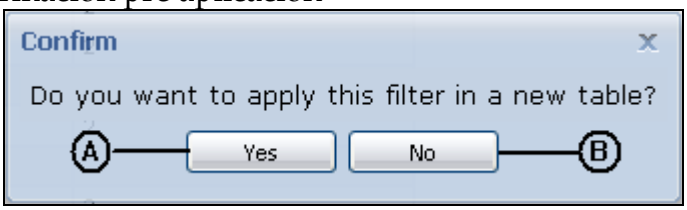
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Change Attribute type (A).	2. Se despliega la ventana Change Attribute Type donde se seleccionan los atributos objeto del filtro.

Figura A75 Parámetros del filtro Change Attribute Type



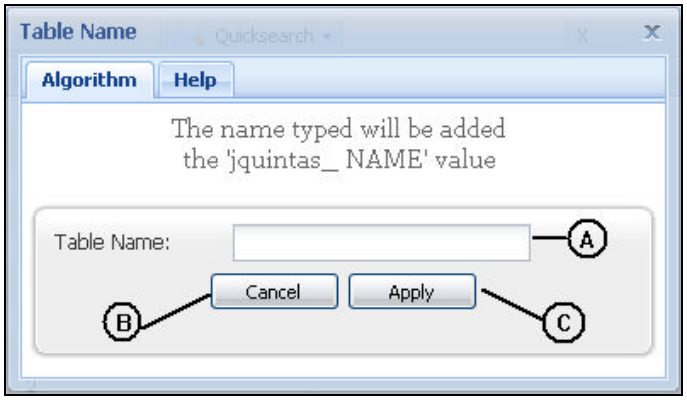
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el atributo a cambiar de tipo Base Attribute (A). 2. El usuario selecciona el nuevo tipo para el atributo seleccionado (B). 3. El usuario hace clic en el botón Cancel (C). 5. El usuario hace clic en el botón Apply (D).	4. Se cancela la aplicación del filtro. 6. Si el nuevo tipo a asignar al atributo es de tipo cadena o numérico, se pasa a la ventana de confirmación para aplicar el filtro, de lo contrario se pasa a una nueva ventana donde se define el tipo de delimitador y el formato que tiene ya sea del tipo date, datetime o time.

Figura A76 Confirmación pre aplicación



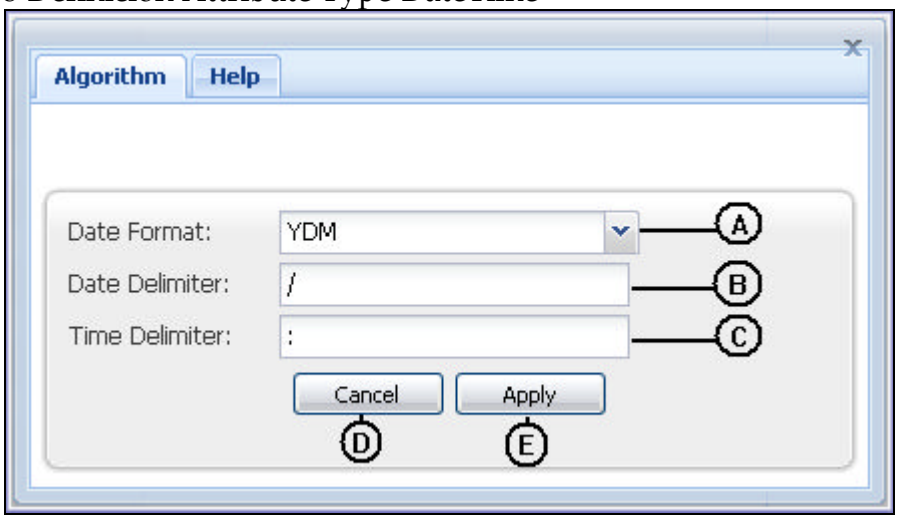
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A). 3. El usuario hace clic en el botón No (B).	2. Se despliega la ventana Table Name 4. El filtro se aplica sobre la misma tabla.

Figura A77 Table Name



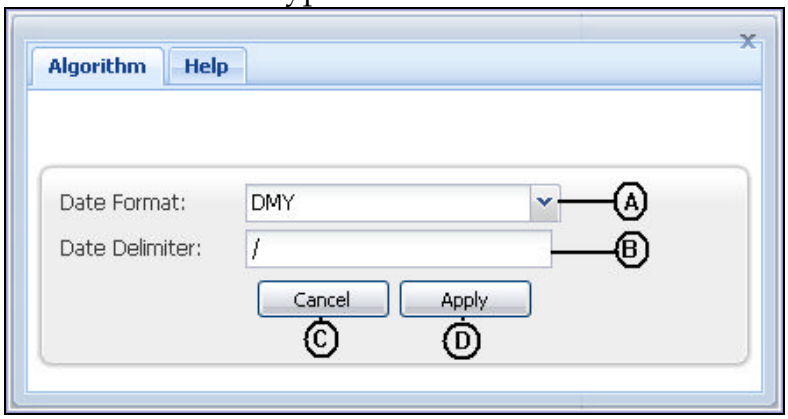
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón No (B)	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas. 6. Se cancela la aplicación del filtro.

Figura A78 Definición Attribute Type DateTime



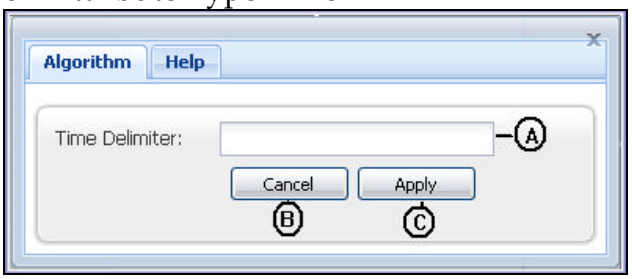
Acción del Usuario	**Respuesta del Sistema**
1. El usuario selecciona el formato de la fecha (A). 2. El usuario escribe el delimitador de fecha (B). 3. El usuario escribe el delimitador de tiempo (C). 4. El usuario hace clic en el botón Cancel (C). 6. El usuario hace clic en el botón Apply (D).	5. Se cancela la aplicación del filtro. 7. Se pasa a la ventana de confirmación para aplicar el filtro.

Figura A79 Definición Attribute Type Date



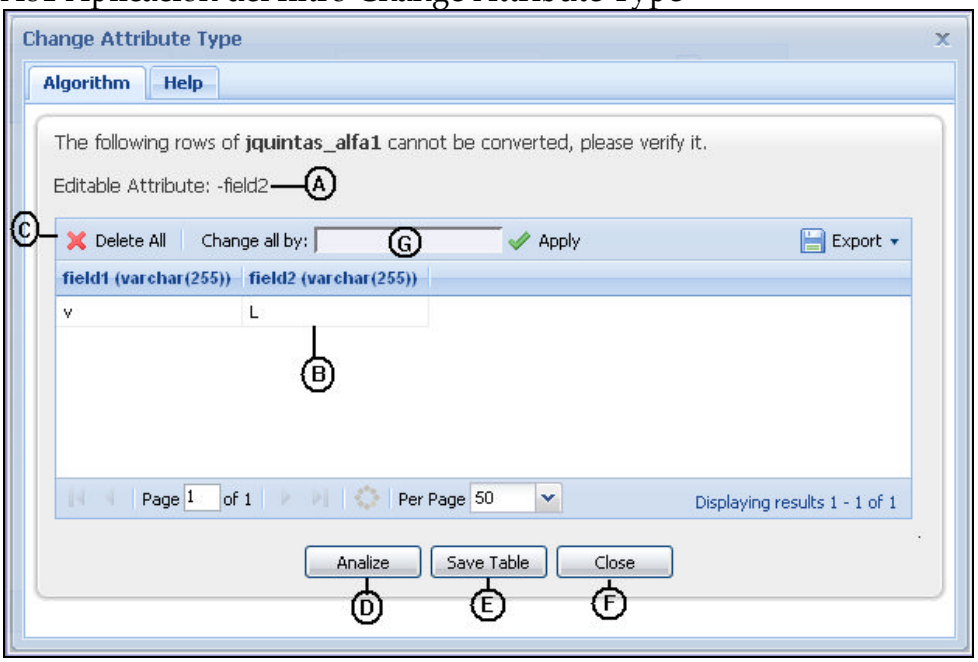
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el formato de la fecha (A). 2. El usuario escribe el delimitador de fecha (B). 3. El usuario hace clic en el botón Cancel (C). 5. El usuario hace clic en el botón Apply (D).	 4. Se cancela la aplicación del filtro. 6. Se pasa a la ventana de confirmación para aplicar el filtro.

Figura A80 Definición Attribute Type Time



Acción del Usuario	Respuesta del Sistema
1. El usuario escribe el delimitador de tiempo (B). 2. El usuario hace clic en el botón Cancel (C). 4. El usuario hace clic en el botón Apply (D).	 3. Se cancela la aplicación del filtro. 5. Se pasa a la ventana de confirmación para aplicar el filtro.

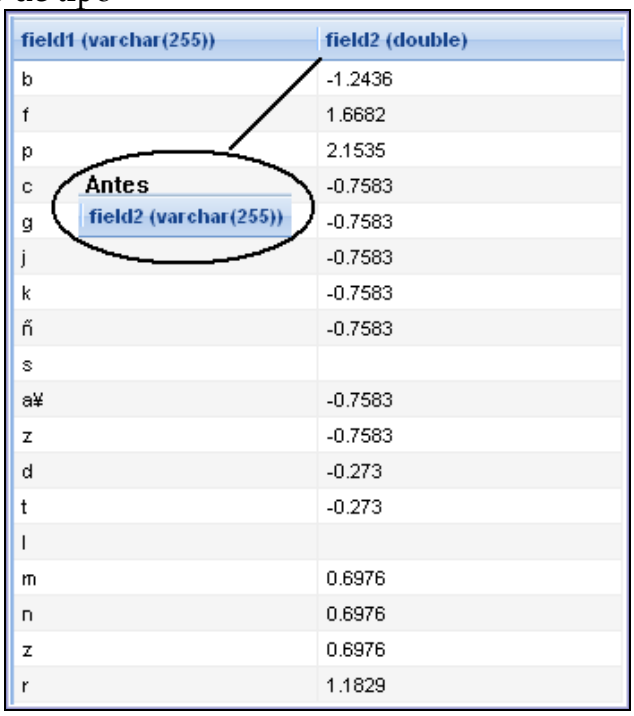
Figura A81 Aplicación del filtro Change Attribute Type



Acción del Usuario	Respuesta del Sistema
	1. Se despliega una ventana donde se encuentran: (A): Indica cuál es el campo editable, ya que el usuario tiene la posibilidad de modificar el contenido del atributo, ajustándolo a los parámetros exigidos. (B): Los registros por los cuales no se ha completado el cambio de tipo. (C): Delete All, permite eliminar todos los registros de la tabla. (D): Analize, si se ha hecho algún tipo de cambio manual se puede analizar nuevamente el dato modificado. (E): Save Table, Permite guardar la tabla. (F): Close, Cierra la ventana. (G): Change all By, remplaza los valores por una cadena de texto escrita en este campo.
2. El usuario hace clic en el campo editable determinado por (A).	3. El campo cambia de estado y se convierte en un campo editable.

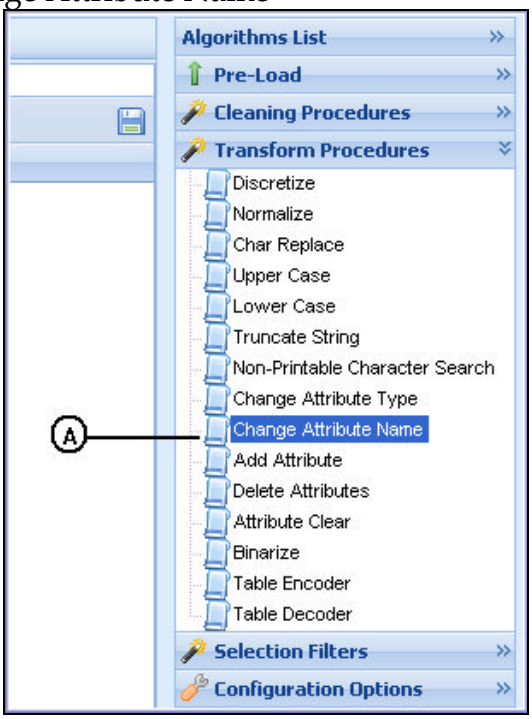
Acción del Usuario	Respuesta del Sistema
4. El usuario hace clic en el botón Delte All (C).	5. Se eliminan todos los registros de la tabla temporal, además estos se borran de la tabla original.
6. El usuario hace clic en el botón Change All By (G).	7. El sistema reemplaza los valores que no cumplen con el formato, por el valor que el usuario llene en el campo.
8. El usuario hace clic en el botón Analize (D).	9. El filtro analiza los datos, si son ahora correctos, se realiza el cambio de tipo.
10. El usuario hace clic en el botón Save Table (E).	11. Se pasa a la ventana Table name donde el usuario asigna el nombre para guardar esta tabla como del sistema.
12. El usuario hace clic en el botón Close (F).	13. Se cierra la ventana y se cancela la aplicación del filtro.

Figura A82 Cambio de tipo



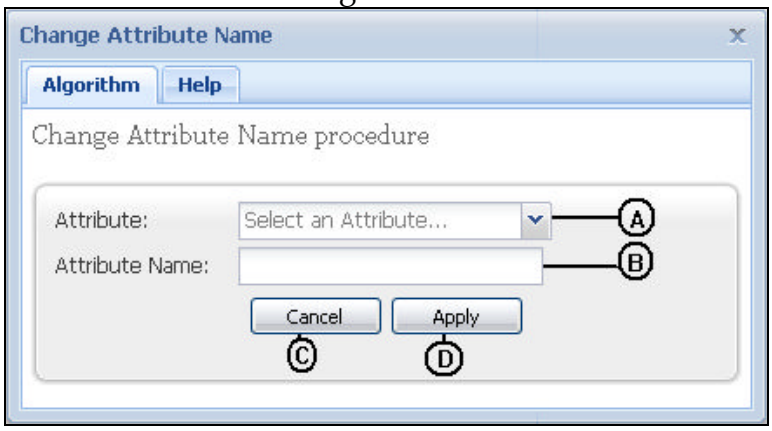
Acción del Usuario	Respuesta del Sistema
	1. Se despliega una ventana con los valores convertidos al nuevo tipo de dato, como se observa en la figura.

Figura A83 Filtro Change Attribute Name



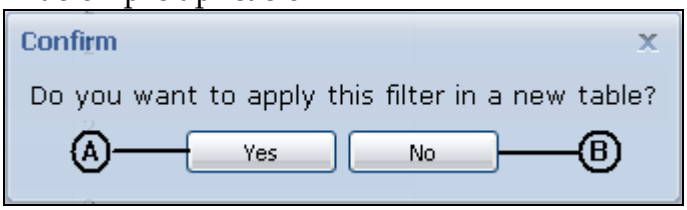
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Change Attribute Name (A).	2. Se despliega la ventana Change Attribute Name donde se establecen los parámetros del filtro.

Figura A84 Parámetros del filtro Change Attribute Name



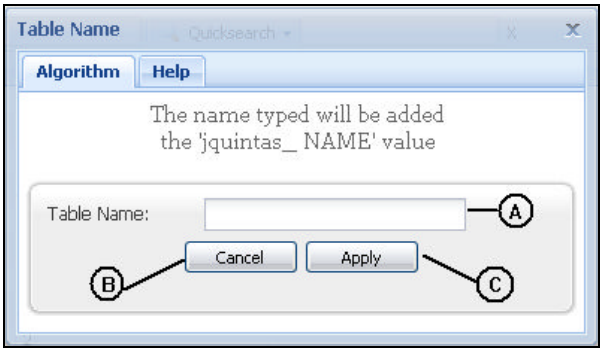
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el atributo al cual se le cambiará el nombre, en el campo Attribute (A).	
2. El usuario escribe el nuevo nombre para el atributo en el campo (B).	
3. Si el usuario hace clic en el botón Cancel (C).	4. Se cancela la aplicación del filtro.
5. Si el usuario hace clic en el botón Apply (D).	6. Se pasa a la ventana de confirmación para la aplicación del filtro.

Figura A85 Confirmación pre aplicación



Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A). 3. El usuario hace clic en el botón No (B).	2. Se despliega la ventana Table Name 4. El filtro se aplica sobre la misma tabla.

Figura A86 Table Name

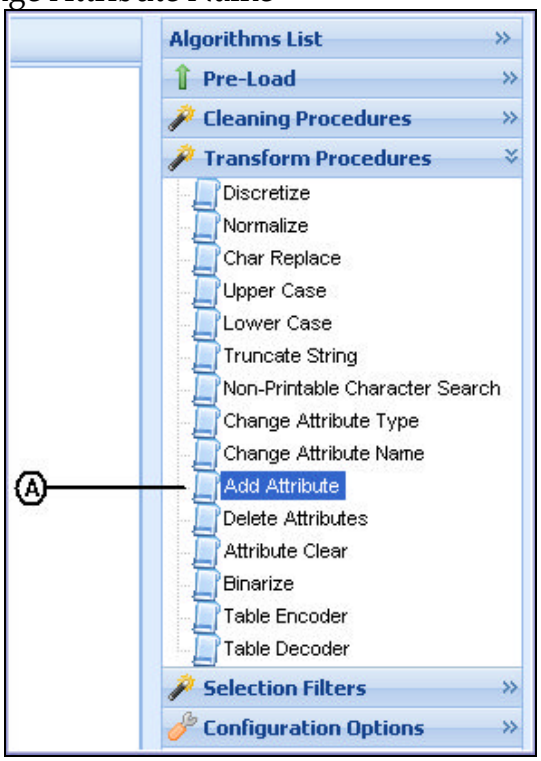


Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón Cancel (B)	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas. 6. Se cancela la aplicación del filtro.

Figura A87 Aplicación filtro Change Column Name

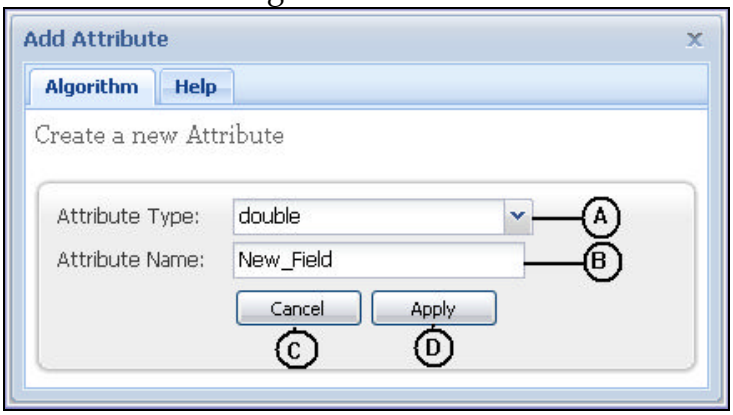
Acción del Usuario	Respuesta del Sistema
	1. Se despliega una ventana ahora con el nuevo nombre del atributo para el atributo seleccionado.

Figura A88 Filtro Change Attribute Name



Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Add Attribute (A).	2. Se despliega la ventana Add Attribute donde se establecen los parámetros del filtro.

Figura A89 Parámetros filtro Change Attribute Name

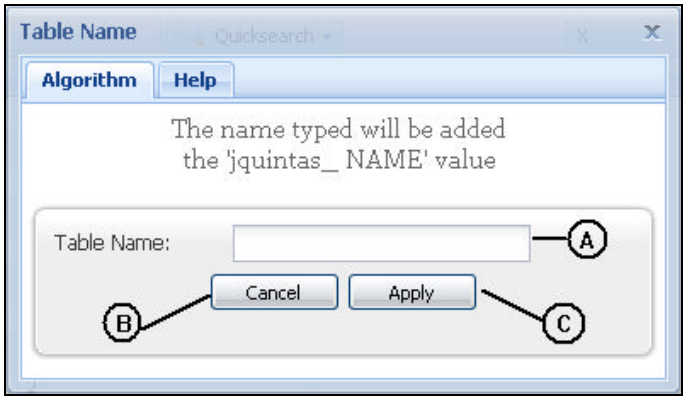


Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el tipo de dato que tendrá el nuevo atributo (A).	
2. El usuario escribe el nombre para el nuevo atributo en el campo (B).	
3. Si el usuario hace clic en el botón Cancel (C).	4. Se cancela la aplicación del filtro.
5. Si el usuario hace clic en el botón Apply (D).	6. Se pasa a la ventana de confirmación para la aplicación del filtro.

Figura A90 Confirmación pre aplicación

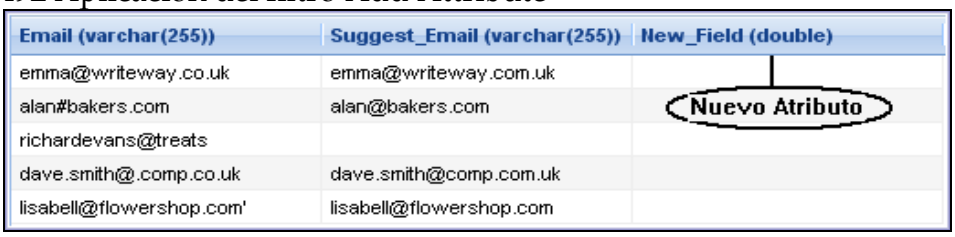
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A). 3. El usuario hace clic en el botón No (B).	2. Se despliega la ventana Table Name 4. El filtro se aplica sobre la misma tabla.

Figura A91 Table Name



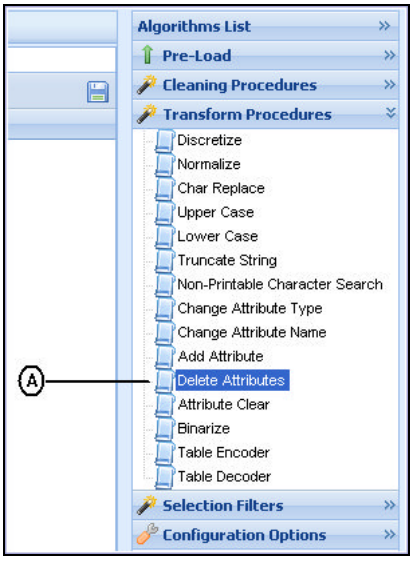
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón Cancel (B)	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas. 6. Se cancela la aplicación del filtro.

Figura A92 Aplicación del filtro Add Attribute



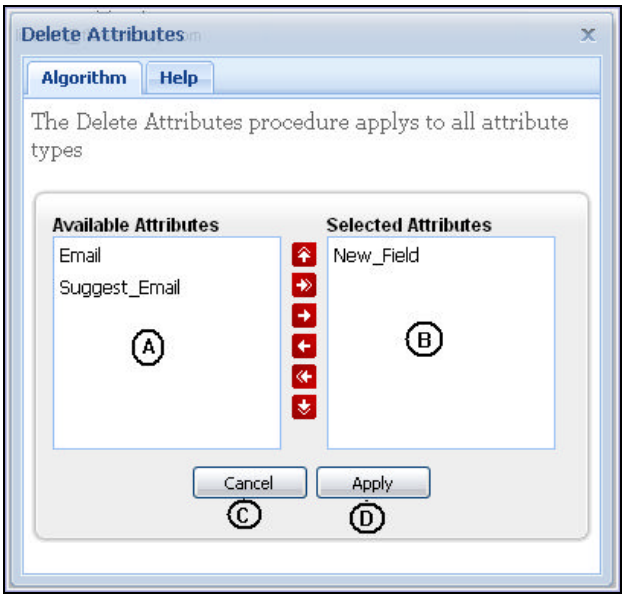
Acción del Usuario	Respuesta del Sistema
	1. Se despliega una ventana ahora con el nuevo atributo especificado por el usuario.

Figura A93 Filtro Delete Attributes



Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Delete Attributes (A).	2. Se despliega la ventana delete Atributes donde se establecen los parámetros del filtro.

Figura A94 Parámetros filtro Delete Attributes

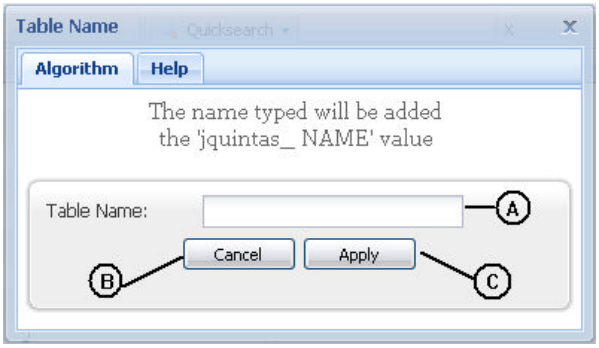


Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el o los campos a borrar, haciendo clic en el campo Available Attributes (A) y transfiriendo los atributos al campo Selected Attributes (B)	
2. El usuario hace clic en el botón Cancel (C).	3. Se cancela la aplicación del filtro.
4. El usuario hace clic en el botón Apply (D).	5. Se pasa a la ventana de confirmación para aplicar el filtro.

Figura A95 Confirmación pre aplicación

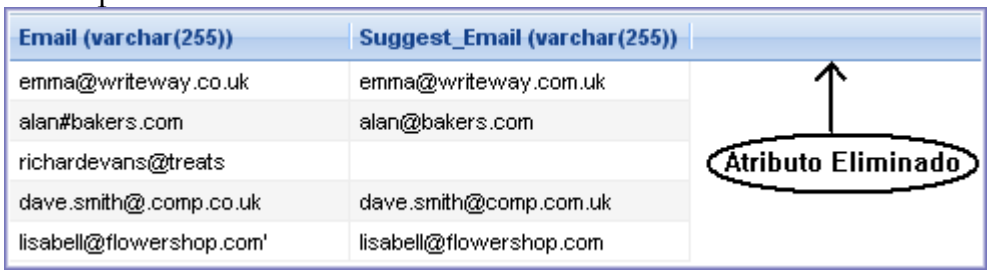
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A). 3. El usuario hace clic en el botón No (B).	2. Se despliega la ventana Table Name 4. El filtro se aplica sobre la misma tabla.

Figura A96 Table Name



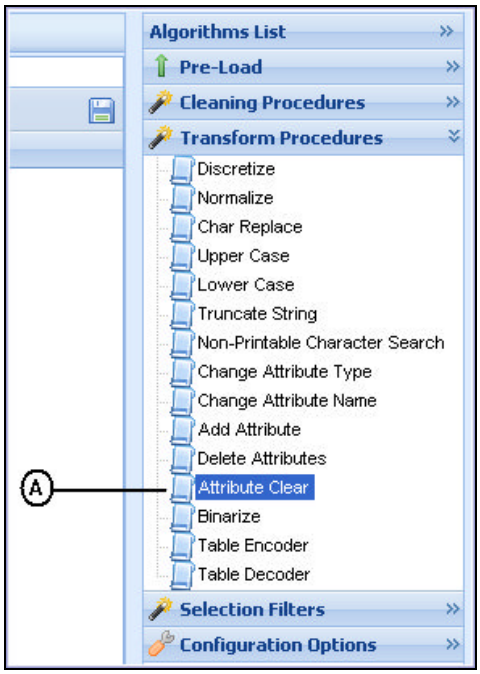
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón Cancel (B)	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas. 6. Se cancela la aplicación del filtro.

Figura A97 Aplicación del filtro Delete Attributes



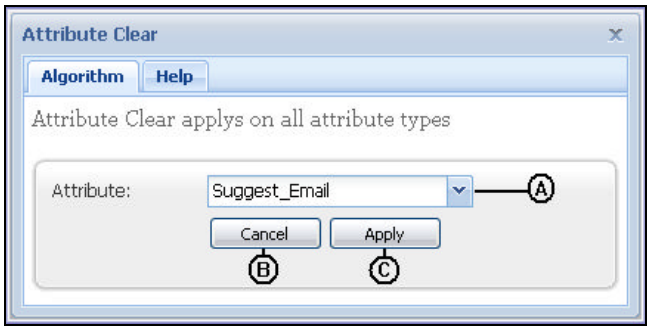
Acción del Usuario	Respuesta del Sistema
	1. Se despliega una ventana ahora sin los atributos eliminados por el usuario.

Figura A98 Filtro Clear Attribute



Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Attribute Clear (A).	2. Se despliega la ventana Attribute Clear donde se establecen los parámetros del filtro.

Figura A99 Parámetros filtro Attribute Clear

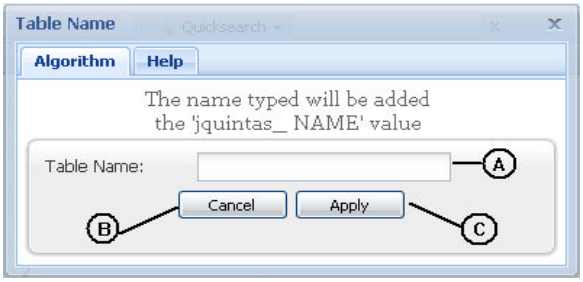


Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el atributo del cual se eliminarán todos sus registros (A).	
2. El usuario hace clic en el botón Cancel (C).	3. Se cancela la aplicación del filtro.
4. El usuario hace clic en el botón Apply (D).	5. Se pasa a la ventana de confirmación para aplicar el filtro.

Figura A100 Confirmación pre aplicación

Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A). 3. El usuario hace clic en el botón No (B).	2. Se despliega la ventana Table Name 4. El filtro se aplica sobre la misma tabla.

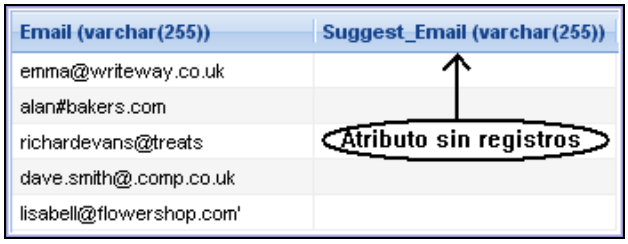
Figura A101 Table Name



Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C).	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas.

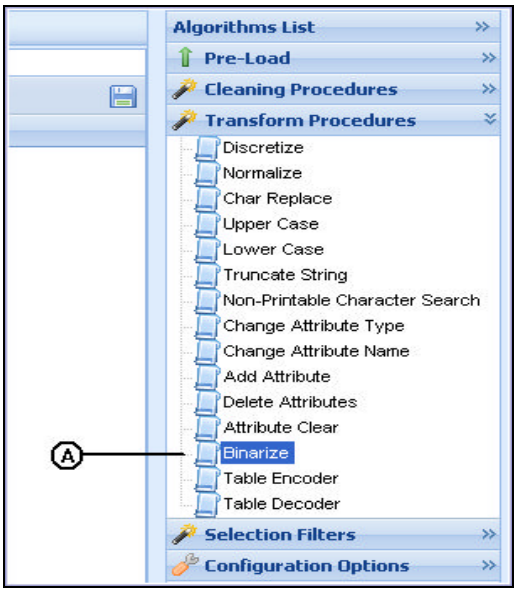
Acción del Usuario	Respuesta del Sistema
5. El usuario hace clic en el botón Cancel (B)	6. Se cancela la aplicación del filtro.

Figura A102 Aplicación del filtro Attribute Clear



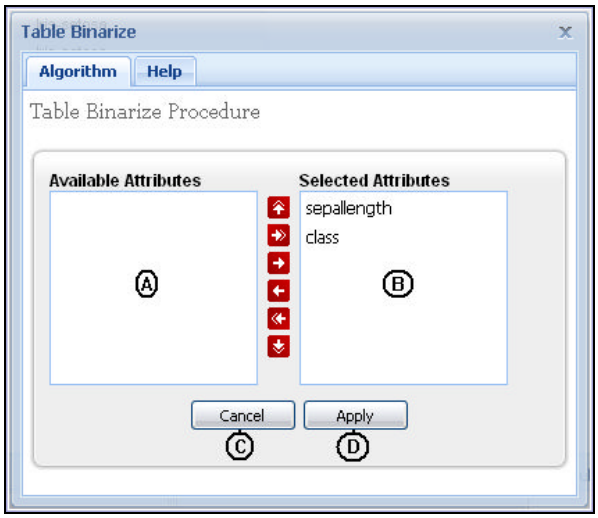
Acción del Usuario	Respuesta del Sistema
	1. Se despliega una ventana sin registros en el atributo seleccionado.

Figura A103 Filtro Binarize



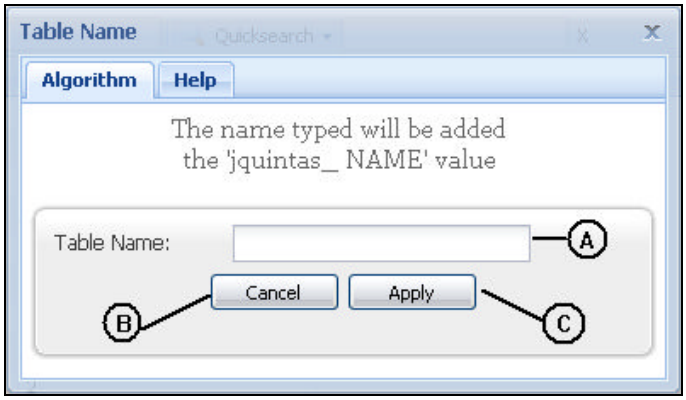
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Binarize (A).	2. Se despliega la ventana Table Binarize donde se establecen los parámetros del filtro.

Figura A104 Parámetros filtro Binarize



Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el o los campos a binarizar, haciendo clic en el campo Available Attributes(A) y transfiriendo los atributos al campo Selected Attributes (B)	
2. El usuario hace clic en el botón Cancel (C).	3. Se cancela la aplicación del filtro.
4. El usuario hace clic en el botón Apply (D).	5. Se pasa a la ventana Table name.

Figura A105 Table Name

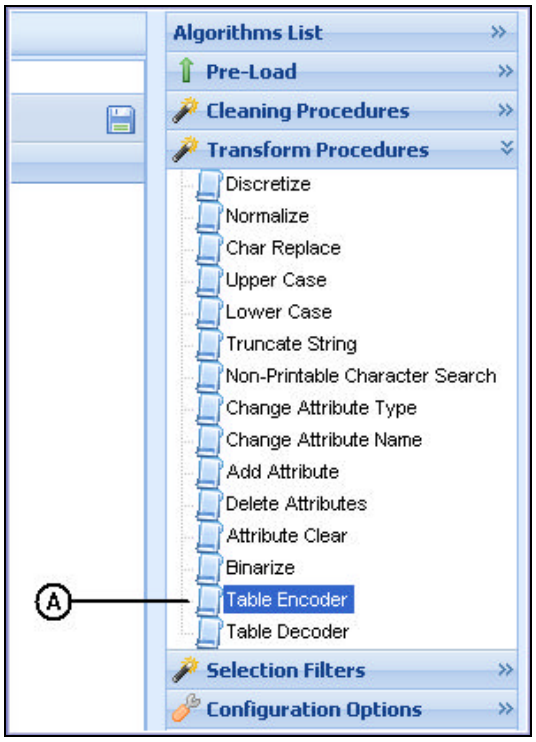


Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón Cancel (B)	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas. 6. Se cancela la aplicación del filtro.

Figura A106 Aplicación del filtro Binarize

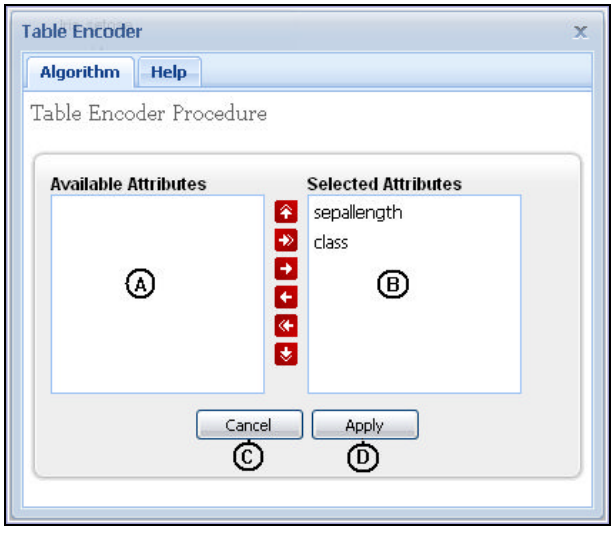
Acción del Usuario	Respuesta del Sistema
	1. Se despliega una ventana mostrando cada uno de los registros de los atributos seleccionados como atributos y donde se encuentran coincidencias se encuentra un 1 y 0 si no se encuentran coincidencias.

Figura A107 Filtro Table Enconder



Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Table Encoder (A).	2. Se despliega la ventana Table Encoder donde se establecen los parámetros del filtro.

Figura A108 Parámetros filtro Table Encoder

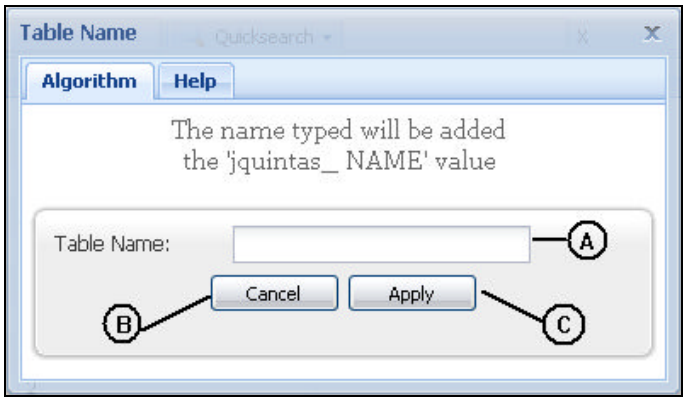


Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el o los campos a codificar, haciendo clic en el campo Available Attributes(A) y transfiriendo los atributos al campo Selected Attributes (B)	
2. El usuario hace clic en el botón Cancel (C).	3. Se cancela la aplicación del filtro.
4. El usuario hace clic en el botón Apply (D).	5. Se pasa a la ventana de confirmación para aplicar el filtro.

Figura A109 Confirmación pre aplicación

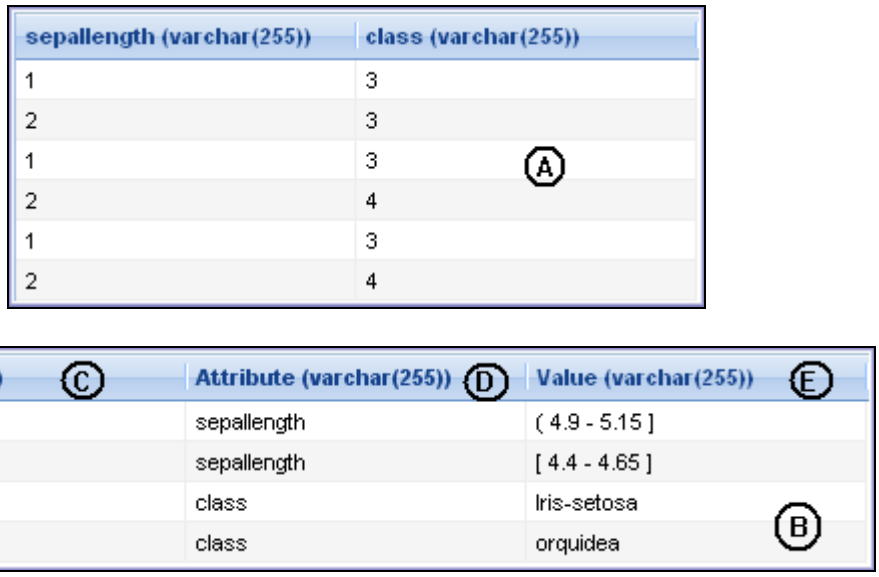
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A).	2. Se despliega la ventana Table Name
3. El usuario hace clic en el botón No (B).	4. El filtro se aplica sobre la misma tabla.

Figura A110 Table Name



Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón Cancel (B)	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas. 6. Se cancela la aplicación del filtro.

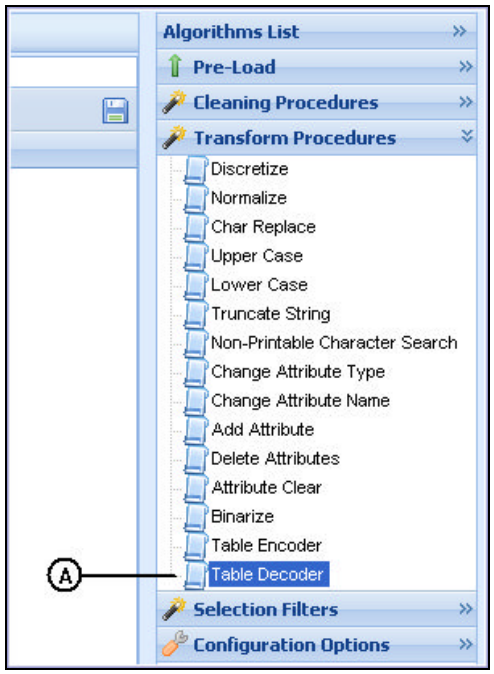
Figura A111 Aplicación del filtro Table Encoder



Acción del Usuario	Respuesta del Sistema
	1. Se despliega una ventana que muestra cada uno de los registros codificados (A), estos códigos están especificados en un diccionario de datos, que a su vez es una tabla del sistema, esta tabla se anexa al árbol de tablas.
2. El usuario selecciona la tabla diccionario de datos asociada a la tabla codificada.	3. Se despliega en el área de trabajo la tabla (B) con la descripción: (C) Code, es el código asignado a cada uno de los diferentes registros de los atributos seleccionados.

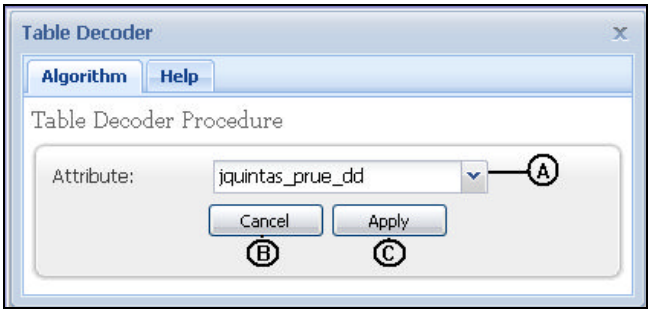
Acción del Usuario	Respuesta del Sistema
	(D) Attribute, hace referencia al atributo de donde originalmente pertenece el registro codificado. (E) Value, es el valor original a codificar.

Figura A112 Filtro Table Decoder



Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Table Decoder (A).	2. Se despliega la ventana Table Decoder donde se establecen los parámetros del filtro.

Figura A113 Parámetros filtro Table Decode

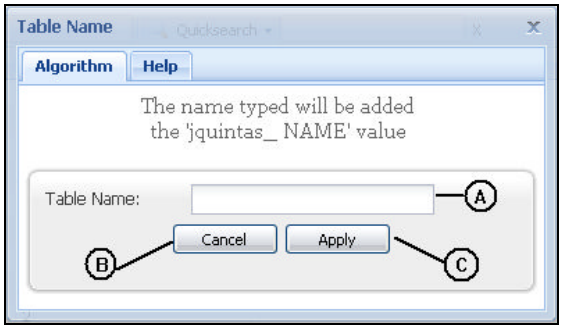


Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el diccionario de datos con el cual quiere decodificar la tabla actualmente seleccionada (A).	
2. El usuario hace clic en el botón Cancel (C).	3. Se cancela la aplicación del filtro.
4. El usuario hace clic en el botón Apply (D).	5. Se pasa a la ventana de confirmación para aplicar el filtro.

Figura A114 Confirmación pre aplicación

Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en el botón Yes (A). 3. El usuario hace clic en el botón No (B).	2. Se despliega la ventana Table Name 4. El filtro se aplica sobre la misma tabla.

Figura A115 Table Name



Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A).	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error.

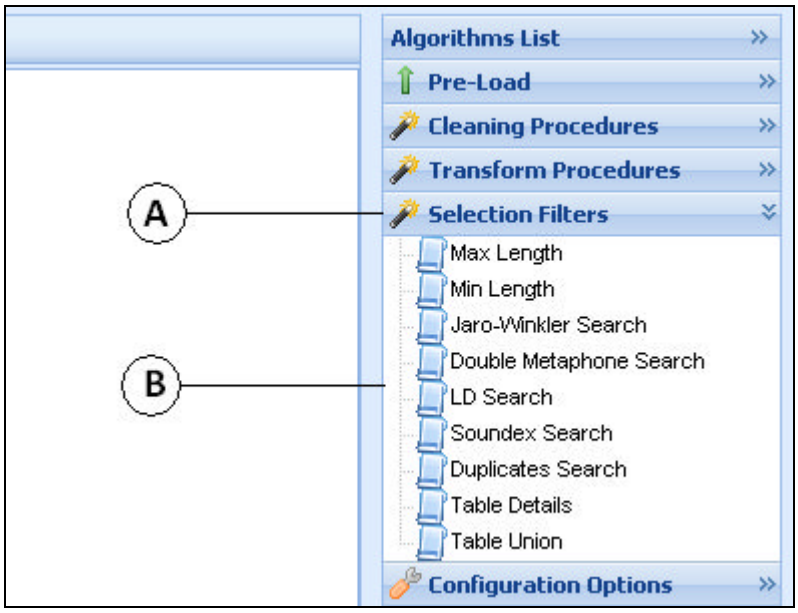
Acción del Usuario	Respuesta del Sistema
3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón Cancel (B)	4. Se crea una nueva tabla sobre la cual se aplica el filtro y se anexa al árbol de tablas. 6. Se cancela la aplicación del filtro.

Figura A116 Aplicación del filtro Table Decoder

sepallength (varchar(255))	class (varchar(255))
(4.9 - 5.15]	Iris-setosa
[4.4 - 4.65]	Iris-setosa
(4.9 - 5.15]	Iris-setosa
[4.4 - 4.65]	orquidea
(4.9 - 5.15]	Iris-setosa
[4.4 - 4.65]	orquidea

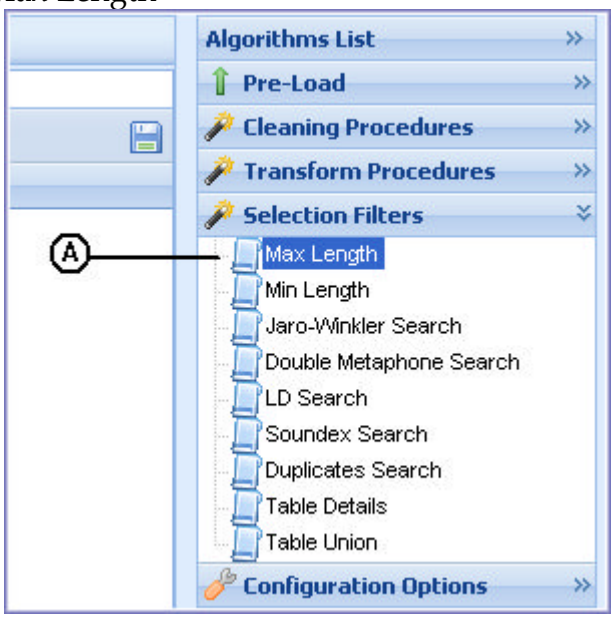
Acción del Usuario	Respuesta del Sistema
	1. Se despliega una ventana mostrando los valores originales antes de la codificación.

Figura A117 Filtros de Selección



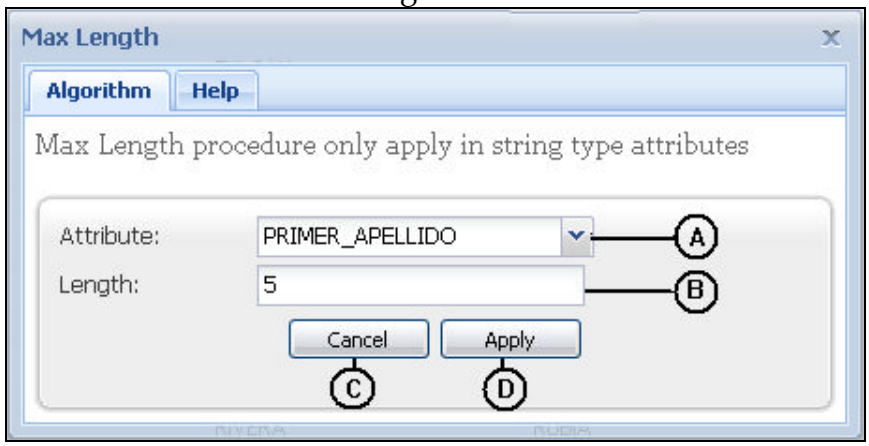
Acción del Usuario	Respuesta del Sistema
1. El usuario hace clic en la pestaña Selection Filters (A).	2. Se despliega una lista de filtros de selección (B).

Figura A118 Filtro Max Length



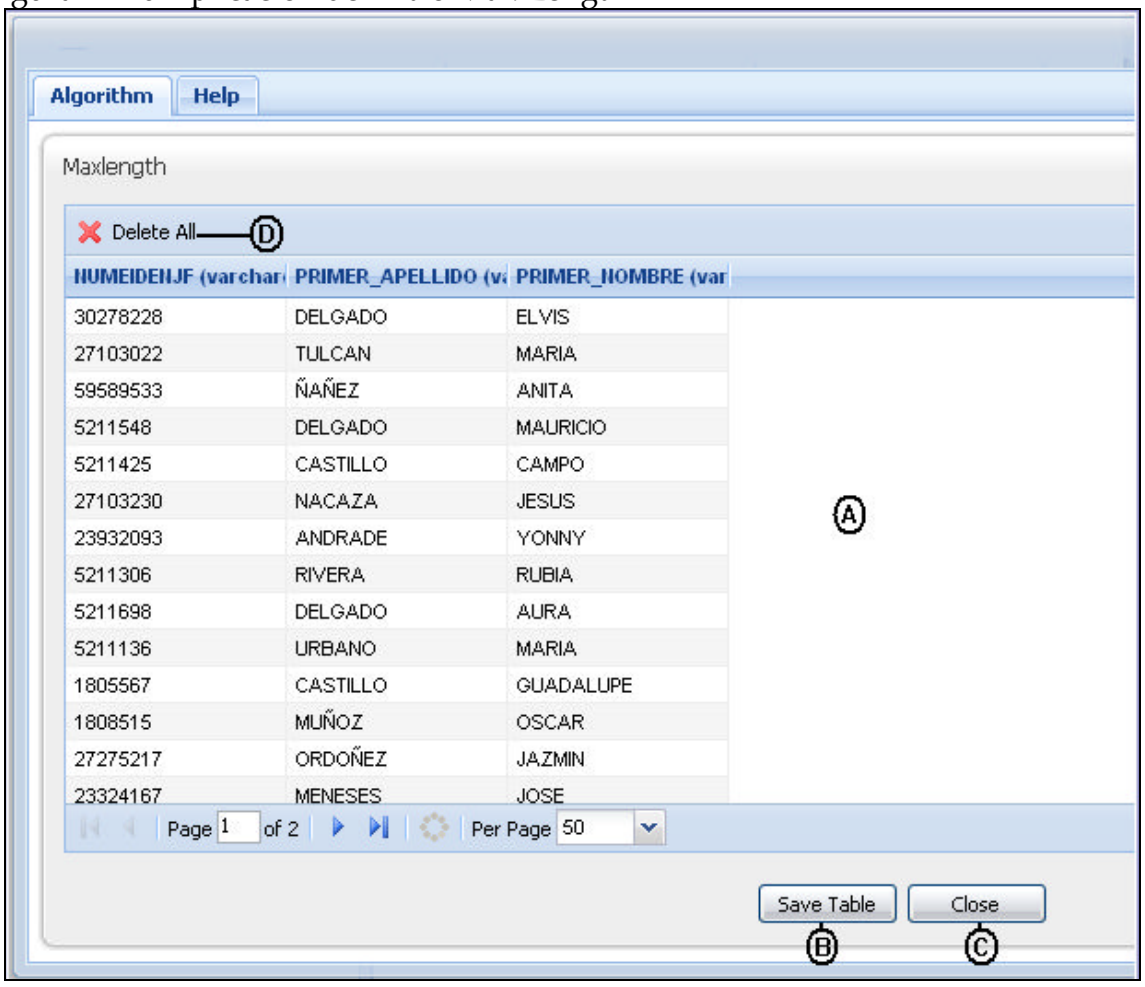
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Max Length (A).	2. Se despliega la ventana Max Length donde se establecen los parámetros del filtro.

Figura A119 Parámetros filtro Max Length



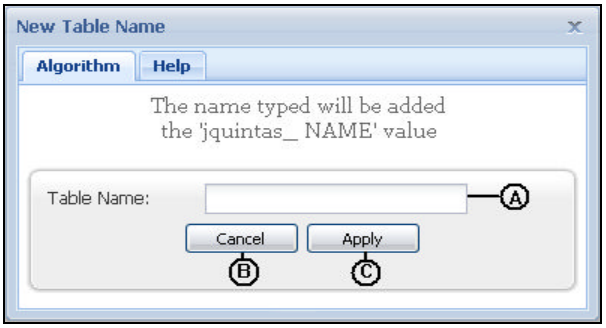
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el atributo objeto del filtro (A). 2. El usuario escribe el tamaño máximo que puede tener una cadena en el atributo seleccionado. 3. El usuario hace clic en el botón Cancel (C). 5. El usuario hace clic en el botón Apply (D).	4. Se cancela la aplicación del filtro. 6. Se aplica el filtro e indica los resultados en la ventana MaxLength.

Figura A120 Aplicación del filtro Max Length



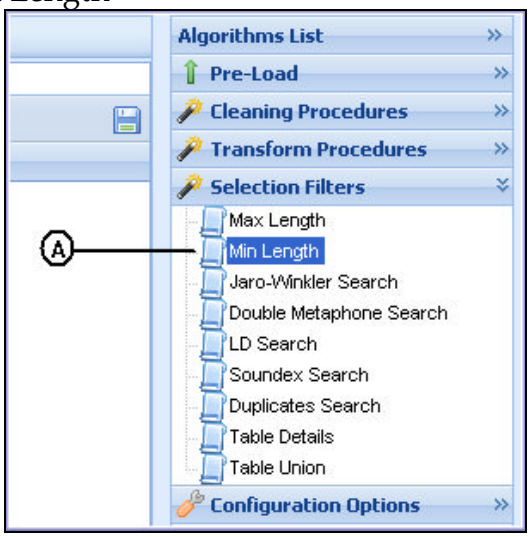
Acción del Usuario	Respuesta del Sistema
	1. Se muestra una ventana que contiene un listado de los registros que en el atributo seleccionado no cumplen con el tamaño estipulado. Esta ventana se compone de: (A) Listado de registros. (B) Save Table, permite guardar la tabla como una propia del sistema. (C) Cierra la ventana de visualización. (D) Permite eliminar todos los registros.
2. El usuario hace clic sobre el registro.	3. Se cambia a modo de edición.
4. El usuario hace clic en el botón Save Table (B).	5. Se pasa a la ventana New Table Name, donde se especifica el nombre de la nueva tabla.
6. El usuario hace clic en el botón Close (C).	7. Se cierra la ventana emergente y se cancela la aplicación del filtro.

Figura A121 New Table Name



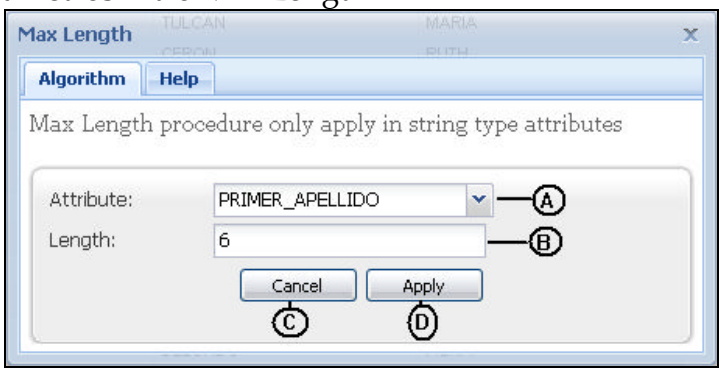
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A).	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error.
3. El usuario hace clic en el botón Apply (C).	4. Se crea una nueva tabla con los atributos en el orden especificado por el usuario y se anexa al árbol de tablas.
5. El usuario hace clic en el botón Cancel (B)	6. Se cancela la acción guardar tabla.

Figura A122 Filtro Min Length



Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Min Length (A).	2. Se despliega la ventana Min Length donde se establecen los parámetros del filtro.

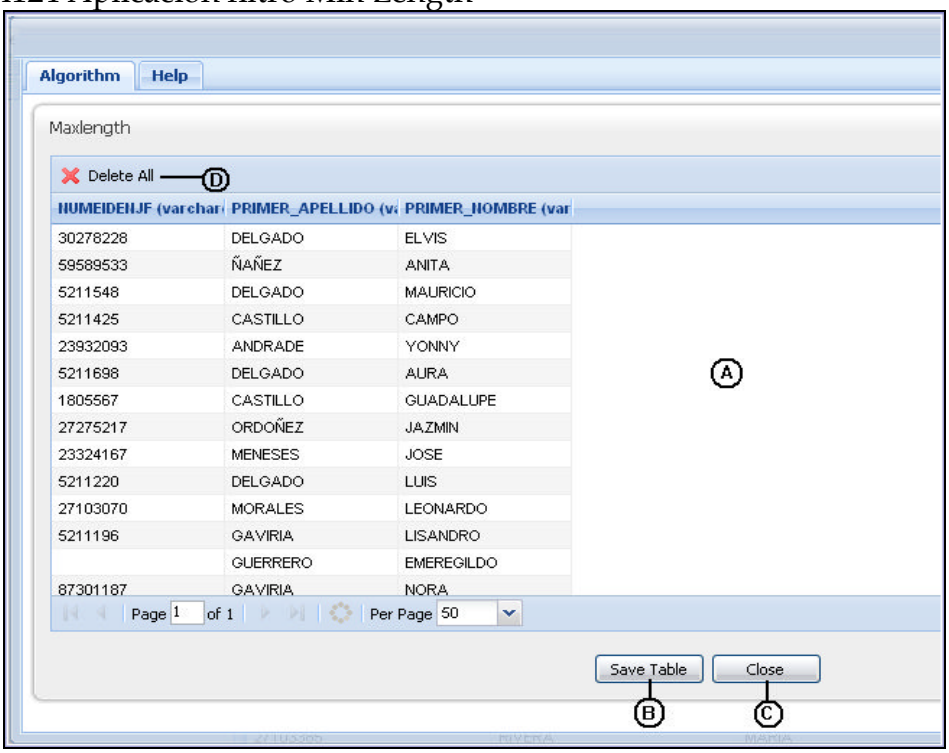
Figura A123 Parámetros filtro Min Length



Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el atributo objeto del filtro (A).	
2. El usuario escribe el tamaño mínimo que puede tener una cadena en el atributo seleccionado.	

Acción del Usuario	Respuesta del Sistema
3. El usuario hace clic en el botón Cancel (C).	4. Se cancela la aplicación del filtro.
5. El usuario hace clic en el botón Apply (D).	6. Se aplica el filtro e indica los resultados en la ventana MinLength.

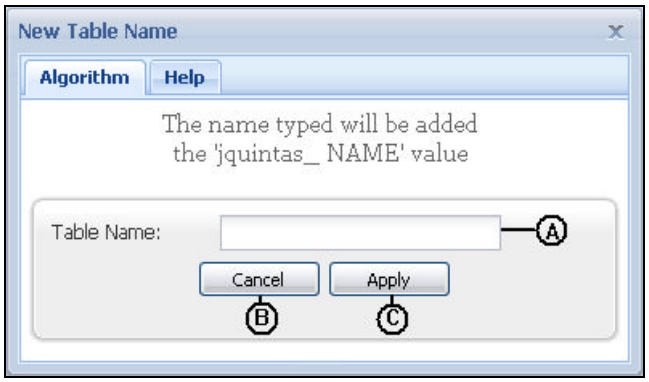
Figura A124 Aplicación filtro Min Length



Acción del Usuario	Respuesta del Sistema
	1. Se muestra una ventana que contiene un listado de los registros que en el atributo seleccionado no cumplen con el tamaño estipulado. Esta ventana se compone de: (A) Listado de registros. (B) Save Table, permite guardar la tabla como una propia del sistema. (C) Cierra la ventana de visualización. (D) Permite eliminar todos los registros.

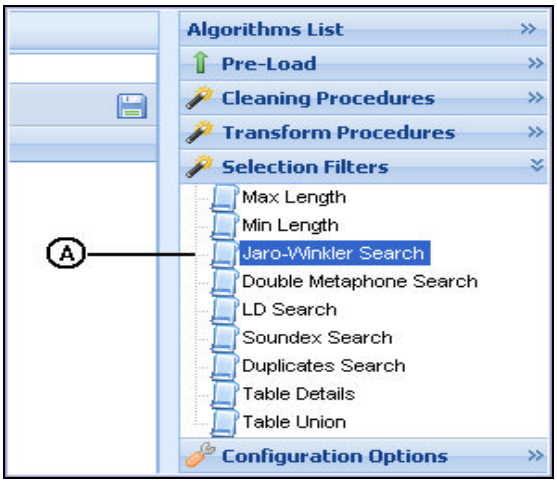
Acción del Usuario	Respuesta del Sistema
2. El usuario hace clic sobre el registro.	3. Se cambia a modo de edición.
4. El usuario hace clic en el botón Save Table (B).	5. Se pasa a la ventana New Table Name, donde se especifica el nombre de la nueva tabla.
6. El usuario hace clic en el botón Close (C).	7. Se cierra la ventana emergente y se cancela la aplicación del filtro.

Figura A125 New Table Name



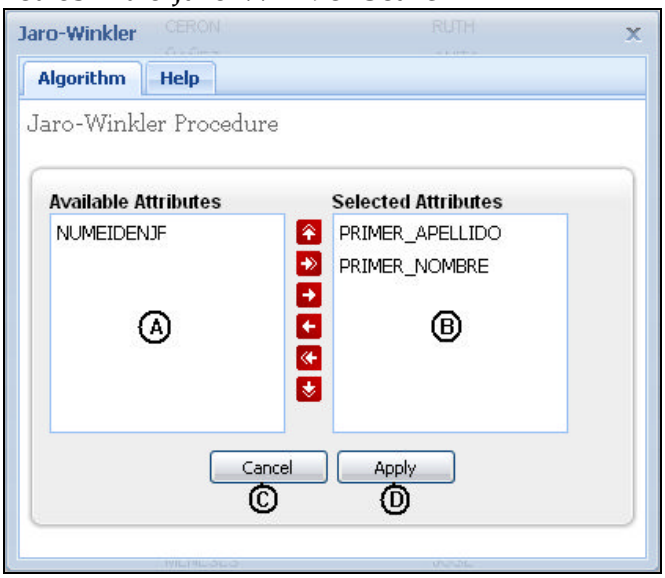
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A).	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error.
3. El usuario hace clic en el botón Apply (C).	4. Se crea una nueva tabla con los atributos en el orden especificado por el usuario y se anexa al árbol de tablas.
5. El usuario hace clic en el botón Cancel (B)	6. Se cancela la acción guardar tabla.

Figura A126 Filtro Jaro-Winkler Search



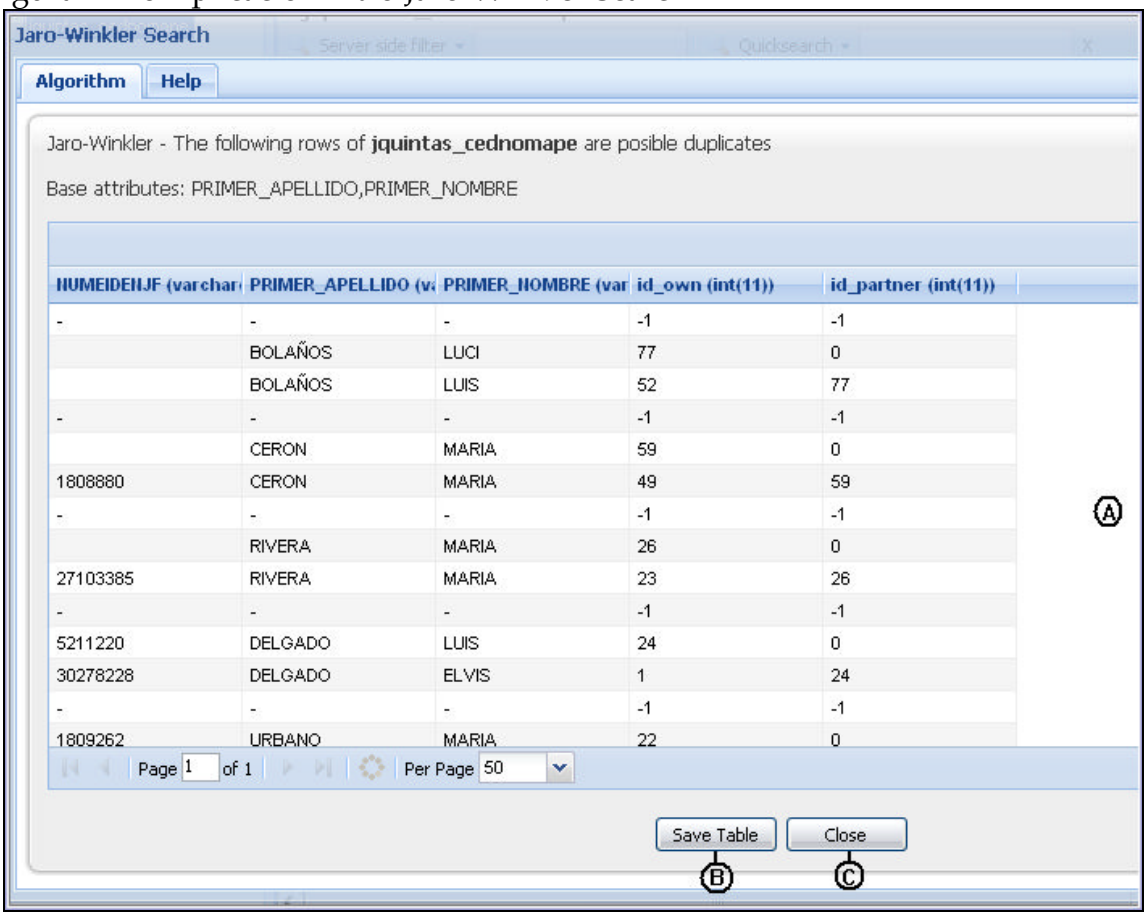
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Jaro-Winkler Search (A).	2. Se despliega la ventana Jaro-Winkler donde se establecen los parámetros del filtro.

Figura A127 Parámetros filtro Jaro-Winkler Search



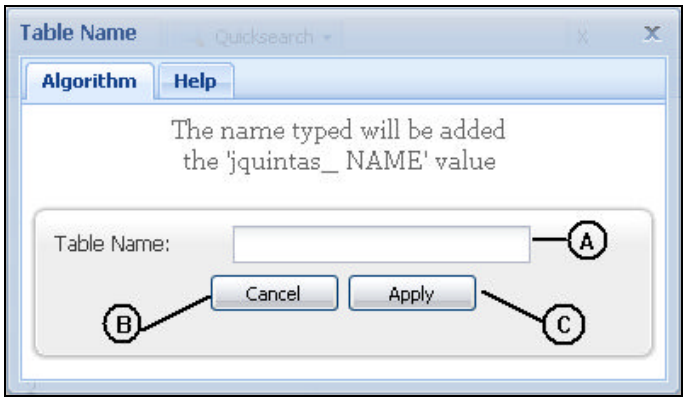
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el o los campos que serán objeto de la búsqueda de homónimos, haciendo clic en el campo Available Attributes(A) y transfiriendo los atributos al campo Selected Attributes (B)	
2. El usuario hace clic en el botón Cancel (C).	3. Se cancela la aplicación del filtro.
4. El usuario hace clic en el botón Apply (D).	5. Se genera la ventana emergente Jaro_winkler Search con el listado de duplicados encontrados.

Figura A128 Aplicación filtro Jaro-Winkler Search



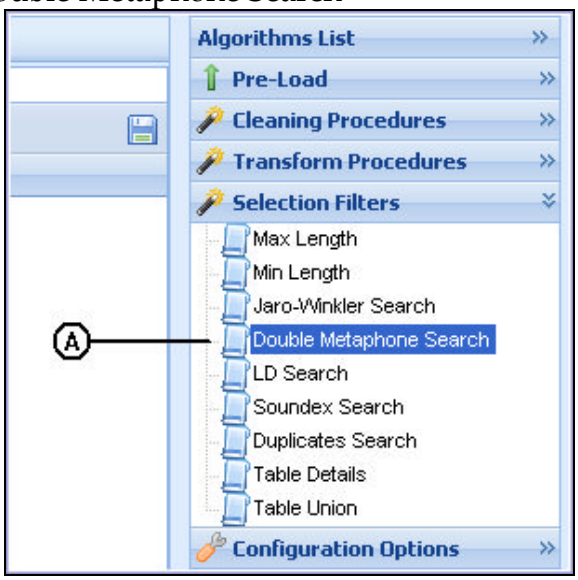
Acción del Usuario	Respuesta del Sistema
	1. Se muestra una ventana que contiene un listado de posibles coincidencias encontradas de acuerdo al algoritmo separadas por guiones. Esta ventana se compone de: (E) Listados de posibles duplicados. (F) Save Table, permite guardar la tabla como una propia del sistema. (G) Cierra la ventana de visualización.
2. El usuario hace clic en el botón Save Table (B).	3. Se pasa a la ventana Table Name, donde se especifica el nombre de la nueva tabla.
4. El usuario hace clic en el botón Close (C).	5. Se cierra la ventana emergente y se cancela la aplicación del filtro.

Figura A129 Table Name



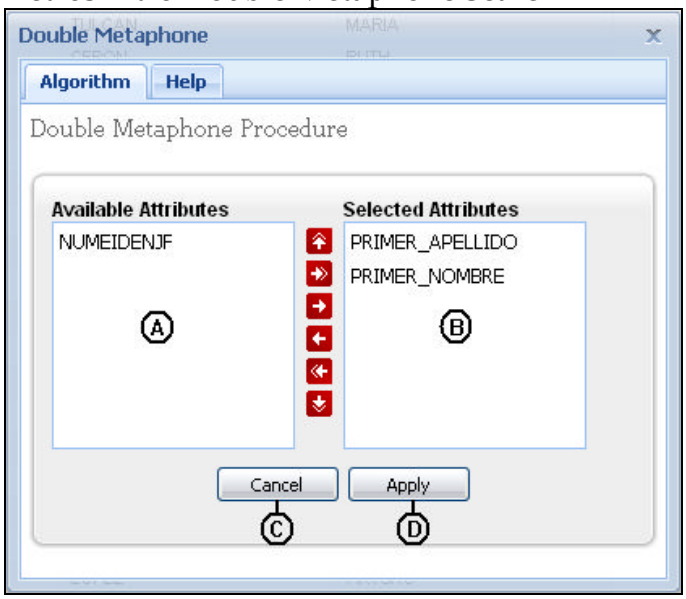
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A).	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error.
3. El usuario hace clic en el botón Apply (C).	4. Se crea una nueva tabla que contiene los resultados de la búsqueda de duplicados y se anexa al árbol de tablas.
5. El usuario hace clic en el botón Cancel (B)	6. Se cancela la acción guardar tabla.

Figura A130 Filtro Double Metaphone Search



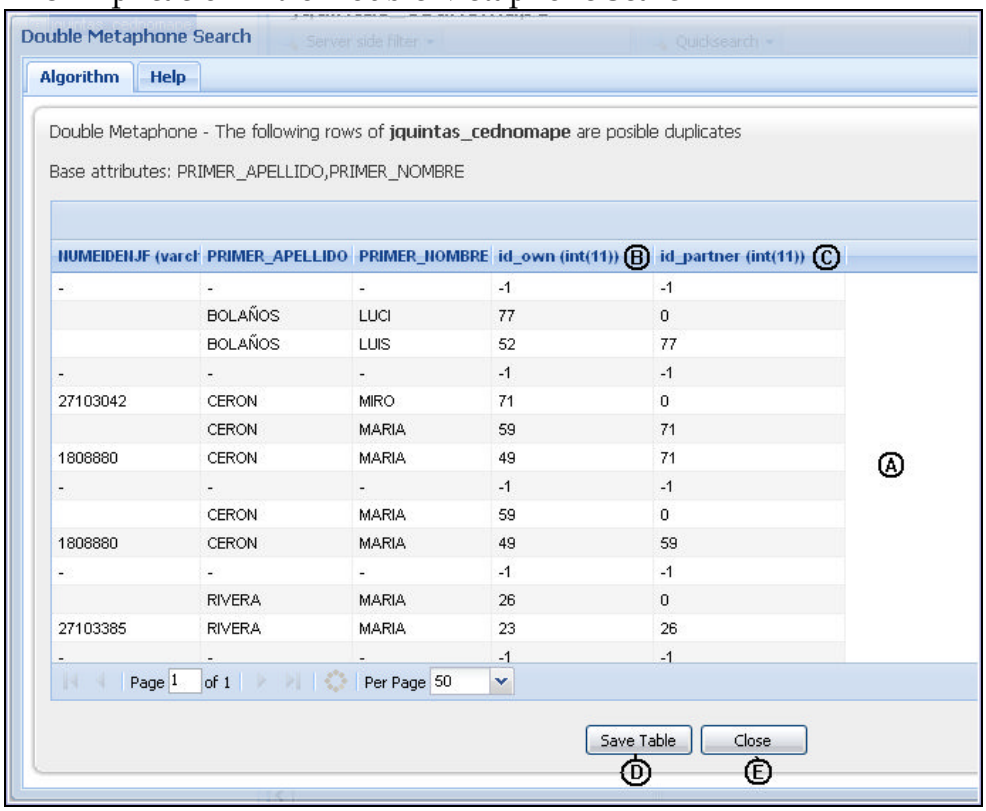
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Double Metaphone Search (A).	2. Se despliega la ventana Double Metaphone donde se establecen los parámetros del filtro.

Figura A131 Parámetros filtro Double Metaphone Search



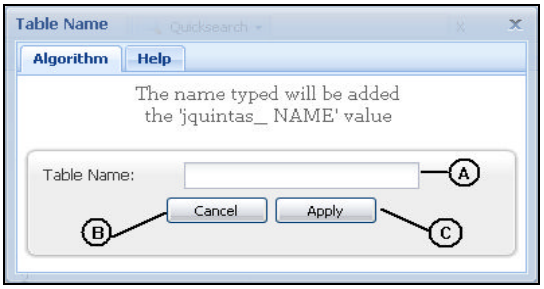
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el o los campos que serán objeto de la búsqueda de homónimos, haciendo clic en el campo Available Attributes(A) y transfiriendo los atributos al campo Selected Attributes (B)	
2. El usuario hace clic en el botón Cancel (C).	3. Se cancela la aplicación del filtro.
4. El usuario hace clic en el botón Apply (D).	5. Se genera la ventana emergente Double Metaphone Search con el listado de duplicados encontrados.

Figura A132 Aplicación filtro Double Metaphone Search



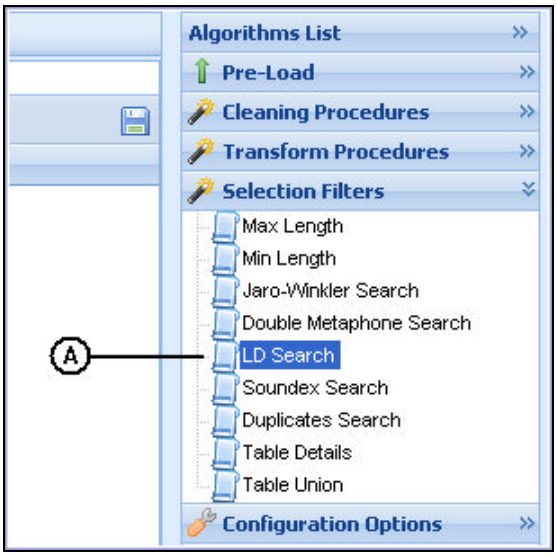
Acción del Usuario	Respuesta del Sistema
	1. Se muestra una ventana que contiene un listado de posibles coincidencias encontradas de acuerdo al algoritmo separadas por guiones. Esta ventana se compone de: (A) Listados de posibles duplicados. (B) Id_own, es un identificador asignado automáticamente para cada registro. (C) Id_partner, es el identificador del registro con el cual el registro buscado se relaciona. (D) Save Table, permite guardar la tabla como una propia del sistema. (E) Cierra la ventana de visualización.
2. El usuario hace clic en el botón Save Table (B).	3. Se pasa a la ventana Table Name, donde se especifica el nombre de la nueva tabla.
4. El usuario hace clic en el botón Close (C).	5. Se cierra la ventana emergente y se cancela la aplicación del filtro.

Figura A133 Table Name



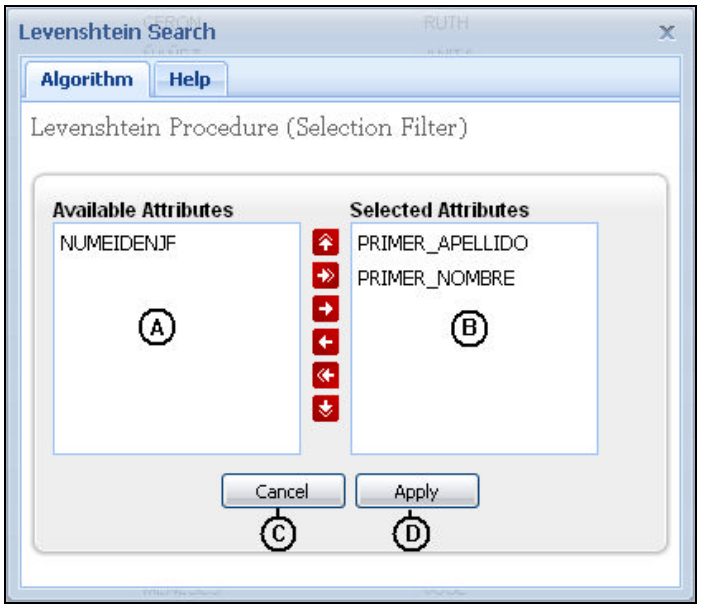
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A).	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error.
3. El usuario hace clic en el botón Apply (C).	4. Se crea una nueva tabla que contiene los resultados de la búsqueda de duplicados y se anexa al árbol de tablas.
5. El usuario hace clic en el botón Cancel (B)	6. Se cancela la acción guardar tabla.

Figura A134 Filtro LD Search



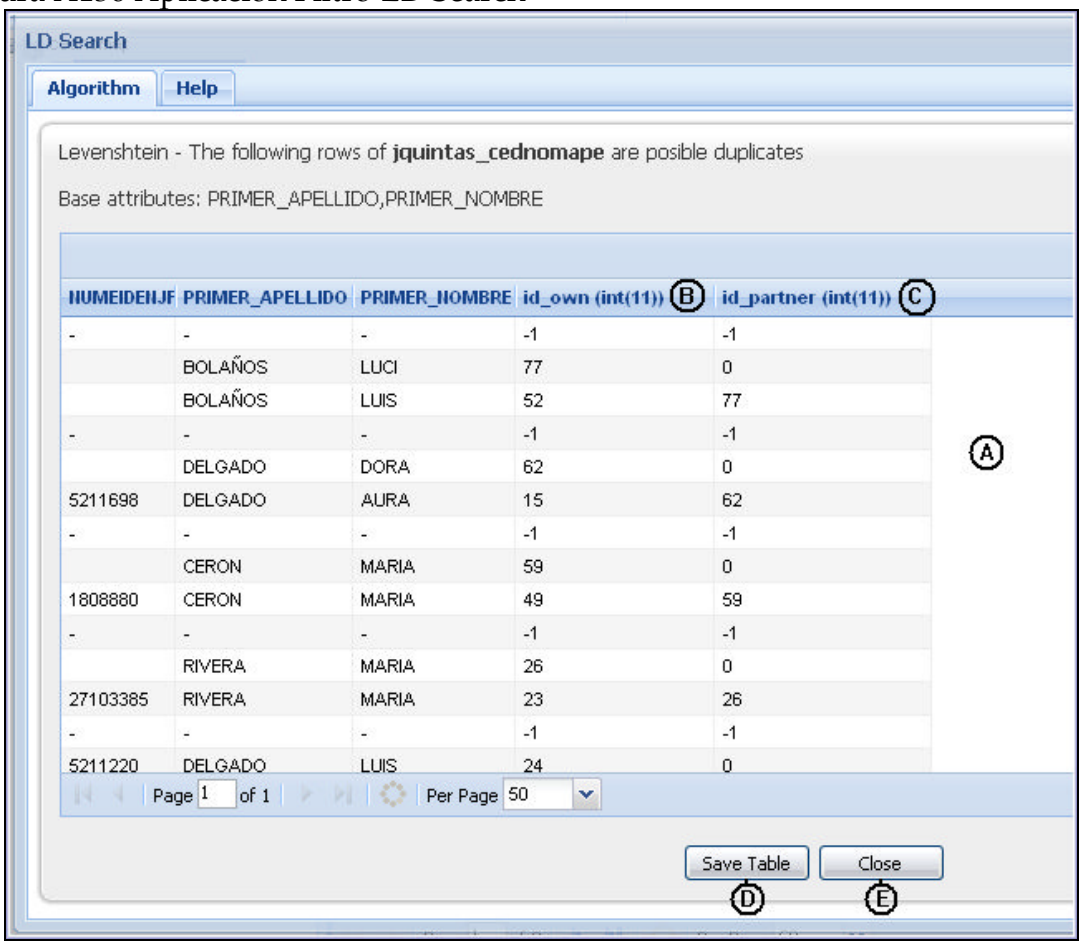
Acción del Usuario	Respuesta del Sistema
El usuario selecciona el filtro LD Search (A).	Se despliega la ventana Levenshtein Search donde se establecen los parámetros del filtro.

Figura A135 Parámetros filtro LD Search



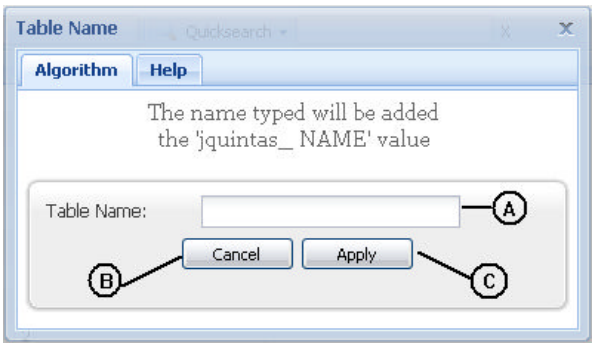
Acción del Usuario	Respuesta del Sistema
El usuario selecciona el o los campos que serán objeto de la búsqueda de homónimos, haciendo clic en el campo Available Attributes (A) y transfiriendo los atributos al campo Selected Attributes (B)	
2. El usuario hace clic en el botón Cancel (C).	3. Se cancela la aplicación del filtro.
4. El usuario hace clic en el botón Apply (D).	5. Se genera la ventana emergente LD Search con el listado de duplicados encontrados.

Figura A136 Aplicación Filtro LD Search



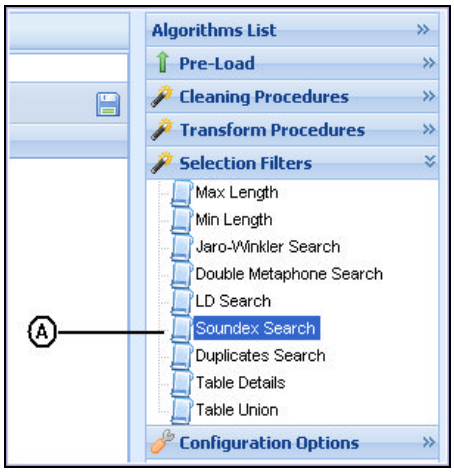
Acción del Usuario	Respuesta del Sistema
	1. Se muestra una ventana que contiene un listado de posibles coincidencias encontradas de acuerdo al algoritmo separadas por guiones. Esta ventana se compone de: (A) Listados de posibles duplicados. (B) Id_own, es un identificador asignado automáticamente para cada registro. (C) Id_partner, es el identificador del registro con el cual el registro buscado se relaciona. (D) Save Table, permite guardar la tabla como una propia del sistema. (E) Cierra la ventana de visualización.
2. El usuario hace clic en el botón Save Table (B).	3. Se pasa a la ventana Table Name, donde se especifica el nombre de la nueva tabla.
4. El usuario hace clic en el botón Close (C).	5. Se cierra la ventana emergente y se cancela la aplicación del filtro.

Figura A137 Table Name



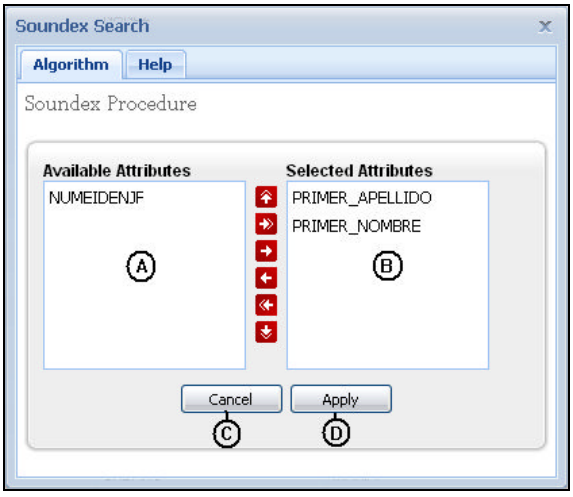
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A).	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error.
3. El usuario hace clic en el botón Apply (C).	4. Se crea una nueva tabla que contiene los resultados de la búsqueda de duplicados y se anexa al árbol de tablas.
5. El usuario hace clic en el botón Cancel (B)	6. Se cancela la acción guardar tabla.

Figura A138 Filtro Soundex Search



Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Soundex Search (A).	2. Se despliega la ventana Soundex Search donde se establecen los parámetros del filtro.

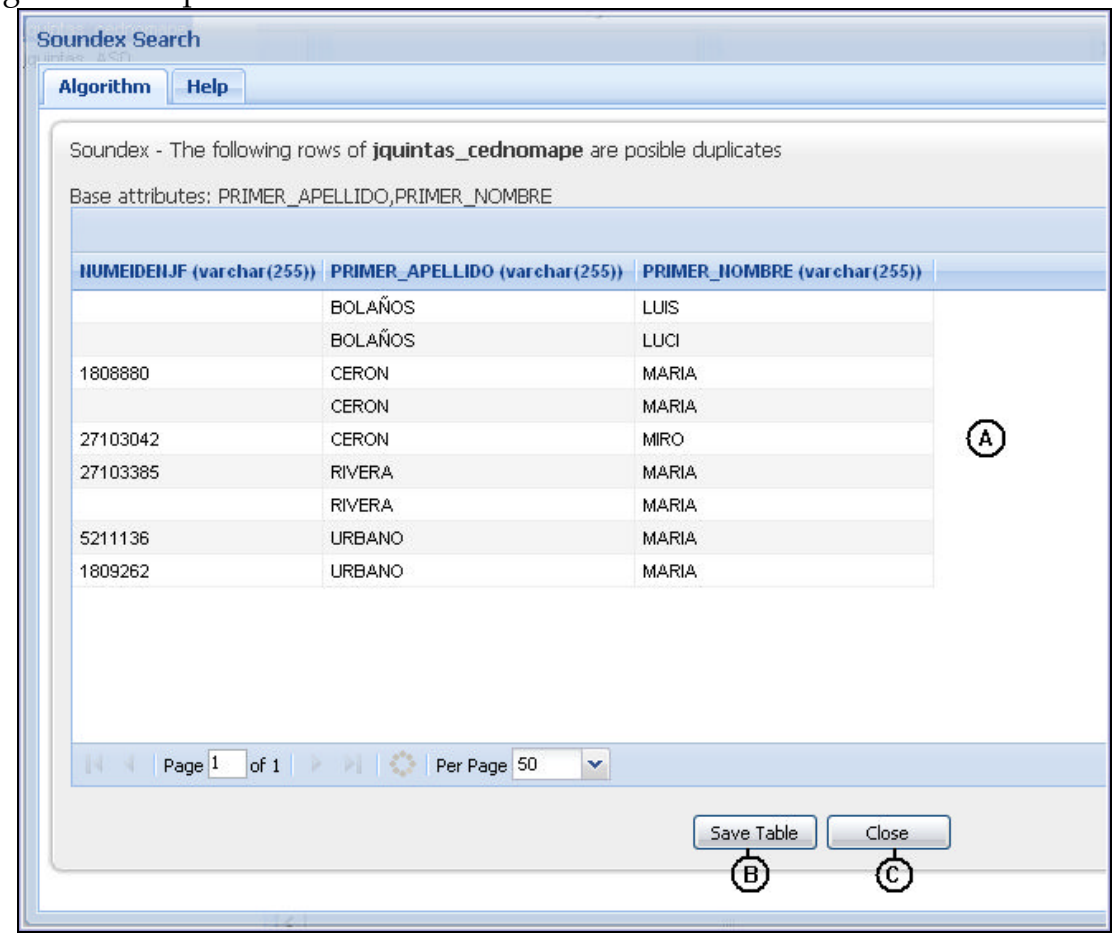
Figura A139 Parámetros Filtro Soundex Search



Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el o los campos que serán objeto de la búsqueda de homónimos, haciendo clic en el campo Available Attributes (A) y	

Acción del Usuario	Respuesta del Sistema
transfiriendo los atributos al campo Selected Attributes (B)	
2. El usuario hace clic en el botón Cancel (C).	3. Se cancela la aplicación del filtro.
4. El usuario hace clic en el botón Apply (D).	5. Se genera la ventana emergente Soundex Search con el listado de duplicados encontrados.

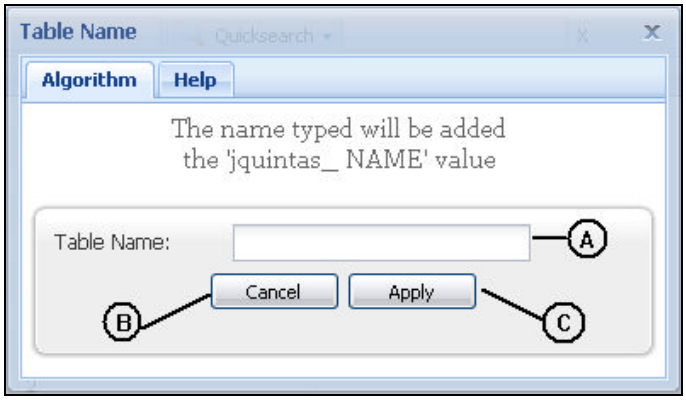
Figura A140 Aplicación Filtro Soundex Search



Acción del Usuario	Respuesta del Sistema
	1. Se muestra una ventana que contiene un listado de posibles coincidencias encontradas de acuerdo al algoritmo. Esta

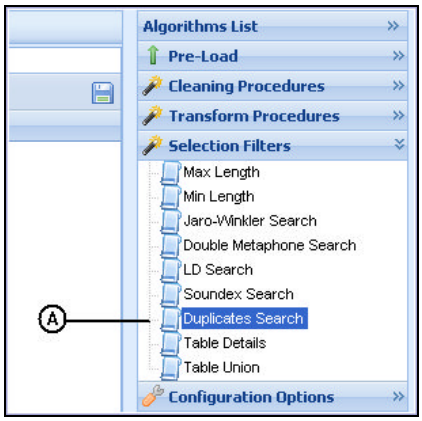
Acción del Usuario	Respuesta del Sistema
	ventana se compone de: (A) Listados de posibles duplicados. (B) Save Table, permite guardar la tabla como una propia del sistema. (C) Cierra la ventana de visualización.
2. El usuario hace clic en el botón Save Table (B).	3. Se pasa a la ventana Table Name, donde se especifica el nombre de la nueva tabla.
4. El usuario hace clic en el botón Close (C).	5. Se cierra la ventana emergente y se cancela la aplicación del filtro.

Figura A141 Table Name



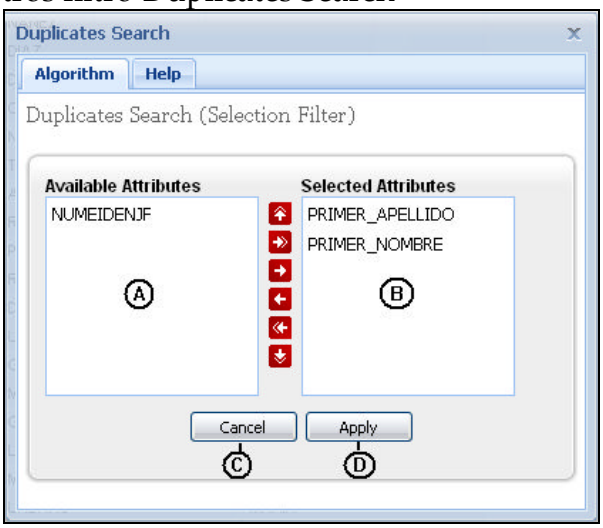
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón Cancel (B)	2. Si el nombre es incorrecto o no se está disponible,el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla que contiene los resultados de la búsqueda de duplicados y se anexa al árbol de tablas. 6. Se cancela la acción guardar tabla.

Figura A142 Filtro Duplicates Search



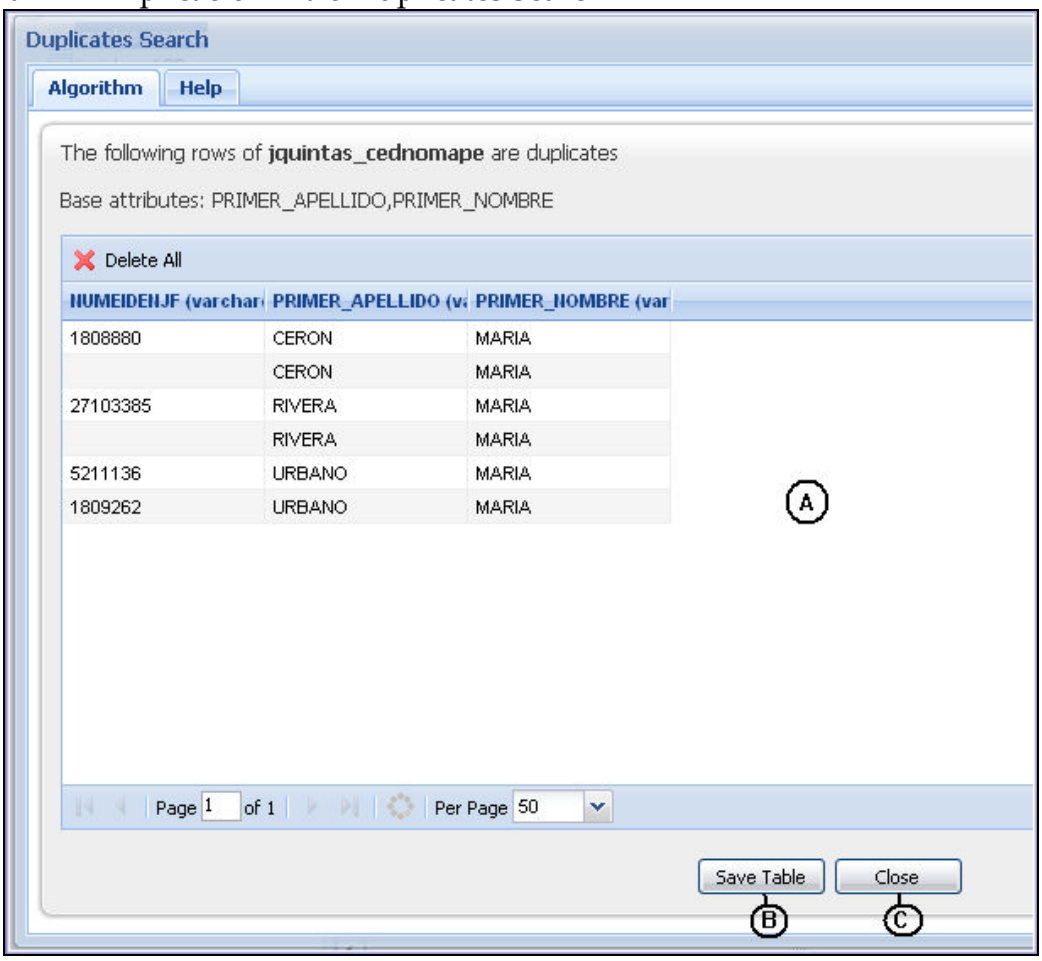
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Duplicates Search (A).	2. Se despliega la ventana Duplicates Search donde se establecen los parámetros del filtro.

Figura A143 Parámetros filtro Duplicates Search



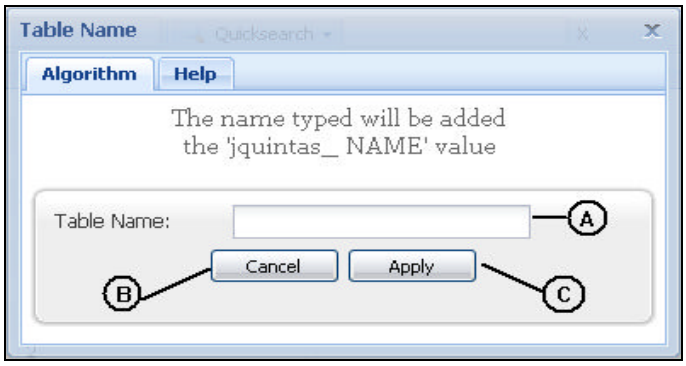
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el o los campos que serán objeto de la búsqueda de duplicado exactos, haciendo clic en el campo (A) y transfiriendo los atributos al campo Selected Attributes (B)	
2. El usuario hace clic en el botón Cancel (C).	3. Se cancela la aplicación del filtro.
4. El usuario hace clic en el botón Apply (D).	5. Se genera la ventana emergente Duplicates Search con el listado de duplicados encontrados.

Figura A144 Aplicación filtro Duplicates Search



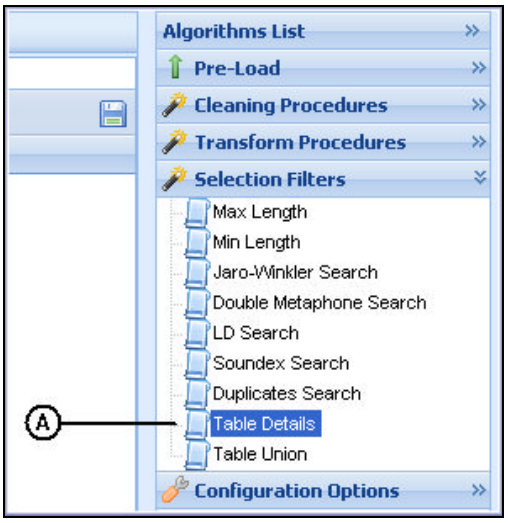
Acción del Usuario	Respuesta del Sistema
	1. Se muestra una ventana que contiene un listado de posibles coincidencias encontradas de acuerdo al algoritmo. Esta ventana se compone de: (A) Listados de posibles duplicados. (B) Save Table, permite guardar la tabla como una propia del sistema. (C) Cierra la ventana de visualización.
2. El usuario hace clic en el botón Save Table (B).	3. Se pasa a la ventana Table Name, donde se especifica el nombre de la nueva tabla.
4. El usuario hace clic en el botón Close (C).	5. Se cierra la ventana emergente y se cancela la aplicación del filtro.

Figura A145 Table Name



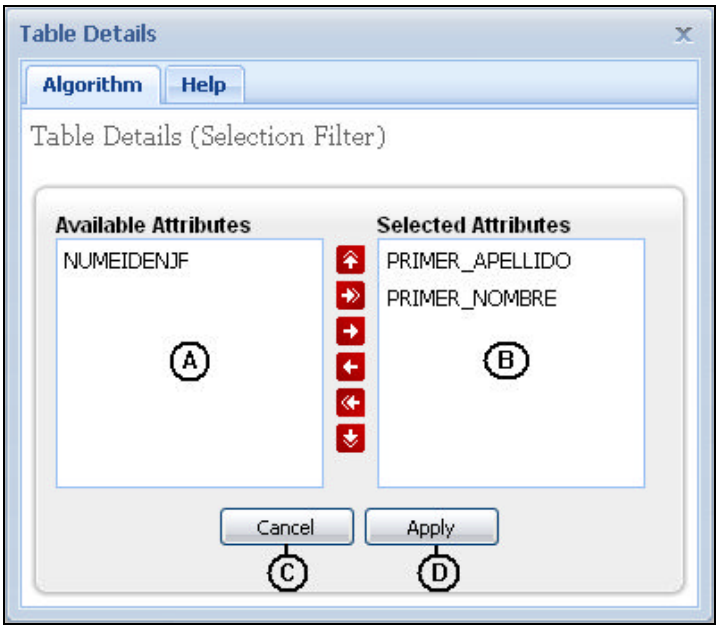
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A). 3. El usuario hace clic en el botón Apply (C). 5. El usuario hace clic en el botón Cancel (B)	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error. 4. Se crea una nueva tabla que contiene los resultados de la búsqueda de duplicados y se anexa al árbol de tablas. 6. Se cancela la acción guardar tabla.

Figura A146 Filtro Table Details



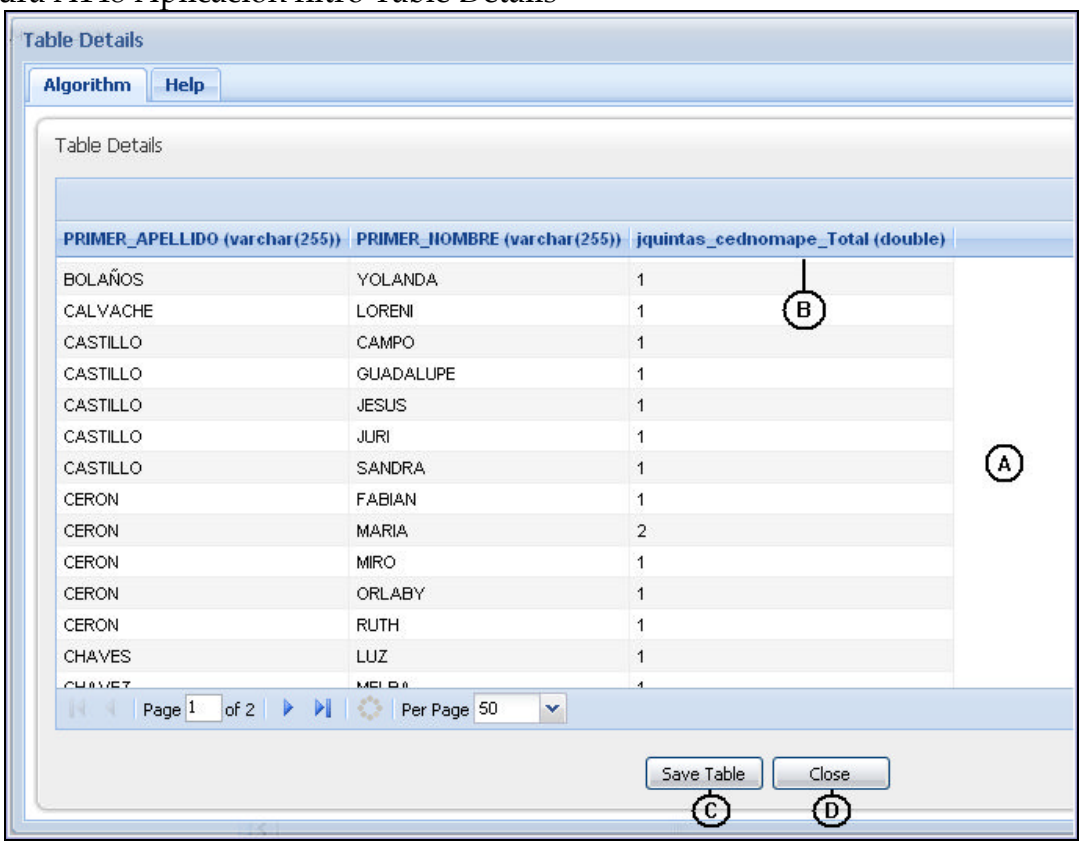
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Table Details (A).	2. Se despliega la ventana Table Details donde se establecen los parámetros del filtro.

Figura A147 Parámetros filtro Table Details



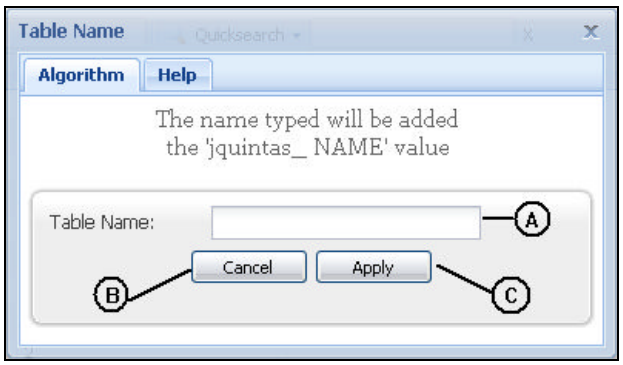
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el o los campos que serán detallados, haciendo clic en el campo Available Attributes (A) y transfiriendo los atributos al campo Selected Attributes (B)	
2. El usuario hace clic en el botón Cancel (C).	3. Se cancela la aplicación del filtro.
4. El usuario hace clic en el botón Apply (D).	5. Se genera la ventana emergente Table Details que contiene cada uno de los diferentes registros agrupados por los atributos seleccionados y la cantidad de registros que se encuentran.

Figura A148 Aplicación filtro Table Details



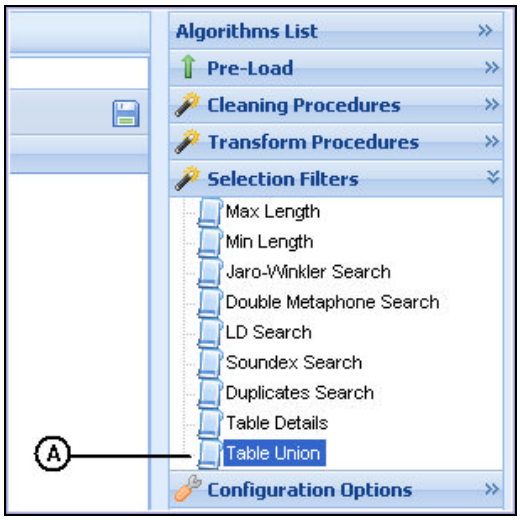
Acción del Usuario	Respuesta del Sistema
	1. Se muestra una ventana que contiene un listado de los diferentes registros y la cantidad de los mismos teniendo en cuenta los atributos seleccionados por el usuario. Esta ventana se compone de: (A) Listados de posibles duplicados. (B) Total, este campo contiene la cantidad de registros existentes. (C) Save Table, permite guardar la tabla como una propia del sistema. (D) Cierra la ventana de visualización.
2. El usuario hace clic en el botón Save Table (B).	3. Se pasa a la ventana Table Name, donde se especifica el nombre de la nueva tabla.
4. El usuario hace clic en el botón Close (C).	5. Se cierra la ventana emergente y se cancela la aplicación del filtro.

Figura A149 Table Name



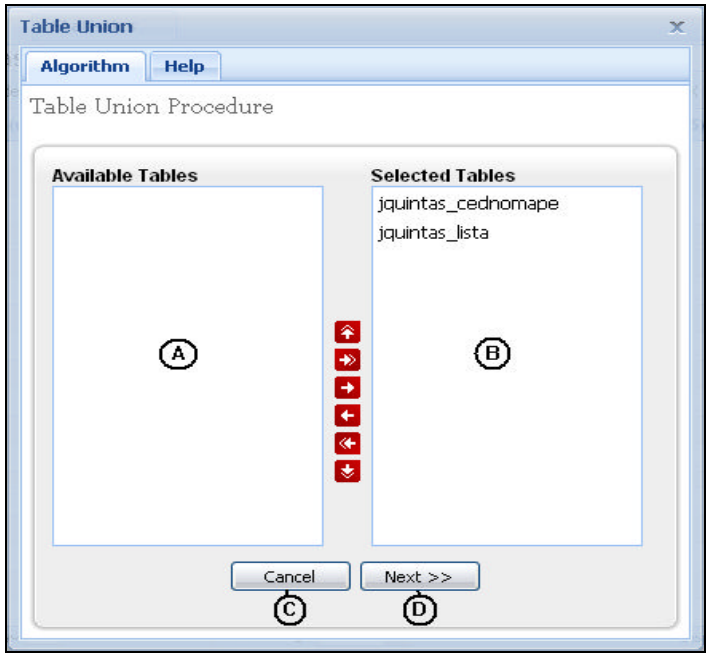
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A).	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error.
3. El usuario hace clic en el botón Apply (C).	4. Se crea una nueva tabla que contiene los detalles de la tabla seleccionada y se anexa al árbol de tablas.
5. El usuario hace clic en el botón Cancel (B)	6. Se cancela la acción guardar tabla.

Figura A150 Filtro Table Union



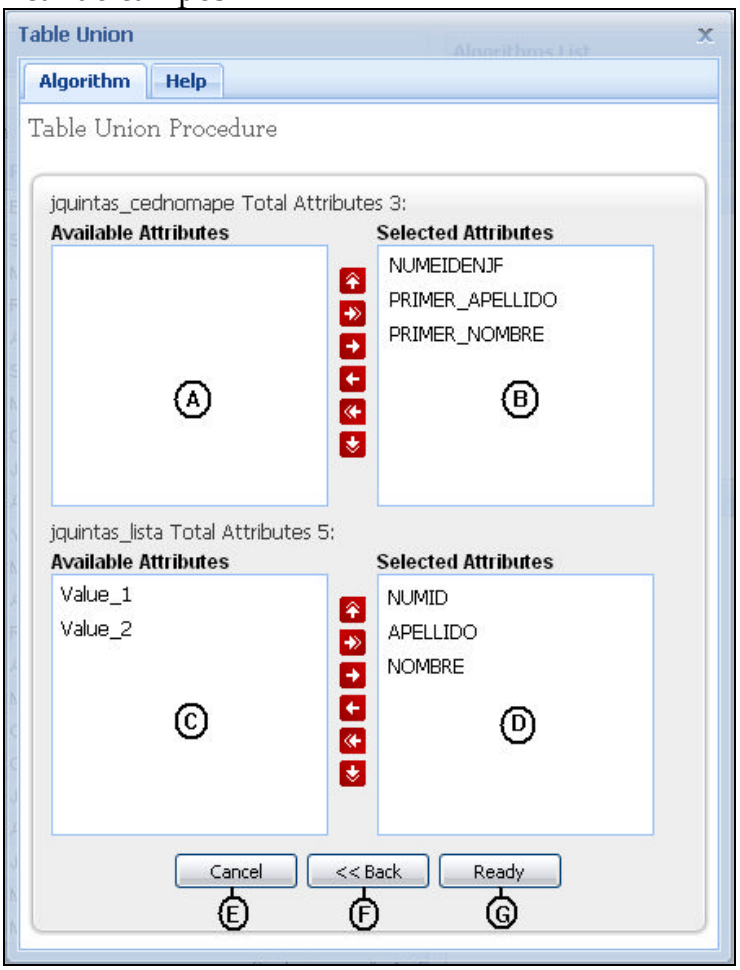
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona el filtro Table Union (A).	2. Se despliega la ventana Table Union donde se establecen los parámetros del filtro.

Figura A151 Selección de tablas a unir



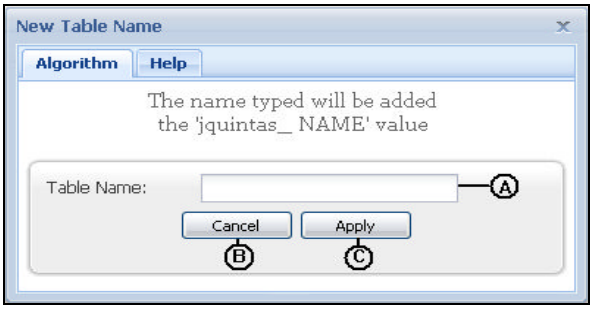
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona la tablas a unir, en este caso no pueden ser más de dos, haciendo clic en el campo Available Tables (A) y transfiriendo las tablas al campo Selected Tables (B)	
2. El usuario hace clic en el botón Cancel (C).	3. Se cancela la unión de tablas.
4. El usuario hace clic en el botón Next (D).	5. Se continúa con el paso siguiente de configuración de los parámetros de la unión de tablas.

Figura A152 Alineando campos



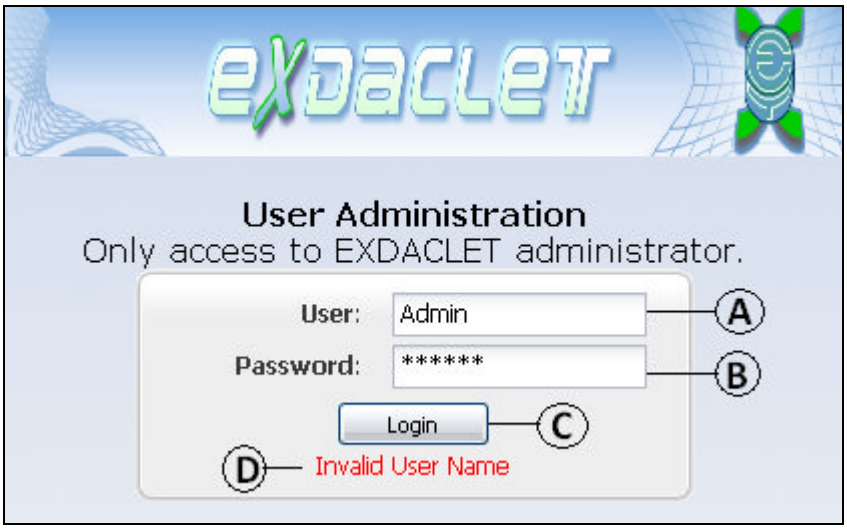
Acción del Usuario	Respuesta del Sistema
1. El usuario selecciona en orden los atributos de la primera tabla seleccionada haciendo clic en el campo Available Attributes(A) y transfiriendo las tablas al campo Selected Attributes (B)	
2. El usuario selecciona en orden los atributos de la segunda tabla seleccionada teniendo en cuenta el orden de los primeros atributos, haciendo clic en el campo Available Attributes(C) y transfiriendo las tablas al campo Selected Attributes (C)	
3. El usuario hace clic en el botón Cancel (E).	4. Se cancela la unión de tablas.
5. El usuario hace clic en el botón Back (F)	6. Se retorna a la ventana de configuración anterior.
7. El usuario hace clic en el botón Ready (G).	8. Se pasa a la ventana New Table Name.

Figura A153 New Table Name



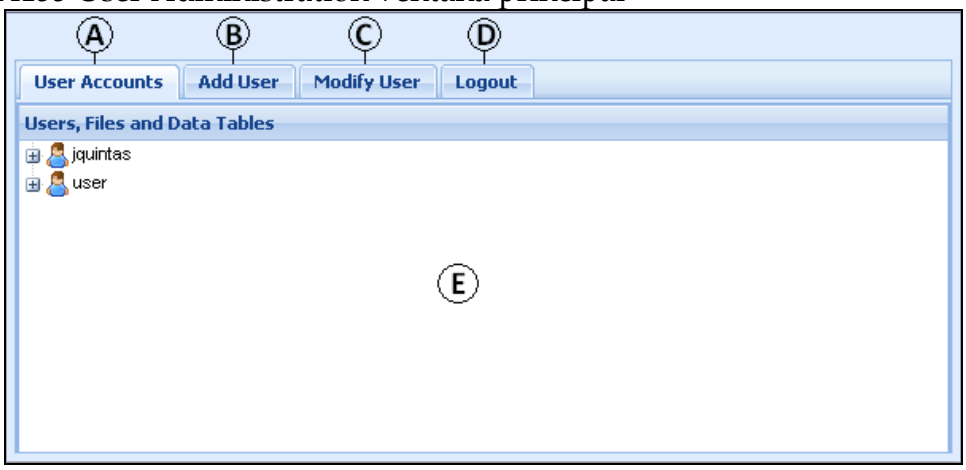
Acción del Usuario	Respuesta del Sistema
1. El usuario ingresa el nombre de la nueva tabla donde se aplicará el filtro (A).	2. Si el nombre no es correcto o no se encuentra disponible, el nombre se marca en rojo, como símbolo de error.
3. El usuario hace clic en el botón Apply (C).	4. Se crea una nueva tabla con los atributos en el orden especificado por el usuario y se anexa al árbol de tablas.
5. El usuario hace clic en el botón Cancel (B)	6. Se cancela la acción guardar tabla.

Figura A154 User Administration



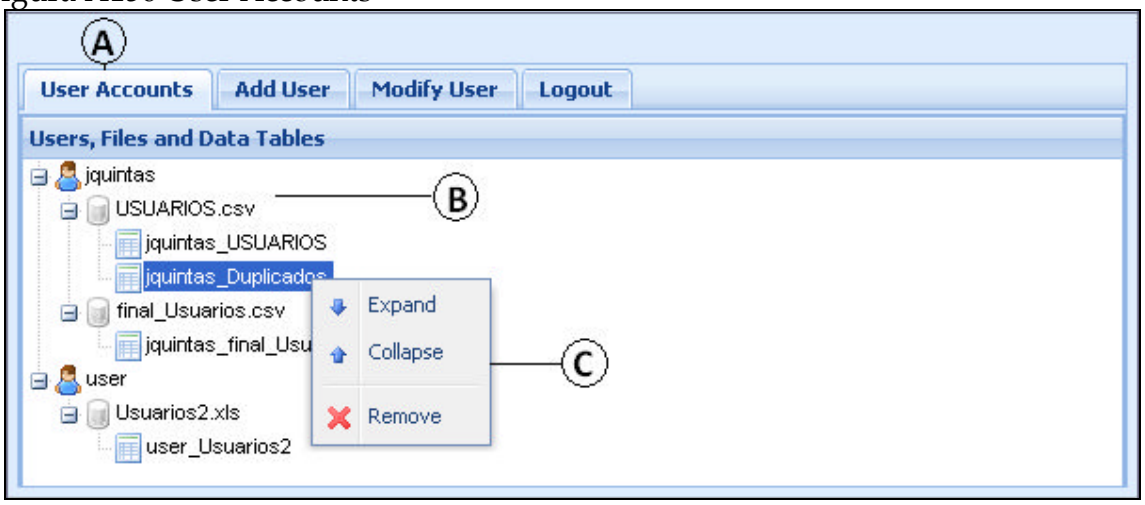
Acción del Usuario	Respuesta del Sistema
	1. Se muestra la ventana User Administration.
2. El usuario escribe su login en el campo User (A). 3. El usuario escribe su contraseña en el campo Password (B)	
4. El usuario hace clic en el botón Login (C).	5. Si la verificación de los datos Login y Password es correcta entonces se muestra la ventana de administración de EXDACLET, de lo contrario se muestra un mensaje de error en el campo (D).

Figura A155 User Administration ventana principal



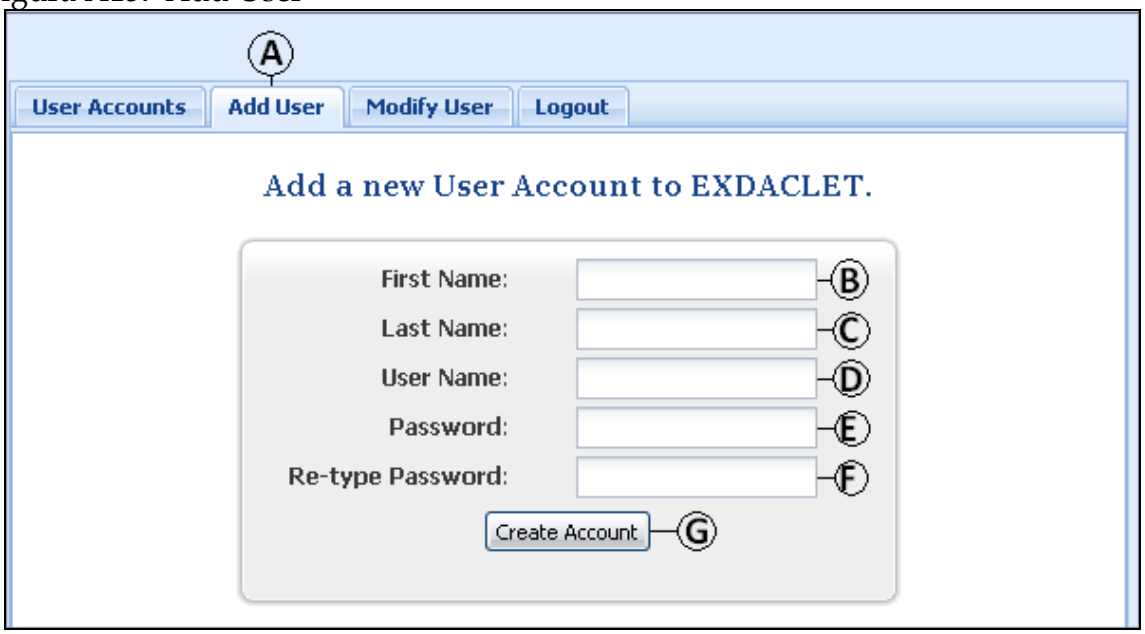
Acción del Usuario	Respuesta del Sistema
	1. Se despliega la ventana de administración de usuarios que contiene: (A): Users Accounts, muestra los usuarios que hacen parte de EXDACLET y los archivos tratados por cada usuario. (B) : Add User, despliega una ventana donde se puede agregar un nuevo usuario. (C) : Modify User, despliega una ventana con los datos de un usuario seleccionado en la cual se pueden modificar sus datos. (D): Logout, permite la salida segura del modulo de administración (E) : Area de trabajo.

Figura A156 User Accounts



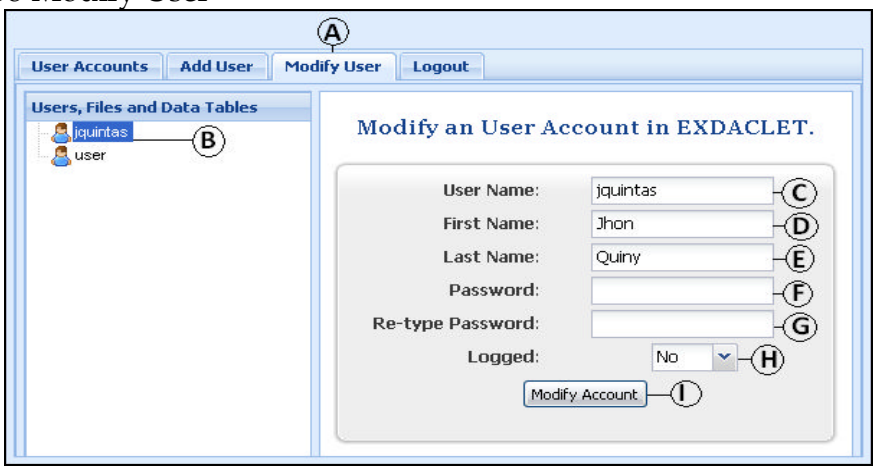
Acción del Usuario	Respuesta del Sistema
1. El administrador hace clic en la pestaña User Accounts (A). 3. El administrador hace clic derecho sobre las hojas del árbol desplegado.	2. Se despliega en el área de trabajo el árbol de usuarios y una lista de los archivos tratados por cada usuario. 4. Se despliega un menú contextual (C) de donde se encuentra la opción Remove, que permite eliminar una tabla o un archivo con sus tablas o un usuario con todos sus archivos y tablas.

Figura A157 Add User



Acción del Usuario	Respuesta del Sistema
1. El administrador hace clic en la pestaña Add User (A).	2. Se despliega en el área de trabajo un formulario solicitando los datos básicos del nuevo usuario.
3. El administrador ingresa los datos solicitados en el formulario: (B) First Name, primer nombre; (C) Last Name, Apellido; (D) User Name, Login o nombre de usuario; (E) Password, Contraseña; (F) Re-type Password, se reescribe la contraseña por seguridad.	
4. El usuario hace clic en el botón Create Account (G).	5. Si los datos requeridos han sido ingresados correctamente se agrega la nueva cuenta a la lista de usuarios.

Figura A158 Modify User



Acción del Usuario	Respuesta del Sistema
1. El administrador hace clic en la pestaña Modify User (A).	2. Se despliega en el área de trabajo un árbol con la lista de los usuarios que hacen parte de EXDACLET.
3. El administrador selecciona un usuario (B) con doble clic.	4. Se despliega un formulario que tiene cargados los datos actuales del usuario a modificar.
5. El administrador modifica los datos solicitados en el formulario: (C) First Name, primer nombre; (D) Last Name, Apellido; (E) User Name, Login o nombre de usuario; (F) Password, Contraseña; (G) Re-type Password, se reescribe la contraseña; (H) Logged, si el usuario se encuentra haciendo uso del sistema actualmente.	
6. El usuario hace clic en el botón Modify Account (G).	7. Si los datos requeridos han sido actualizados correctamente se modifica cuenta del usuario.

A2. MANUAL DE INSTALACION

EXDACLET es una herramienta libre orientada a la Web, por lo tanto, necesita de ciertos componentes para su funcionamiento.

- Servidor Web HTTP Apache.
- Bibliotecas de compilación de PHP.
- Sistema gestor de Bases de Datos MYSQL.

Tanto el servidor apache como MYSQL y las bibliotecas de PHP se consiguen fácilmente en grupos denominados triadas AMP (Apache, MySQL, PHP) que tras un proceso de instalación sencillo, deja estos tres componentes configurados; entre las triadas más conocidas se encuentran WAMP, XAMP y AppServ.

Las aplicaciones Web son un concepto novedoso, por esta razón es recomendado que se tenga la última versión compilada de cada componente instalado.

Una vez instalados los componentes anteriormente mencionados es necesario modificar los archivos de configuración **httpd.conf, php.ini y my.ini**.

httpd.conf. En este archivo se deben escribir las direcciones ip que tendrán acceso completo al servidor para poder inicializar la aplicación. Por ejemplo si se quiere dar acceso a las direcciones ip 192.168.20.65 y 192.168.20.13, entonces se debe buscar la sección de acceso de hosts y escribir las siguientes líneas.

```
Deny from All
Allow from 192.168.20.65
Allow from 192.168.20.13
```

Por lo general, la sección se acceso a hosts se encuentra dentro de las siguientes líneas:

<Directory "***ruta***/www/">
Options Indexes followSymLinks
AllowOverride all
Order Deny, Allow
Deny from All
…….
<\Directory>

Además, se debe verificar que exista la línea ***AddDefaultCharset utf-8***, si no existe, se debe agregar al final del archivo.

php.ini. En este archivo, se deben modificar las siguientes secciones:

- Resource Limits: Se deben cambiar los valores de las variables ***max_execution_time*** para aumentar el tiempo en segundos, de ejecución de un script, es necesaria para que la herramienta pueda trabajar con conjuntos de datos grandes y sobre los cuales se apliquen filtros que tomen una buena cantidad de tiempo para generar resultados. Se sugiere establecer esta variable con un valor máximo de 432000. **m*ax_input_time*** para definir cuanto tiempo en segundos requiere un script para extraer los datos. Se sugiere establecer un valor de 14400. **m*emory_limit*** referente a la cantidad de memoria que un script puede requerir para su ejecución. Este valor debe ser definido de acuerdo a las características del servidor, para un equipo con 1 GB de memoria se sugiere establecerlo en 256M.
- Data Handling: Se debe cambiar la variable ***post_max_size*** a 150M, que es el tamaño máximo de archivos que pueden ser cargados en EXDACLET.
- File Uploads: la variable ***file_uploads*** debe estar en "on" y la variable ***upload_max_filesize*** debe contener el mismo valor especificado para la variable ***post_max_size*** es decir, 150M.
- Dynamic Extensions: se debe garantizar que no esté en comentarios la línea **extensión=php_gd2.dll** que es utilizada por EXDACLET para generar los gráficos estadísticos

Para LINUX se deben garantizar los permisos de escritura sobre la carpeta temporal en la cual php pone los archivos que se van a cargar, de otro modo, en la sección File Uploads, se debe especificar la ruta en la cual serán cargados los archivos definiéndola en la variable ***upload_tmp_dir = "ruta_carpeta_temporal"***.

Esa ruta_carpeta_temporal debe tener todos los permisos de esritura y lectura. Además se debe garantizar que esté instalado y en funcionamiento el módulo gráfico GD.

my.ini. En este archivo se debe garantizar que las variables ***default-character-set*** y ***default-storage-engine*** tengan los valores ***latin1*** y ***INNODB*** RESPECTIVAMENTE.

Una vez modificados los anteriores archivos de configuración, se deben reiniciar el servidor apache y MySQL para que los cambios surtan efecto, posteriormente descomprimir el archivo EXDACLET.RAR para Windows o EXDACLET.TAR.GZ para Linux, en la carpeta de publicación de sitios de la triada instalada.

Para LINUX se deben garantizar los permisos de escritura para la carpeta EXDACLET, de esta manera:

[root]# chmod 0777 /var/www/html/exdaclet –R –v -c

Los siguientes son los procedimientos que se deben seguir para poner en funcionamiento EXDACLET:

- Iniciar una consola de MySql.
- Iniciar sesión con la contraseña respectiva.
- En la línea de comandos escribir las siguientes sentencias:
  mysql> create database exdaclet;
  mysql> use exdaclet
  mysql> source /***ruta***/exdaclet.sql

Donde ***ruta*** es la dirección completa en donde se encuetra el archivo exdaclet.sql.

A partir de este momento es posible utilizar EXDACLET abriendo el navegador Web **Mozilla Firefox** y escribiendo en la dirección la siguiente url: *http://localhost/exdaclet.*

Para usuarios que se conecten a la herramienta, en la dirección, deben escribir la url: *http://direccion_ip_servidor/exdaclet*

Para el buen funcionamiento de la herramienta se sugiere garantizar que no existan otros servicios que utilicen el puerto de comunicaciones 80.