Universidad Central “Marta Abreu” de las Villas.
Facultad Matemática Física y Computación
Licenciatura en Ciencias de la Computación


# TRABAJO DE DIPLOMA

Titulo:

## DETERMINACIÓN DE UNA TAXONOMÍA DE ERRORES EN LOS SISTEMAS OPERACIONALES DE NUESTRO ENTORNO

Autor: Daynel Marmol Lacal
Tutor: MSc. Beatríz López Porrero

Santa Clara
2005


Hago constar que el presente trabajo fue realizado en la Universidad Central "Marta Abreu" de las Villas como parte de la culminación de los estudios de la especialidad de Ciencia de la Computación, autorizando a que el mismo sea utilizado por la institución, para los fines que estime conveniente, tanto de forma parcial como total y que además no podrá ser presentado en eventos, ni publicados sin autorización de la Universidad.

_____
Firma del Autor

Los abajo firmantes, certificamos que el presente trabajo ha sido realizado según acuerdos de la dirección de nuestro centro y el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

_____
Firma del Tutor

_____
Firma del Jefe de Seminario
donde se defiende el trabajo

_____
Firma del Responsable de
Información Científico- Técnica

“Quién nunca descansa,
quien en su corazón y la sangre piensa en lograr lo imposible:
Triunfa”

Ché

Agradezco a todos los profesores que a través de mi trayectoria estudiantil confiaron en mí y estimularon mis inquietudes inclinándome a mi verdadera vocación para que así culmine hoy con éxitos mis estudios.

Agradezco especialmente a la Msc. Beatriz López Porrero mi tutora y al Dr. Ramiro Pérez, quienes me han brindado su ayuda en todo momento.

A mis amigos, aquellos que me apoyaron incondicionalmente.

Agradezco también de forma especial, y no por ser de último menos importante, a mi hermana Damaris por todo su apoyo durante estos años.

A mi padre que no pudo estar conmigo, por haber sido su gran anhelo que llegara este día de mi graduación.

A mi madre por todo el apoyo y desvelo durante toda mi vida.

A mi hermano Darién

A mi mismo

# Indice

## Resumen

La limpieza de datos generalmente conocida como el proceso mediante el cual se eliminan errores e inconsistencia de los datos para mejorar su calidad, es muy importante para cualquier sistema que realice manejo de datos, y lo es más aún para aquellos sistemas que trabajan con grandes volumenes de datos y convierten estos en información para sobre ellos tomar decisiones.

Para realizarla se han desarrollado varios métodos que tienen en común, ser de dominio específico, o sea, herramientas para limpiar direcciones de EE.UU, limpiar listas de nombres, limpiar fechas, entre otras.

Muchas de estas herramientas son tan específicas que no serían efectivas al aplicarlas en nuestro país, por eso nuestro trabajo ha estado encaminado a la determinación de tipos de errores que aparecen en Sistemas de Bases de datos de nuestro entorno, para posteriormente desarrollar herramientas capaces de limpiarlos.

## Abstract

Data Cleansing generally known as the process by means of which bugs and inconsistencies of the information are eliminated to improve his quality, it is very important for any system that it realizes handling of information, and it it is even more for those systems that work with big volumes of data and convert these into information for on them to take decisions.

To realize it there have developed several methods that it has in common, be of specific domain, for example, softwares to clean addresses of USA, to clean lists of names, to clean dates, between others.

Many of this software are so specific that they would not be effective of being applied in our country, that's why our work has been directed to the determination of the types of errors that appear in Systems of Databases of our country, later to develop software capable of cleaning them.

# Introducción

Hoy en día gracias a la rápida y cada vez más creciente automatización de los diferentes sistemas de control en empresas e instituciones, tanto a nivel mundial como en nuestro país, y a la gran utilización de Bases de Datos para manipular toda la información contenida en dichos sistemas, se hace necesario la utilización de diferentes herramientas que de forma automatizada nos ayude a “limpiar” los datos que allí se manejan.

En general los pasos que desarrollan las diferentes herramientas para un adecuado proceso de limpieza son:

- Definir y determinar los tipos de error.
- Buscar e identificar las instancias erróneas.
- Corregir los errores.
- Documentar las instancias erróneas y los tipos de errores.
- Modificar los procedimientos de entrada de datos en aras de reducir errores futuros.

Debido a que la mayoría de las herramientas actuales son inaplicables en nuestros sistemas independientemente de su eficiencia, producto de su característica fundamental de estar preparadas para hacer limpiezas en dominios específicos y para analizar datos con un formato predefinido y en idioma inglés, nuestro proyecto está dirigido hacia la determinación de los errores más frecuentes en los Sistemas Informativos de nuestro entorno para poder posteriormente desarrollar herramientas de limpiezas adecuadas.

Este proyecto tiene como objetivo general:

- Construir una taxonomía de errores frecuentes en los sistemas operacionales.

Y como objetivos específicos:

- Resumir taxonomías de errores en sistemas informativos.
- Estudiar sistemas operacionales en nuestro contexto para determinar la aparición de errores a limpiar.
- Realizar análisis estadísticos para comprobar la aparición de dichos errores.

Este trabajo se compone de tres capítulos:

**Capítulo 1**: Estado del Arte del problema de la Limpieza de Datos.

**Capítulo 2**: Análisis y diseño de la Herramienta “DB Analyzer”

**Capítulo 3**: Resultados obtenidos con la Herramienta “DB Analyzer” y propuesta de taxonomía.

# Capítulo 1. Estado del arte del problema de la Limpieza de datos

En muchas aplicaciones relacionadas con los temas de investigación de Descubrimiento de conocimiento en los datos, Almacenes de datos, y Toma de decisiones, un aspecto crítico lo constituye el nivel de corrección de los datos con que se trabaja. Este tipo de aplicaciones generalmente se nutren de bases de datos operacionales, y con frecuencia en las mismas se encuentran registros de datos con información incompleta o errónea, pues aunque se plantea que existen varios factores que pueden influir en la calidad, consistencia e integridad de los datos; muchas veces, el origen de los datos, constituye un factor crucial. Aún cuando los desarrolladores de sistemas hagan ingentes esfuerzos por evitar los errores en los datos, la razón de error[1] es aproximadamente de un 5% [1].

La existencia de "datos sucios", como también se le llama a estos errores en los datos, tiene un gran impacto en las instituciones, reflejándose esto en un alto costo operacional, toma de decisiones inadecuadas, incremento de la inseguridad y una desviación de la atención de las direcciones de las instituciones [2].

La solución lógica a este problema es tratar de "limpiar" los datos de alguna forma; o sea, explorarlos para encontrar posibles errores y tratar de corregirlos. La realización de este proceso de forma manual es casi imposible por el número de horas-hombre requeridas para el mismo, además de ser por sí mismo un proceso muy laborioso, lento y susceptible de introducir nuevos errores en los datos; de ahí que la automatización de la limpieza de datos sea considerada una nueva e importante área de trabajo científico.

## 1.1 Definición de Limpieza de datos:

El proceso de limpieza de datos aborda diferentes elementos, por ejemplo, el tratamiento de valores ausentes o faltantes, la determinación de la utilidad de los registros, la determinación de datos erróneos, entre otros.

No existe una definición general establecida sobre este proceso, pues depende del área específica en que se aplique. Las principales áreas que incluyen el proceso de limpieza de datos como parte de su propio proceso son: los Almacenes de Datos (DW),

[1] Se define la razón de error como el número de errores por campos sobre el número total de campos

el Manejo de calidad total de los datos (TDQM), y Descubrimiento de conocimiento en bases de datos (KDD)

En los Almacenes de datos el proceso de limpieza de datos se aplica especialmente cuando varias bases de datos son mezcladas; artículos que se refieren a la misma entidad, se presentan en formatos diferentes en diferentes conjuntos de datos, o son representados erróneamente. De esta forma pueden aparecer **registros duplicados** en las bases de datos mezcladas. La tarea entonces es, identificar y eliminar estos duplicados. A este problema se le conoce como problema de Mezcla/Limpieza (**Merge/Purge**) y en la literatura sobre el tema es abordado además usando otros términos como Enlace de registros (**Record Linkage**), Integración semántica, Identificación de instancias, o Problema de identidad del objeto.

Desde este punto de vista, la limpieza de datos se define de varias formas aunque muy similares.

Algunas de estas definiciones son:

“Limpieza de datos es el proceso de eliminación de errores e inconsistencias en los datos, y aclaración del problema de identidad del objeto” [3].

“Limpieza de datos se define como el problema de Mezcla/limpieza” [4].

Sin embargo es importante notar que la limpieza de datos es algo más que una actualización de un registro de información con datos correctos. Con frecuencia este proceso involucra fases de descomposición y reensamble de los datos, lo que se tiene en cuenta en la definición dada por Kimball que establece que el proceso de limpieza de datos consta de seis pasos: Elementarización, Standarización, Verificación, Flujo inverso de datos limpios y Documentación.

En el área de TDQM, la limpieza de datos está relacionada con la aplicación de la calidad a los ciclos de adquisición y uso de los datos, y está compuesta de una serie de actividades: valoración, ajuste del análisis y decantación de los datos [5]. En este mismo marco, la calidad de los datos se mide en función de 4 dimensiones: “corrección, actualización, integridad y consistencia”.

Una definición del proceso de limpieza de datos en TDQM es:

“La limpieza de datos es el proceso que determina la corrección de los datos y mejora su calidad”

En el área de KDD, la limpieza de datos es el primer paso del preprocesamiento [6]. Tampoco existe aquí una definición precisa sobre la misma, y muchos sistemas de KDD y minería de datos resuelven las tareas de limpieza de datos, dirigiéndolas a un dominio específico, por ejemplo, son tareas de limpieza las siguientes:

- el descubrimiento de patrones de información poco significativos
- el problema de clasificación, empleando técnicas de *machine learning*.

Sin embargo, es usual encontrar como definición de limpieza de datos "el proceso que implementa métodos computarizados para examinar las bases de datos, detectar datos incorrectos y faltantes, y corregir los errores".

El campo de trabajo de la limpieza de datos en general es:

- definir y determinar los tipos de errores
- buscar y determinar las instancias con error
- corregir los errores
- documentar las instancias con errores y los tipos de errores
- modificar los procedimientos de entrada de datos para reducir errores futuros.

## 1.2 Principios de la Limpieza de datos.

La necesidad de la limpieza de datos se centra fundamentalmente alrededor de mejorar la calidad de los datos para hacerlos "apropiados para su uso" por los usuarios mediante la reducción de los errores en los datos y mejorando su documentación y presentación. La prevención de errores es mucho mejor que la detección de los mismos y su posterior limpieza, que es menos costoso y más eficiente prevenir la ocurrencia de errores que tratar de encontrarlos para después corregirlos. No importa cuan eficiente sea el proceso de entrada de datos, los errores siempre ocurrirán, por lo tanto ni la validación de datos ni la corrección pueden ser ignorados. La detección de errores, validación y limpieza son procesos que tienen un gran peso. Un importante producto de la limpieza de datos es la identificación de las principales causas de los errores detectados, y usando esa información, mejorar el proceso de entrada de datos para evitar que los mismos errores se repitan.

### 1.2.1 Principios de la limpieza de datos:

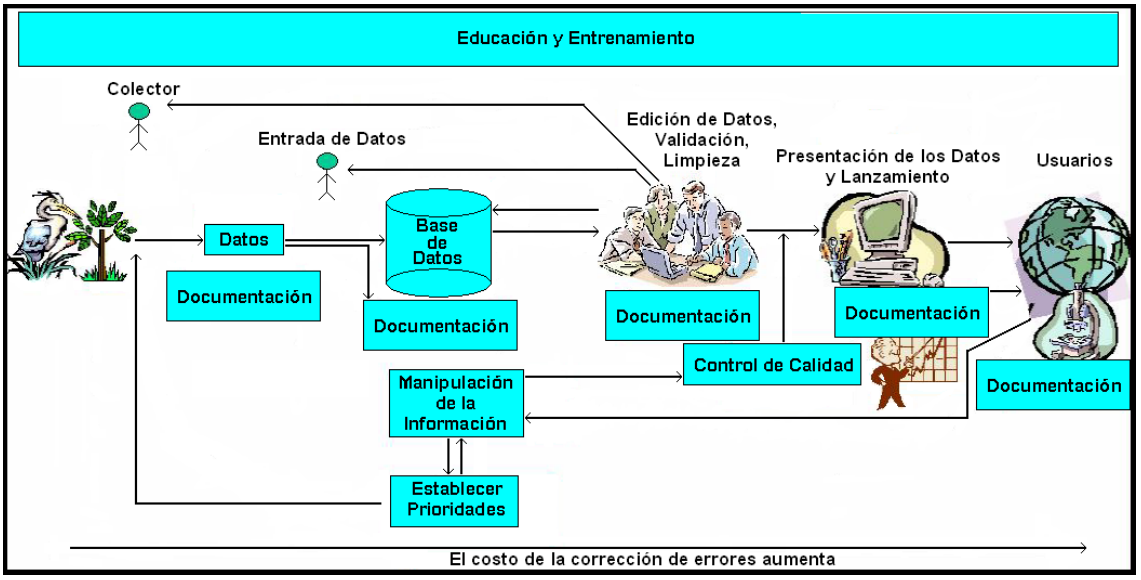


Figura 1: Cadena de Manipulación de la Información, muestra como el costo de la corrección de errores aumenta a medida que se avanza en la cadena. Educación y Entrenamiento son necesarios en todos los pasos (Principles of Data Cuality).

– La planificación es esencial (desarrollar una visión, política y estrategia)

Una buena planificación es parte esencial de una buena política de manipulación de datos. La Cadena de Manipulación de la Información incluye la limpieza de datos como una parte central que necesita ser incorporada en la visión de la organización de los datos y la política. Una estrategia para implementar limpieza de datos y validación dentro de una cultura de organización puede mejorar la calidad de la organización de los datos.

– Organizar los datos mejora la eficiencia

Organizar los datos antes de chequearlos, validarlos y corregirlos puede mejorar la eficiencia y reducir considerablemente el tiempo y costo de la limpieza de datos.

– La prevención es mejor que la cura

Es más eficiente y menos costoso prevenir un error antes que ocurra, que detectarlo y corregirlo después. Es muy importante que funcionen los mecanismos de retroalimentación para asegurar que los errores no ocurran de nuevo durante el proceso de entrada.

– La responsabilidad es de todos (colector, administrador y usuario)

La responsabilidad de la limpieza de los datos pertenece a todos. La principal responsabilidad recae en el administrador. El colector también tiene

responsabilidad y esta consiste en responder al administrador cuando este encuentre un error o ambigüedades, para entonces deben ir atrás, donde está la información recogida por él. El usuario por su parte tiene la responsabilidad clave de retroalimentar a los administradores de algún error u omisión.

– La asociación mejora la eficiencia.

Las asociaciones son un método muy eficiente para manipular la limpieza de los datos, como ya se ha mencionado los usuarios son quienes están en la mejor posición de encontrar los errores en los datos. Si los administradores desarrollan una política de asociaciones entre ellos y los usuarios claves, entonces estos errores no serán ignorados, serán más fáciles de documentar y corregir, y no será necesario duplicar algunos procesos de validación.

– El establecimiento de prioridades, reduce la duplicación

Así como con la organización y el ordenamiento, el establecimiento de prioridades ayuda a reducir los costos y mejorar la eficiencia. Concentrarse en la limpieza de aquellos registros donde grandes cantidades de datos pueden ser limpiados con un mínimo costo (aquellos que se limpian con métodos automatizados) antes de ir a los más complicados. Esto mejora las relaciones entre cliente-proveedor e incentiva a los proveedores a seguir mejorando la calidad de los datos debido a que estos tienen un uso inmediato.

– Establecer objetivos y desarrollar medidas

El desarrollo de medidas es una adición valiosa en los procedimientos de control de calidad y ayuda a dirigir el proceso de limpieza de datos. El desarrollo de medidas puede incluir hasta el chequeo estadístico de los datos[7].

– Minimizar la repetición del proceso de limpieza de los datos

La repetición del proceso de limpieza de datos en la mayoría de las organizaciones es el principal factor de encarecimiento del mismo. La documentación de los procedimientos de validación reduce grandemente el re-manejo de los datos. Experiencias en el mundo de los negocios dicen que el uso de la Cadena de Manipulación de la Información puede reducir la duplicación y el re-manejo de los datos hasta un 50%, y una reducción de los costos al usar datos pobres de hasta dos terceras partes. Esto es principalmente debido a que se gana eficiencia a través de la asignación de responsabilidades para la manipulación de los datos y

control de la calidad, minimizando los cuellos de botella y los tiempos de cola, minimizando la duplicación mediante diferente personal re-haciendo los chequeos de control de calidad y mejorando la identificación de los métodos de trabajo [8].

– La retroalimentación es un camino de dos vías

Los usuarios finales de los datos también encuentran errores, por tanto es muy importante que existan vías para que estos comuniquen a los administradores de los mismos, además que es más probable que uno de estos usuarios buscando determinado dato encuentre un error que un administrador trabajando en solitario. Es de esta forma en que la incidencia de futuros errores puede ser reducido y sobre todo la calidad de los datos puede ser mejorada [7].

– Técnicas para mejorar el entrenamiento y la educación

Un pobre entrenamiento, especialmente en la recolección de datos y en la entrada de los datos (en la Cadena de Calidad de la Información), es la causa de la gran proporción de errores en los datos primarios. Los colectores de datos deben ser educados en los requerimientos de dichos datos, para que los correctos sean los recogidos. Un buen entrenamiento por parte de los operadores de entrada puede reducir los errores de forma considerable, reducen los costos de la entrada de datos y mejoran por sobre todo la calidad de los mismos.

– Responsabilidad, transparencia, habilidad en la auditoria

Responsabilidad, transparencia, habilidad en la auditoria, son elementos esenciales en la limpieza de datos. Los ejercicios de limpieza de datos casuales y no planeados son ineficientes y generalmente improductivos. Dentro de las políticas de calidad de datos y estrategias, líneas de responsabilidad concretas deben ser establecidas. Para mejorar la “aptitud para el uso” de los datos, además de su calidad, el proceso de limpieza de datos debe de ser transparente y bien documentado dentro de una senda auditada para reducir la duplicación, y asegurar que una vez corregidos, los errores no se repitan.

– Documentación

La documentación es la clave para una buena calidad en los datos, sin una buena documentación es muy difícil para los usuarios determinar el grado de corrección de dichos datos, y se hace difícil para el administrador saber cuál y por quién el chequeo de calidad de los datos se ha llevado a cabo. La documentación

generalmente es de dos tipos, el primero está unido a cada registro, registros en los cuales el chequeo de los datos ya se ha llevado a cabo y que cambios han sido hechos y por quién, el segundo tipo es el metadato que registra la información al nivel de grupo de datos. Ambos tipos de documentación son importantes, y sin ellos una buena calidad de los datos se vería comprometida.

## 1.3 Taxonomía de errores.

Son múltiples las formas en que se agrupan y presentan los errores que la limpieza de datos trata de corregir.

Una clasificación de los tipos de errores fundamentales sobre los que la limpieza de datos trabaja los agrupa en [9]:

### 1.3.1 Anomalías sintácticas:

**Errores léxicos**: nombran las discrepancias entre la estructura de los datos y el formato especificado, es decir, el número de valores es inesperado (mayores o menores) para una tupla t, o, el grado de una tupla #t es diferente de #R, el grado del esquema de relación previsto para la tupla.

**Errores en el formato del dominio**: especifica los errores donde el valor dado por un atributo A no se ajusta con el dominio de formato previsto G(dom(A)).

**Irregularidades**: son las relacionadas con el uso no uniforme de valores, unidades (de medida, de peso, etc.) y abreviaturas. Esto pasa por ejemplo, si usamos diferentes tipos de monedas para especificar el salario de los empleados de una determinada empresa, lo cual se convierte en un problema mayor si los tipos de moneda no son explícitamente listados con cada valor correspondiente. Esto resulta en valores con una representación correcta de los hechos si tenemos el conocimiento necesario acerca su representación para poder interpretarlos. Otro ejemplo es el uso o el diferente uso de las abreviaturas.

### 1.3.2 Anomalías semánticas:

**Violaciones de las restricciones de integridad**: describe tuplas (o grupos de tuplas) que no satisfacen una o más de las restricciones de integridad. Las restricciones de integridad son usadas para describir nuestro entendimiento del mini-mundo mediante el conjunto de instancias válidas. Cada restricción es una regla que representa el

conocimiento acerca del dominio y los valores permitidos para una representación certera de los hechos.

**Contradicciones**: son valores dentro de una tupla o entre tuplas que violan algún tipo de dependencia entre valores. Un ejemplo del primer caso pudiera ser una contradicción entre el atributo EDAD y FECHA_NACIMIENTO para una tupla que representa personas. Contradicciones son: violaciones de la dependencia funcional que pueden ser representadas como restricciones de integridad o duplicados con valores inexactos.

**Duplicados**: son dos o más tuplas representando la misma entidad del mini-mundo. Los valores de estas tuplas no necesariamente deben ser idénticos. Los duplicados inexactos son casos específicos de contradicción entre dos o más tuplas. Ellos representan la misma entidad pero con diferentes valores para todas o algunas de sus propiedades. Esto dificulta la detección de duplicados y su fusión.

**Tuplas no válidas**: representan la anomalía más complicada encontrada en colecciones de datos. Por no válidas queremos decir aquellas tuplas que no muestran anomalías del tipo de las definidas anteriormente pero todavía no representan entidades válidas del mini-mundo. Las tuplas no válidas pueden además representar excepciones y por consiguiente no deben considerarse como errores.

### 1.3.3 Anomalías de alcance:

**Valores que faltan**: son el resultado de omisiones en el proceso de colecta de datos.

**Tuplas que faltan**: resulta de las omisiones de entidades completas existentes en el mini-mundo que no son representadas por tuplas en la colección de datos [8].

Existen otras taxonomías o clasificaciones de errores que en esencia se resumen en la descrita anteriormente, ejemplo de estas son las dadas en [10], [11].

## 1.4 Métodos usados para la limpieza de datos

Existen una multitud de métodos distintos usados dentro del proceso de limpieza de datos. Aquí daremos un pequeño resumen de los métodos más populares.

### 1.4.1 Análisis Gramatical (*Parsing*)

El análisis gramatical es desarrollado para la detección de errores sintácticos. Un analizador para una gramática G es un programa que decide para una cadena dada si

es un elemento del lenguaje definido por la gramática G. en el contexto de los compiladores para los lenguajes de programación las cadenas representan programas. En el proceso de limpieza de datos las cadenas son o tuplas completas de una instancia relacional o valores atributos de un dominio.

La existencia de un gran número de errores sintácticos en una colección de datos depende de la extensión del esquema aplicado en el ambiente donde los datos son mantenidos. Si los datos son salvados en ficheros planos existe la posibilidad de errores léxicos y de dominio. En este caso, una gramática derivada de la estructura del fichero es usada y las cadenas representan tuplas completas. Los datos son manejados por sistemas de administración de bases de datos, y estos no esperan que los datos contengan errores léxicos o de dominio, pero los errores de este tipo pueden existir para cada uno de los atributos [12], [13].

### 1.4.2 Transformación de datos *(Data Transformation)*

La transformación de datos propone transformar los datos del formato dado a un formato esperado por la aplicación. Esto involucra el esquema de las tuplas, así como el dominio de sus valores. El esquema de transformación es además desarrollado conjuntamente con la limpieza de datos. Los datos de varios esquemas son transformados en un esquema común más adecuado a las necesidades de la aplicación proyectada. La corrección de los valores ha de ser desarrollada solo en casos donde los datos de entrada no corresponden con el esquema y puedan llevar a fallos posteriores en el proceso de transformación. Esto hace que la limpieza de datos y el esquema de transformación se vean como tareas suplementarias.

La estandarización y la normalización son transformaciones en el nivel de instanciación utilizadas con la intención de eliminar irregularidades en los datos. Esto incluye conversión de valores simples o funciones de traducción, así como, normalizar valores numéricos que están en un intervalo fijo dado por un valor máximo y un mínimo [14], [15].

### 1.4.3 Aplicación de las restricciones de integridad *(Integrity Constraint Enforcement)*

La aplicación de las restricciones de integridad describe el problema de garantizar el cumplimiento de las restricciones, después de transacciones, modificando la colección

de datos, insertando, borrando o actualizando tuplas. Las dos diferentes soluciones son: chequeo de las restricciones de integridad y mantenimiento de las restricciones de integridad. El chequeo de las restricciones de integridad rechaza las transacciones que si se efectúan pueden violar alguna restricción de integridad, mientras que el mantenimiento de las restricciones de integridad tiene que ver con la identificación de las actualizaciones adicionales (por ejemplo: reparaciones), para ser añadidas a la transacción original y garantizar que la colección de datos resultante no viole alguna restricción de integridad [16].

### 1.4.4 Eliminación de duplicados *(Duplicate Elimination)*

Existen varios métodos para la eliminación de duplicados o encadenamiento de registros (*record linkage*). Cada método de detección de duplicados propuesto requiere de un algoritmo para determinar si dos o más tuplas son representaciones duplicadas de la misma entidad. Para una detección eficiente de duplicados, cada tupla ha de ser comparada con todas las demás usando su método de detección de duplicados. A continuación se explican algunos de estos:

*Sorted Neighbourhood Metod* [17]: es un método rápido, y reduce el número de comparaciones requeridas, mediante el ordenamiento de las tuplas por una llave construida de los atributos de la relación, que brinda los duplicados cerca unos de otros, entonces solo las tuplas que están en una ventana son comparadas entre ellas para encontrar los duplicados. La identificación de tuplas duplicadas se hace usando reglas basadas en el conocimiento específico del dominio. En aras de mejorar la precisión, los resultados de varios pases de la detección de duplicados puede ser combinada con el cálculo de cercanía transitiva de todos los pares de tuplas duplicadas encontradas. No mucho se ha dicho de este método en como los duplicados se fusionan.

Otro método es extendiendo el anterior a tuplas que no cumplen con el formato de dominio, porque las tuplas con errores en el formato de dominio que pudieran ser duplicados, pueden no caer cerca unas de otras después del ordenamiento. Por tanto, los atributos son llevados a unidades léxicas y entonces ordenados dentro de cada atributo antes de ser ordenada la relación completa.

Otro método es el de eliminación de duplicados borrosos (*Fuzzy Duplicates*), que propone una solución que nos evita los problemas de los métodos de ordenamiento. Los cuales confían en las dimensiones jerárquicas típicamente asociadas con las tablas

dimensionales en un almacén de datos. Estas son jerarquías de tablas típicamente en relaciones 1-n expresadas por relaciones entre llaves (de llaves extranjeras). Cada tupla en la relación 1 es asociada con un grupo de tuplas de la relación n. El grado de solapamiento entre grupos asociados con dos tuplas de la relación 1 es una medida de la co-ocurrencia entre ellos, y puede ser usada para detectar duplicados.

### 1.4.5 Métodos estadísticos *(Statistical Metods)*

Los métodos estadísticos pueden ser usados para la verificación de los datos, así como en la corrección de anomalías. La detección y eliminación de complejos errores que representan tuplas no válidas va más allá del chequeo y la aplicación de las restricciones de integridad, a menudo involucran relaciones entre dos o más atributos que son difíciles de descubrir y describir por las restricciones de integridad. Esto puede ser visto como un problema en la detección perfilada, como por ejemplo, una minoria de las tuplas y valores que no están acordes a las características generales de una colección de datos dada.

Analizando los datos usando los valores de la media, la desviación estándar, el rango, o algoritmos de agrupamiento, un experto de dominio (domain expert) puede encontrar valores que son inesperados indicando posibles tuplas no válidas. La corrección de estos errores es muchas veces imposible (excepto si simplemente los borramos) porque los verdaderos valores son desconocidos. Una solución posible incluye métodos estadísticos como poniendo los valores en algún valor medio u otra medida estadística. Valores por fuera pueden ser detectados como violaciones de las reglas de asociación u otros patrones existentes en los datos.

Otra anomalía manipulada por los métodos estadísticos son los valores que faltan (missing values), estos valores son manipulados basados en la entrada de uno o más valores posibles [18].

## 1.5 Herramientas de limpieza de datos

En esta sección describimos algunos de los proyectos existentes de limpieza de datos.

### 1.5.1 AJAX

AJAX [19], [20] es una armazón (*framework*) flexible y extensible que intenta separar los niveles lógicos y físicos de la limpieza de datos. El nivel lógico sustenta el diseño del

flujo de trabajo de la limpieza de datos y la especificación de las operaciones de limpieza desarrolladas, mientras que en el nivel físico recae su implementación. El mayor interés de AJAX es transformar los datos existentes de una o mas colecciones de datos en un esquema destino y eliminar los duplicados dentro de este proceso. Para este propósito, se define un lenguaje declarativo basado en un grupo de operaciones de transformación, las cuales son: *mapping*, *view*, *matching*, *clustering* y *merging*.

El operador *mapping* expresa arbitrarios mapeos (uno-a-muchos) entre una relación de entrada simple y una o más relaciones de salida. El operador *view* es equivalente al view de SQL simplemente expresando los mapeos limitados a muchos-a-uno con un chequeo adicional de integridad. El operador *matching* calcula aproximadamente una unión entre dos relaciones asignando un valor distancia a cada par en el producto cartesiano usando una función de distancia arbitraria. Este operador es fundamental en la detección de duplicados. El operador *merge* toma una simple relación como entrada, la particiona acorde a los atributos de agrupamiento y luego contrae cada partición en una tupla simple usando una función de agregación arbitraria. La semántica de los cinco operadores incluye la generación de excepciones que proveen la interacción con el usuario experto.

El proceso de limpieza de datos es especificado colocando las operaciones de transformación como un grafo de flujo de datos lineal con cada operación, tomando la salida de una o más operaciones precedentes como su entrada. Un mecanismo de linaje de datos permite al usuario inspeccionar las excepciones, analizar su proveniencia en el proceso de limpieza de datos y luego corregir las tuplas que contribuyeron a su generación. Los datos corregidos pueden ser re-integrados dentro del mismo proceso de limpieza de datos.

### 1.5.2 FraQL

FraQL [21], [15] es otro lenguaje declarativo de apoyo a la especificación del proceso de limpieza de datos. El lenguaje es una extensión del SQL basado en un modelo de datos objeto-relacional. Soporta la especificación de esquemas de transformación así como transformaciones de datos a nivel de instancias, como por ejemplo estandarización y normalización de valores. Esto puede hacerse mediante funciones definidas por el usuario, las cuales deben hacerse para los requerimientos específicos del dominio dentro del proceso individual de limpieza de datos.

Con sus operadores extendidos *union* y *join* en conjunción con las funciones de conciliación definidas por el usuario, FraQL maneja la detección y eliminación de duplicados. Muy similar al SQL los operadores mencionados anteriormente pueden ser redefinidos mediante una cláusula *on* especificando los atributos de comparación (para el operador union), a estos también se les puede aplicar una cláusula adicional *reconciled by*, la cual denota una función definida por el usuario para la resolución de contradicciones entre tuplas que cumplen las cláusulas de comparación.

### 1.5.3 Potter's Wheel

Potter's Wheel [13] es un sistema interactivo de limpieza de datos que integra la transformación de datos y la detección de errores usando una hoja de cálculo como interfase. Los efectos de las operaciones desarrolladas son mostrados inmediatamente en tuplas visibles en pantalla. La detección de errores es hecha para la colección de datos completa de forma automática como un proceso de fondo. Un grupo de operaciones son especificadas y soportan los esquemas de transformaciones comunes sin una programación explícita.

Las especificaciones para el proceso de limpieza de datos son hechas de forma interactiva. La retroalimentación inmediata de las transformaciones llevadas a cabo y las detecciones de errores permiten a los usuarios un desarrollo gradual y pulir el proceso. Esto permite la reacción individual ante excepciones. El proceso completo de limpieza no está documentado.

### 1.5.4 ARKTOS

ARKTOS [22] es un armazón (*framework*) capaz de modelar y ejecutar el proceso de Extracción-Transformación-Carga (*ETL process* (*Extraction-Transformation-Load*)) para la creación de un almacén de datos (*data warehouse*). Los autores consideran que la limpieza de datos es una parte integral de este proceso de ETL el cual consiste en simples pasos para extraer datos relevantes de la fuente, transformarlo en el formato destino y limpiarlo, para luego cargarlo dentro del almacén de datos. Un meta-modelo es especificado la modelación del proceso completo de ETL. Las operaciones de limpieza dentro del proceso son llamadas actividades. Cada actividad es enlazada a las relaciones de entrada y de salida. La lógica desarrollada por una actividad es descrita declarativamente mediante una instrucción SQL. Cada instrucción está asociada con un

tipo de error particular y una política que específica el comportamiento (la acción que será desarrollada) en caso de la ocurrencia de un error.

Las políticas para la corrección de errores simplemente son: IGNORAR, pero sin marcar explícitamente la tupla errónea, BORRAR así como ESCRIBIR A FICHERO e INSERTAR A UNA TABLA con las semánticas esperadas. Las últimas dos proveen la única posibilidad de interactuar con el usuario.

### 1.5.5 IntelliClean

IntelliClean [23], [24] es un software de limpieza de datos basado en reglas con un mayor enfoque en la eliminación de duplicados. El armazón (*framework*) propuesto consta de tres etapas. En la etapa de pre-procesamiento los errores sintácticos son eliminados y los valores estandarizados en formato y consistencia del uso de abreviaturas. La etapa de procesamiento representa la evaluación de las reglas de limpieza en los datos condicionados que especifican acciones a tomar bajo determinadas circunstancias. Existen cuatro tipos de reglas. Las reglas de identificación de duplicados especifican las condiciones bajo las cuales las tuplas se consideran como duplicadas. Las reglas de fusión/depuración especifican como las tuplas duplicadas van a ser manipuladas. Las reglas de actualización especifican la forma en que los datos van a ser actualizados en una particular situación, esto habilita la especificación de las reglas de puesta en vigor de las restricciones de integridad, para cada restricción de integridad una regla define una regla de actualización define como modificar la tupla para así satisfacer la restricción. Las reglas de alerta especifican las condiciones bajo las cuales el usuario es notificado para ciertas acciones.

Durante las dos primeras etapas del proceso de limpieza de datos las acciones tomadas fueron registradas, proporcionando documentación de las acciones desarrolladas. En la etapa de verificación humana y validación estos registros son investigados para verificar y posiblemente corregir las acciones desarrolladas.

### 1.5.6 Comparación de las herramientas

En la tabla que sigue los diferentes software de limpieza de datos descritos anteriormente son comparados en cuanto a las anomalías manipuladas por ellos y una pequeña indicación acerca de los métodos y técnicas usadas o definidas por cada uno

de ellos. El término *Definido por el Usuario* indica que detectar y eliminar esa anomalía en específico es posible pero no especificado en detalle.

| | **AJAX** | **FraQL** | **Potter's Wheel** | **ARKTOS** | **IntelliClean** |
|---|---|---|---|---|---|
| **Errores Léxicos** | | | | | |
| **Errores en el Formato de Dominio** | Definido por el Usuario | Definido por el Usuario | Aprendizaje de Patrones | Definido por el Usuario | Definido por el Usuario |
| **Irregularidades** | Definido por el Usuario | Definido por el Usuario | | | Definido por el Usuario |
| **Violación de Restricciones** | Filter violating tuples | | | Tres tipos de restricciones | Regla de Alerta y Actualización |
| **Valores que Faltan** | Definido por el Usuario | Valores Estadísticos | | | Regla de Actualización |
| **Tuplas que Faltan** | | | | | |
| **Duplicados** | Match, Cluster and Merge | Union, Join and Reconciliation | | | Merge/ Purge rule |
| **Tuplas no Válidas** | | Métodos Estadísticos | | | |

# Capítulo 2: Análisis y diseño de la Herramienta “DB Analyzer”

En este capítulo desarrollamos el Análisis y diseño de la Herramienta que proponemos para lograr el análisis de los datos de las Bases de datos que permitirán dar una conclusión preliminar de los principales tipos de errores en el entorno de los Sistemas Informativos de nuestro entorno.

## 2.1 *Casos de Uso*

En la modelación de este sistema se utilizó la notación del UML (Unified Modeling Language), que es un lenguaje visual estándar que se utiliza para especificar, visualizar, construir y documentar los diferentes aspectos relativos al desarrollo de un software

Una técnica del UML, que permite mejorar la comprensión de los requerimientos del sistema, es la identificación de casos de uso y actores. Los casos de usos son los procesos que debe llevar a cabo la aplicación en los que toma parte cada uno de los actores (agentes externos). Normalmente un actor estimula al sistema con eventos de entradas o recibe algo de él.

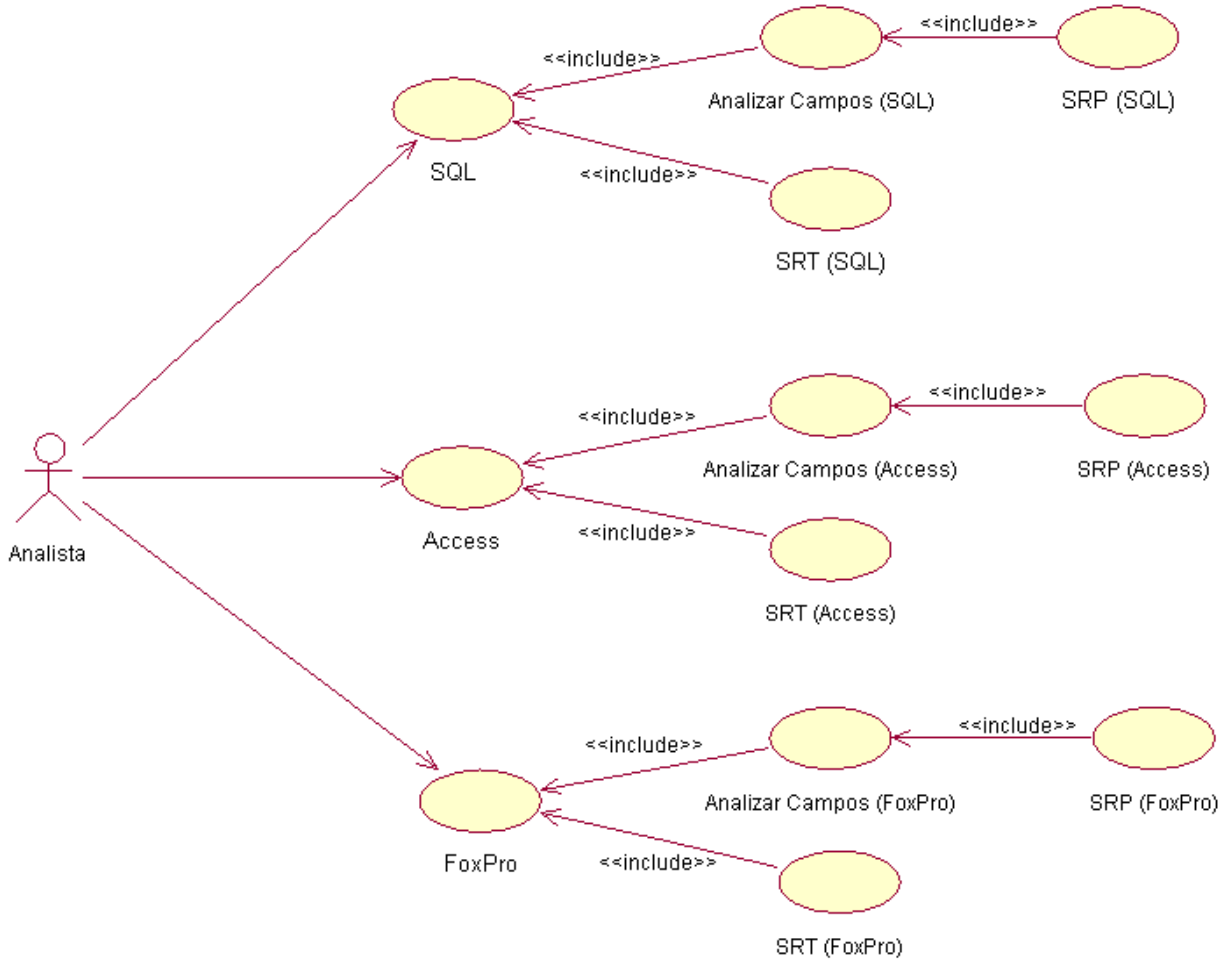


**Figura 2.1** Casos de uso para el actor Analista.

Para la herramienta que se presenta, identificamos 9 casos de uso y 1 actor, que es el Analista. En la Figura 2.1 se presenta el Diagrama de Casos de Uso del actor Analista, a continuación describimos los casos de uso para este actor:

- SQL: caso de uso en el cual se inicializa la conexión con el servidor de SQL.
- Access: caso de uso mediante el cual se establece la conexión con el fichero de tipo Access.
- FoxPro: caso de uso mediante el cual se establece la conexión con el fichero (.dbf).
- Analizar campos (SQL): este caso de uso se puede ejecutar después del caso de uso SQL, y sus precondiciones son: que se haya seleccionado un o más campos de la tabla escogida, y las acciones que realiza es mostrar los resultados del análisis a los campos seleccionados.

- SRT (SQL): este caso de uso se puede ejecutar después del caso de uso SQL, la acción que realiza es salvar el resultado del análisis a la base de datos completa.
- SRP (SQL): este caso de uso se puede ejecutar después del caso de uso Analizar campos (SQL) y la acción que realiza es salvar el resultado del análisis parcial efectuado.
- Analizar campos (Access): este caso de uso se puede ejecutar después del caso de uso Access, y sus precondiciones son: que se haya seleccionado un o más campos de la tabla escogida, y las acciones que realiza es mostrar los resultados del análisis a los campos seleccionados.
- SRT (Access): este caso de uso se puede ejecutar después del caso de uso Access, la acción que realiza es salvar el resultado del análisis a la base de datos completa.
- SRP (Access): este caso de uso se puede ejecutar después del caso de uso Analizar campos (Access) y la acción que realiza es salvar el resultado del análisis parcial efectuado.
- Analizar campos (FoxPro): este caso de uso se puede ejecutar después del caso de uso FoxPro, y sus precondiciones son: que se haya seleccionado uno o más campos de la tabla escogida, y las acciones que realiza es mostrar los resultados del análisis a los campos seleccionados.
- SRT (FoxPro): este caso de uso se puede ejecutar después del caso de uso FoxPro, la acción que realiza es salvar el resultado del análisis a la base de datos completa.
- SRP (FoxPro): este caso de uso se puede ejecutar después del caso de uso Analizar campos (FoxPro) y la acción que realiza es salvar el resultado del análisis parcial efectuado.

## 2.2 Clases que conforman el sistema

La herramienta de trabajo diseñada prepara la información, de forma tal que al leer de una determinada base de datos un campo, los contenidos de este se guardan en dos arreglos, uno que almacena el valor leído y el otro la cantidad de veces que aparece dicho valor, así, si el campo es de tipo entero, el arreglo de los elementos sería ListInt y el de la cantidad de repeticiones de cada elemento CantInt, en el caso de un campo de

tipo real sería: ListReal (arreglo de reales) y CantReal (arreglo de enteros) y un campo de tipo cadena sería ListStr (arreglo de cadenas) y CantStr (arreglo de enteros).

Las clases que conforman el sistema son las que a continuación se relacionan:

1. TArrayEntero: en esta clase están todas las operaciones que se van a hacer sobre los elementos de tipo entero. A continuación se muestran los métodos de esta clase:
   - function Count (CantInt: IntArr): Integer;
     función que se le pasa como parámetro un arreglo de enteros, y devuelve un entero con la suma de cada uno de los elementos de dicho arreglo.
   - function CountZero (ListInt, CantInt: IntArr): Integer;
     función que se le pasan como parámetros dos arreglos de entero y devuelve un entero, donde busca en el primer arreglo la aparición del elemento 0, y en la posición en que aparezca busca en el otro arreglo y devuelve ese valor, si en el primer arreglo el 0 no está, entonces devuelve 0.
   - function Media (ListInt, CantInt: IntArr): Real;
     función a la que se le pasan como parámetros dos arreglos de enteros y devuelve un real, ese valor representa la media .
   - function Moda (ListInt, CantInt: IntArr): Integer;
     función a la que se le pasan como parámetros dos arreglos de enteros y devuelve un entero, donde en el segundo arreglo busca el mayor elemento, y en la posición en que este aparece devuelve el valor correspondiente al otro arreglo.
   - function Maximo (ListInt: IntArr): Integer;
     función a la que se le pasa como parámetro un arreglo de enteros y devuelve un entero que es el mayor elemento que esté en dicho arreglo.
   - function Minimo (ListInt: IntArr): Integer;
     función a la que se le pasa como parámetro un arreglo de enteros y devuelve un entero que es el menor elemento que esté en dicho arreglo.
   - function Cardinalidad (CantInt: IntArr): Integer;
     función a la que se le pasa como parámetro un arreglo de enteros y devuelve un entero que es la cantidad de elementos que tenga el arreglo.
   - procedure DesviacionStandard(ListInt, CantInt: IntArr; Media: Real;var DesvStand: Real;var debajo,encima: Real);

procedimiento al que le paso dos arreglos de tipo entero y una variable de tipo real que representa el valor de la media, y me devuelve tres valores reales, el primero representa la desviación estándar, el segundo la frontera inferior y el tercero la frontera superior, estos dos últimos valores se utilizan para encontrar los elementos fuera de rango.

2. TArrayReal

- function Count (CantReal: IntArr): Integer;

  función que se le pasa como parámetro un arreglo de enteros, y devuelve un entero con la suma de cada uno de los elementos de dicho arreglo.

- function Media (ListReal: RealArr; CantReal: IntArr): Real;

  función a la que se le pasan como parámetros dos arreglos, uno de enteros y el otro de reales y devuelve un real, ese valor representa la media.

- function Moda (ListReal: RealArr; CantReal: IntArr): Real;

  función a la que se le pasan como parámetros dos arreglos, uno de enteros otro de reales y devuelve un real, donde en el segundo arreglo busca el mayor elemento, y en la posición en que este aparece devuelve el valor correspondiente al otro arreglo.

- function Maximo (ListReal: RealArr): Real;

  función a la que se le pasa como parámetro un arreglo de reales y devuelve un real que es el mayor elemento que esté en dicho arreglo.

- function Minimo (ListReal: RealArr): Real;

  función a la que se le pasa como parámetro un arreglo de reales y devuelve un real que es el menor elemento que esté en dicho arreglo.

- function Cardinalidad (CantReal: IntArr): Real;

  función a la que se le pasa como parámetro un arreglo de enteros y devuelve un entero que es la cantidad de elementos que tenga el arreglo.

- procedure DesviacionStandard(ListReal: RealArr; CantReal: IntArr; Media: Real; var DesvStand: Real; var debajo, encima: Real);

  procedimiento al que le paso dos arreglos, uno de tipo entero y uno de tipo real, y una variable de tipo real que representa el valor de la media, y me devuelve tres valores reales, el primero representa la desviación estándar, el segundo la frontera inferior y el tercero la frontera superior, estos dos últimos valores se utilizan para encontrar los elementos fuera de rango.

3. TArrayStrClass
   - function CantNull (ListStr: StrArr; CantStr: IntArr): Integer;
     función que se le pasan como parámetros dos arreglos, una de cadenas y otro de enteros y devuelve un entero, donde busca en el primer arreglo la aparición del elemento 'NUL' o 'NULO', y en la posición en que aparezca busca en el otro arreglo y devuelve ese valor, si en el primer arreglo no está el valor nulo, entonces devuelve 0.
   - function Count (CantStr: IntArr): Integer;
     función que se le pasa como parámetro un arreglo de enteros, y devuelve un entero con la suma de cada uno de los elementos de dicho arreglo
   - function Cardinalidad (CantStr: IntArr): Integer;
     función a la que se le pasa como parámetro un arreglo de enteros y devuelve un entero que es la cantidad de elementos que tenga el arreglo.
   - Procedure ValuePerCent (CantStr:IntArr; var b: RealArr);
     procedimiento al que se le pasa un arreglo de enteros y devuelve un arreglo de reales, donde en cada posición devuelve el por ciento de ocurrencia de cada cadena.
4. TFormaVisual
   - procedure TFormaVisual.ButtonSQLClick(Sender: TObject);
     método mediante el cual nos conectamos al servidor de SQL.
   - procedure TFormaVisual.ButtonAccessClick(Sender: TObject);
     método mediante el cual nos conectamos a un fichero de tipo Access.
   - procedure TFormaVisual.ButtonFoxProClick(Sender: TObject);
     método mediante el cual nos conectamos a un fichero de tipo (.dbf).
   - procedure TFormaVisual.ButtonSQLSelectFieldsClick(Sender: TObject);
     método mediante el cual se le hacen los análisis correspondientes a los campos seleccionados de una tabla en SQL y se muestran en un Memo.
   - procedure TFormaVisual.ButtonAccessSelectFieldsClick(Sender: TObject);
     método mediante el cual se le hacen los análisis correspondientes a los campos seleccionados de una tabla en Access y se muestran en un Memo.
   - procedure TFormaVisual.ButtonFoxProSelectFieldsClick(Sender: TObject);
     método mediante el cual se le hacen los análisis correspondientes a los campos seleccionados de un fichero en FoxPro y se muestran en un Memo.

- procedure TFormaVisual.ButtonSalvarRPClick(Sender: TObject);

  método mediante el cual se salvan en un fichero texto los análisis correspondientes a los campos seleccionados (lo que se muestra en un Memo).

- procedure TFormaVisual.ButtonSalvarRTClick(Sender: TObject);

  método mediante el cual se salvan los análisis correspondientes a la base de datos seleccionada y se salva en un fichero texto.

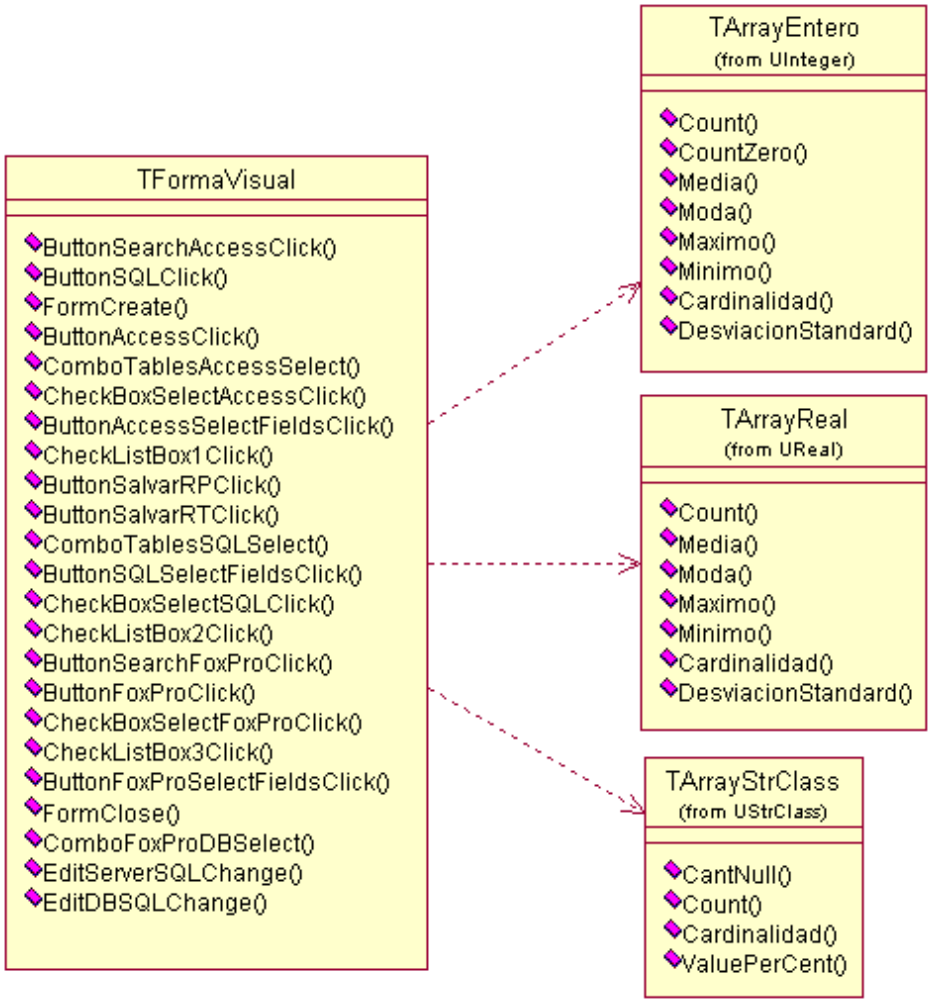


**Figura 2.2** Diagrama de Clases

## 2.3 *Diagrama de Estado o Actividad*

Los diagramas de actividad proporcionan una forma de modelar el flujo de la información dentro de un proceso. Por ejemplo, una compañía podría usar diagramas de actividad para modelar el flujo para una aprobación de pedidos. Una firma de

contabilidad podría usar diagramas de actividad para modelar cualquier número de transacciones financieras. Una compañía de software podría usar diagramas de actividad para modelar la parte de un proceso de desarrollo de software.

Un diagrama de actividad es considerado un caso especial de una máquina de estado en la cual la mayor parte de los estados son actividades y la mayor parte de las transiciones son implícitamente provocadas por la finalización de las acciones en las actividades. Un diagrama de actividad es típicamente usado para modelar la secuencia de actividades en un proceso.

En nuestro caso el diagrama de estado de la figura 2.3 nos guiará en la secuencia lógica para un correcto uso de la herramienta.

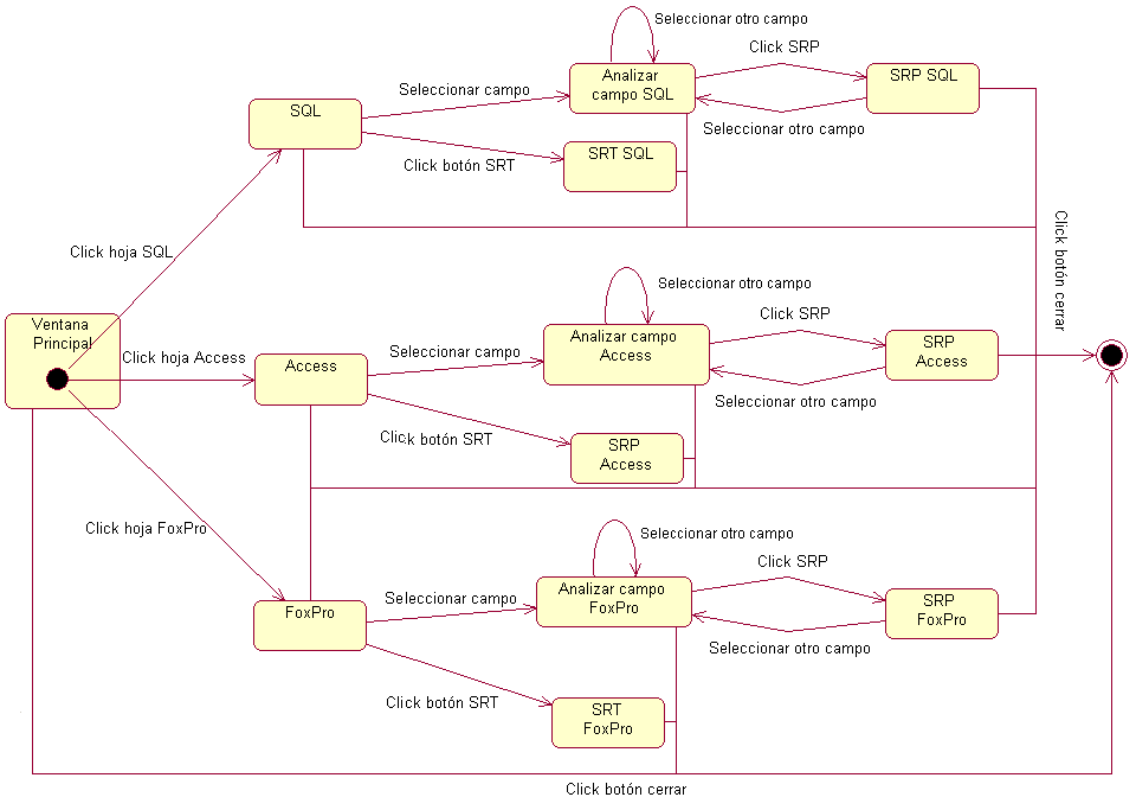


**Figura 2.3** Diagrama de Estado.

# Capítulo 3: Resultados obtenidos con la Herramienta “DB Analyzer” y propuesta de taxonomía.

La limpieza de datos tiene varias fases, la primera es una de las más importantes pues constituye la base para el resto: el análisis de los datos. En esta fase se determinan los problemas fundamentales que pueden presentar los datos. Una forma de hacerlo es realizar algunas pruebas a los datos de las que se obtiene un perfil de los mismos. Estos resultados son sumamente importantes para determinar los métodos y las herramientas que son necesarias para la limpieza.

En general la limpieza de datos incluye varias etapas [Rah00]:

1. Análisis de los Datos.
   Es necesario para determinar que tipos de errores e inconsistencias deber ser limpiados.

2. Definición de flujos de transformación y reglas de mapeo.
   Dependiendo del número de fuentes de datos y de su grado de heterogeneidad y grado de “suciedad” deben realizarse un número grande de transformaciones en los datos entre las que se encuentran la necesidad de hacer corresponder datos de diversas fuentes.

3. Verificación
   Debe ser evaluada la corrección y efectividad del paso anterior.

4. Flujo inverso de datos limpios.
   Después que los datos son limpiados, no solo deben servir para cargar el almacén de datos sino también deben ser utilizados para reemplazar a los datos “sucios” originales.

En una primera aproximación al problema de la creación de herramientas de limpieza para nuestro medio, también este análisis general resulta importante, pues dará cuales son los problemas fundamentales que presentan los datos en los sistemas operacionales y por lo tanto brindará una guía de cómo atacar el problema de la creación de herramientas para la limpieza de los datos.

## 3.1 El análisis de los datos en la limpieza de datos

Los metadatos guardados en el esquema de una base de datos son insuficientes para asegurar la calidad de los datos de dicha base. En la mayoría de los casos, solo algunas reglas de integridad son declaradas y almacenadas en el esquema. Por ello es importante analizar las instancias actuales para obtener metadatos reales (reingeniería) y patrones inusuales de los datos. Los metadatos ayudan a encontrar problemas en la calidad de los datos.

Existen dos formas de realizar análisis de los datos: realizando un perfil de los datos y utilizando minería de datos [Rah00].

El perfil de los datos centra el análisis de las instancias en atributos individuales. Se deriva información tal como: tipo de dato, longitud, rango de valores, valores discretos y su frecuencia, varianza, unicidad, ocurrencia de nulos, patrones típicos de cadena. Todos estos elementos nos pueden dar una visión exacta de los datos, de su calidad. [Kim02]

La minería de datos, por su parte, ayuda a descubrir patrones de datos en conjuntos grandes de datos. Se utilizan modelos descriptivos de minería de datos, los cuales incluyen clustering, sumarización, descubrimiento de asociaciones, descubrimiento de secuencias. Se pueden derivar dependencias funcionales, algunas reglas de negocio, las cuales pueden servir para completar datos ausentes, descubrir datos ilegales, identificar duplicados, etc.

A partir de estas ideas se construyó una herramienta computacional que es capaz de realizar el perfil de datos almacenados en una base.

El perfil de los datos puede ser importante para detectar problemas presentes. En la tabla que sigue se dan algunos ejemplos:

| Problemas | Metadato | Ejemplos/heurísticas |
|---|---|---|
| Valores ilegales | cardinalidad | Ej. Si cardinalidad(sexo)>2 indica problemas. |
| | max, min | Los valores max, min no deben estar fuera del rango permitido. |
| | varianza, desviación | La varianza y desviación no deben ser mayores que el umbral. |
| Errores ortográficos | valores de atributos | Ordenando los valores de un atributo, hace que datos con errores ortográficos queden cerca y puedan detectarse. |
| Valores ausentes | valores nulos | Por ciento / número de valores nulos. |
| | valores por defecto | La presencia de valores por defecto puede indicar realmente un valor nulo. |
| Duplicados | cardinalidad+unicidad | La cardinalidad del atributo debe coincidir con el número de filas. |
| | valores de atributos | Ordenar los valores por el número de ocurrencia; Más de una ocurrencia indica duplicados. |

Esta herramienta realiza un análisis de cada uno de los atributos de la base y determina:

Para todos los datos

- Tipo de dato.
- Cantidad de valores ausentes.
- Cardinalidad.

Para datos numéricos

- Valores máximo y mínimo.
- Valor medio, moda, desviación standard.
- Contar ceros (pueden indicar valores ausentes).

Para datos tipo cadena

- Contar cadenas vacías (pueden indicar valores ausentes).
- Valores y su por ciento.

La herramienta está confeccionada en Delphi y realiza la conexión con la base de datos utilizando la tecnología ADO y ODBC, por lo que para examinar una base de datos determinada solo requerimos tener el driver ODBC para dicho gestor y construir una conexión con esos datos.

Presenta una interfaz amigable con el usuario de tal manera que se puede realizar el análisis de la tabla que se decida y de la columna que se desee. Los datos inicialmente los brinda por pantalla, pero pueden ser almacenados para su posterior análisis.

Evidentemente la herramienta solo brinda información que debe ser procesada conociendo la semántica de los datos, pero estos elementos pueden ser muy útiles en la determinación de la calidad de los datos.

## 3.2 Descripción de la herramienta y sus funcionalidades.

La herramienta que ponemos a su disposición se ha concebido para el análisis de diversos tipos de Bases de Datos: SQL, Access y FoxPro. Para la implementación del análisis a las Bases de Datos de tipo Access y SQL se usó la componente ADO que nos brinda Delphi, mientras que para analizar las BD de tipo FoxPro usamos ODBC por problemas de compatibilidad de ADO con algunos ficheros (.dbf).

Un aspecto a tener en cuenta antes de analizar cualquier tabla de FoxPro (.dbf) es que se tiene que crear primeramente el ODBC correspondiente al driver de FoxPro, y para este ser creado correctamente, en el directorio 'c:\windows\system32' debe existir el archivo 'vfpodbc.dll' (versión 6.0 que tiene un tamaño de 912 KB), el cual brindamos conjuntamente con el software.

El software en dependencia del tipo de campo que esté analizando nos brinda diferentes cálculos estadísticos, así, si el campo analizado es de tipo entero, real o moneda nos dice el valor máximo, el mínimo, la media, la moda, la cantidad de ceros, cuantos elementos vacíos tiene, la cardinalidad (cantidad de valores diferentes), la desviación estándar y los posibles valores que pueden estar fuera de rango, cálculo que se hace teniendo en cuenta el valor de la media y la desviación estándar. Si el campo es de tipo cadena nos dice la cantidad de valores diferentes que hay (cardinalidad), lista cada uno de los valores del campo junto con su por ciento de ocurrencia y la cantidad de veces que aparece, además de cuantos elementos vacíos y nulos tiene. Si es de tipo fecha nos dice cuantas fechas distintas hay y cuantos elementos vacíos tiene dicho campo. Estos datos estadísticos nos pueden ayudar a encontrar o determinar hasta cierto punto cuando un elemento determinado tiene un valor incorrecto o fuera de rango, o se encuentra repetido más veces de las que debiera.

## 3.3 Resultados del trabajo con la herramienta DB Analyzer

Con el objetivo de lograr definir una Taxonomía de errores para las Bases de Datos de nuestro entorno y probar la efectividad de la herramienta diseñada, fueron analizadas 5 bases de datos, de sistemas informáticos realizados en nuestro país, estas son:

- En SQL las bases de datos Zafra y ZafraNueva proporcionadas por el Minaz Provincial.
- En ACCESS la base de datos de Control de estipendios del Sistema de Control de estudiantes de la UCLV.
- En FOXPRO las Bases de datos del Sistema de Control de la Mortalidad nacional de los años 1999 y 2003.

Describimos a continuación el resultado de los análisis realizados a cada una.

### 3.3.1 SQL

La primera de las Bases de Datos se llama Zafra, y está relacionada con el sistema de contabilidad referido a las diferentes instituciones involucradas en el proceso de la zafra. Esta BD tiene un total de 406 tablas y tiene un tamaño en disco de 64 MB. El análisis de la misma con nuestro sistema nos llevó a los siguientes resultados:

- Hay un total de 131 tablas vacías
- Hay un total de 14 tablas con solamente 1 tupla
- Hay un total de 2 tablas con 2 tuplas:
- Hay un total de 6 tablas con uno de sus campos vacíos.
  - La tabla “fin_correccionbanco” tiene 4 campos, 3 tuplas, y el campo vacío es de tipo entero.
  - La tabla “fin_errorbanco” tiene 6 campos, 19 tuplas, y el campo vacío es de tipo moneda.
  - La tabla “gen_histentidad” tiene 4 campos, 13 tuplas y el campo vacío es de tipo entero.
  - La tabla “inv_documentotransfrec” tiene 4 campos, 17 tuplas y el campo vacío es de tipo entero.
  - La tabla “inv_existencialote” tiene 7 campos, 287 tuplas y el campo vacío es de tipo fecha.
  - La tabla “mb_mov_entrada” tiene 27 campos, 241 tuplas y el campo vacío es de tipo cadena.
  - La tabla “rep_tblParameters” tiene 8 campos, 633 tuplas y el campo vacío es de tipo cadena.

El resultado de este análisis sugiere que el diseño de la base de datos es incorrecto.

Ahora vamos a proceder a describir las tablas con los campos que tengan posibles errores, de acuerdo con los otros resultados brindados por la herramienta:

- La tabla “con_comprobante” tiene un total de 6 campos y 4708 tuplas
  - “idunidad” que es de tipo entero el único valor que tiene es 5. Es posible que este campo sea innecesario, por tener el mismo valor en una muestra tan grande.
- La tabla “con_cuentanat” tiene 4 campos y un total de 1358 tuplas:

- "descripcion" es de tipo cadena y tiene una cardinalidad de 527, del análisis de la cardinalidad se observó que existen valores que responden a la misma entidad y están reflejados de forma diferente, por ejemplo:

| Cadena | Veces |
|---|---|
| Santiago | 22 |
| Gramma | 22 |
| Ciego | 20 |
| Habana | 23 |
| Tunas | 9 |
| Santi spiritus | 28 |
| Santiago de cuba | 16 |
| Granma | 16 |
| Las tunas | 29 |
| Ciego de avila | 18 |
| Sancti spiritus | 7 |
| La habana | 15 |

- La tabla "con_apertura" está compuesta por 4 campos y un total de 218 tuplas:
  - "idunidad" que es de tipo entero tiene un total de 186 elementos vacíos, y el único valor que tienen los elementos no vacíos en este campo es 5.
- La tabla "con_comprobanteoperacion" tiene un total de 8 campos y 4708 tuplas:
  - "comentario" es un campo de tipo no manipulado por la herramienta.
- La tabla "con_criterio" tiene tres campos y 683 tuplas:
  - "descripcion" es de tipo cadena y tiene cardinalidad 676, los siguientes elementos son los que se repiten

| Cadena | Veces |
|---|---|
| 33 - Cambio de Utiles para AFT | 7 |

  La aparación de esta descripción con cardinalidad 7 comparada con el resto de las descripciones con cardinalidad 1 nos indica que esto puede ser un error en la información de dicho campo.
- La tabla "con_elementoanalisis" cuenta con 3 campos y 981 tuplas:

- “activo” es de tipo cadena, con cardinalidad 1 y el valor es ‘True’, lo cual puede sugerir que el campo sea innecesario.

- La tabla “con_grupo” tiene 3 campos y 126 tuplas:
  - “descripcion” es de tipo cadena y tiene 125 tuplas, siendo la cadena ‘% SOFTWARE DE LAS PROVINCIAS(TEICO)’ la única que se repite 2 veces.
- La tabla “con_pase” tiene un total de 4 campos y 17591 tuplas:
  - “importe” es de tipo moneda con cardinalidad 14617, y sus valores están entre -668914.16 y 703376.17, siendo 8.75 el valor que más se repite, tiene 14245 posibles valores fuera de rango.
- La tabla “con_saldo” tiene un total de 5 campos y 376 tuplas:
  - “debitos” es de tipo moneda con cardinalidad 368, y sus valores están entre 0 y 2325756.36, siendo el 1000 el más repetido, tiene 355 posibles valores fuera de rango.
  - “creditos” es de tipo moneda con cardinalidad 355, y sus valores están entre -2289893.44 y 0, siendo el -4200 el valor que más se repite.
  - “saldo” es de tipo moneda con cardinalidad 102 y sus valores están entre -465051.66 y 409409.2, siendo el 0 el más repetido.
- La tabla “cos_destinosaldo” con un total de 2 campos y 96 tuplas:
  - “basedistribucion” es de tipo moneda cuyo único valor es 0. de este análisis se puede apreciar que este campo puede ser un campo vacío que está representado por el valor por defecto y es posible que esté de más.
- La tabla “cos_pasecentro” tiene un total de 4 campos y 9950 tuplas:
  - “importe” es de tipo moneda, con cardinalidad 9230 y sus valores están entre -181991.96 y 182841.96, siendo el 248.75 el más repetido, tiene 9229 posibles valores fuera de rango.
- La tabla “cos_registrogasto” con un total de 3 campos y 2044 tuplas, no tiene definido una llave:
  - “importe” es de tipo moneda, con cardinalidad 1878 y sus valores entre -668914.16 y 703376.17, siendo el 7500 el valor que más se repite, y los 1878 son posibles valores fuera de rango.
- La tabla “cos_saldo” con un total de 6 campos y 175 tuplas:

- “saldo” es de tipo moneda, cuyos valores están entre -127372.59 y 127372.59, siendo el 0 el valor que más se repite, tiene 69 posibles valores fuera de rango.
- “debito” es de tipo moneda, con cardinalidad 164 y valores entre 0 y 430055.1, siendo el 0 el valor que más se repite, tiene 133 posibles valores fuera de rango.
- “credito” es de tipo moneda, cuyos valores están entre -312583.05 y 0, siendo el 0 el valor que más se repite.

- La tabla “fac_detalleservicios” tiene un total de 12 campos y 2454 tuplas:
  - “cantidad” es de tipo moneda con cardinalidad 577 y sus valores están entre 1 y 396954.1, siendo el 40 el valor que más se repite, tiene 523 posibles valores fuera de rango.
  - “porcrecdesc” es de tipo moneda, con cardinalidad 2 y sus valores son -45 y 0, siendo el 0 el valor que más se repite, y ambos son, posibles valores, fuera de rango.
  - “imprecdescmn” es de tipo moneda, con cardinalidad 2 y sus valores son -2160 y 0, siendo el 0 el valor que más se repite, y ambos son, posibles valores, fuera de rango.
  - “imprecdescmlc” es de tipo moneda y el único valor que tiene es 0.
  - “preciomn” es de tipo moneda, con cardinalidad 694 y sus valores están entre 0 y 30000, siendo el 112.5 el valor que más se repite, tiene 681 posibles valores fuera de rango.
  - “preciomlc” es de tipo moneda, con cardinalidad 16 y sus valores están entre 0 y 5860, siendo el 0 el valor que más se repite, tiene 16 posibles valores fuera de rango (0, 500, 200, 4860, 3760, 2377, 150, 270, 2450, 1355, 1795, 1200, 1100, 3830, 2310, 5860).
- La tabla “fin_regdocum” tiene un total de 8 campos y 32 tuplas, y un campo de tipo no manejado por la herramienta.
- La tabla “fin_regdocumtalon” tiene un total de 3 campos y 5 tuplas, tiene un campo de tipo no manejado por la herramienta.
- La tabla “fin_tipocontrapartida” con un total de 7 campos y 16 tuplas, tiene 2 campos de tipo no manejado por la herramienta.
- La tabla “gen_subsistema” con un total de 3 campos y 6 tuplas, tiene un campo de tipo desconocido no manejado por la herramienta.

- La tabla “inv_regdocum” tiene un total de 6 campos, 24 tuplas y 2 campos de tipo no manejado por la herramienta.

En el análisis de esta Base de Datos se resumen los siguientes posibles problemas: mal diseño (tablas y campos innecesarios), valores repetidos, diferentes códigos con el mismo significado y valores ausentes.

La otra de las BD en SQL se llama ZafraNueva y en esta se guarda la información referente al corte y alza de caña, así como de las estadísticas de los diferentes centrales, esta Base de Datos tiene un total de 169 tablas, y un tamaño en disco de 664 MB. Su análisis nos brindó los siguientes resultados:

- Hay un total de 18 tablas vacías

Ahora vamos a proceder a poner las tablas con los campos que tengan posibles errores, de acuerdo con los resultados brindados por la herramienta:

- La tabla “decena” tiene 7 campos y 11424 tuplas:
  - “CAI_Codigo” es de tipo entero, con cardinalidad 14, valores entre 401 y 999, el 999 es el valor que más se repite y tiene un posible valor fuera de rango (999), esto nos indica que dicho valor este usado para representar una información que falta.
  - “Valor_Decena” es de tipo real, con cardinalidad 5705, valores entre -162 y 303706.2, el 499 es el valor que más se repite y tiene 2240 posibles valores fuera de rango.
  - “Valor_HF” es de tipo real, con cardinalidad 6878, valores entre 0 y 2402197.3, el valor que más se repite es el 1125 y tiene 4287 posibles valores fuera de rango.

    En los dos campos anteriores los resultados muestran que los valores están muy dispersos y puede ser consecuencia de errores.
- La tabla “Decena_Edad_Cepa” tiene 8 campos y 8320 tuplas:
  - “CCortada_Dec” es de tipo moneda, con cardinalidad 1394, valores entre 0 y 23109.22, el 0 es el valor que más se repite y tiene 1172 posibles valores fuera de rango.

- “CCortada_HF” es de tipo moneda, con cardinalidad 2045 y valores entre 0 y 152899.33, el 0 es el valor que más se repite y tiene 1867 posibles valores fuera de rango.

Se vuelve a observar en estos dos campos dispersión en los valores.

- La tabla “Decena_Estimado” tiene 23 campos y 640 tuplas:
  - “ARg_Dec” es de tipo moneda, con cardinalidad 154, valores entre 0 y 4939.6, el 0 es el valor que más se repite y tiene 138 posibles valores fuera de rango.
  - “ARg_Hf” es de tipo moneda, con cardinalidad 213, valores entre 0 y 4939.6, el 0 es el valor que más se repite y tiene 176 posibles valores fuera de rango.
  - “ERg_Dec” es de tipo moneda, con cardinalidad 153, valores entre 0 y 27523.18, el 0 es el valor que más se repite, y tiene 125 posibles valores fuera de rango.
  - “ERg_Hf” es de tipo moneda, con cardinalidad 207, valores entre 0 y 105737.83, el 0 es el valor que más se repite, y tiene 176 posibles valores fuera de rango.
  - “PRg_Dec” es de tipo moneda, con cardinalidad 153, valores entre 0 y 27726.61, el 0 es el valor que más se repite y tiene 125 posibles valores fuera de rango.
  - “PRg_Hf” es de tipo moneda, con cardinalidad 212, valores entre 0 y 106952.34, el 0 es el valor que más se repite, y tiene 179 posibles valores fuera de rango.
  - “ASc_Dec” es de tipo moneda, con cardinalidad 413, valores entre 0 y 698.56, el 3 es el valor que más se repite y tiene 222 posibles valores fuera de rango.
  - “ASc_Hf” es de tipo moneda, con cardinalidad 469, valores entre 0 y 3335.51, el valor que más se repite es 54.25 y tiene 292 posibles valores fuera de rango.
  - “ESc_Dec” es de tipo moneda, con cardinalidad 414, valores entre 0 y 16794.34, el 0 es el valor que más se repite, tiene 186 posibles valores fuera de rango.
  - “ESc_Hf” es de tipo moneda, con cardinalidad 471, valores entre 0 y 81262.86, el valor que más se repite es el 2137.25, tiene 249 posibles valores fuera de rango.
  - “PSc_Desc” es de tipo moneda, con cardinalidad 412, valores entre 0 y 20595.93, el valor que más se repite es el 90, tiene 192 posibles valores fuera de rango.
  - “PSc_Hf” es de tipo moneda, con cardinalidad 471, valores entre 0 y 85224.14, el valor que más se repite es el 90, tiene 239 posibles valores fuera de rango.
  - “ATtl_Dec” es de tipo moneda, con cardinalidad 443, valores entre 0 y 4939.6, el valor que más se repite es el 3, tiene 280 posibles valores fuera de rango.
  - “ATtl_Hf” es de tipo moneda, con cardinalidad 490, valores entre 0 y 4939.6, el valor que más se repite es 54.25 y tiene 277 posibles valores fuera de rango.

- “ETtl_Dec” es de tipo moneda, con cardinalidad 443, valores entre 0 y 29657.18, 0 es el valor que más se repite y tiene 164 posibles valores fuera de rango.
- “ETtl_Hf” es de tipo moneda, con cardinalidad 491, valores entre 0 y 118223.26, el valor que más se repite es el 2137.0 y tiene 263 posibles valores fuera de rango.
- “PTtl_Dec” es de tipo moneda, con cardinalidad 442, valores entre 0 y 29866.44, el valor que más se repite es el 90, tiene 171 posibles valores fuera de rango.
- “PTtl_Hf” es de tipo moneda, con cardinalidad 487, valores entre 0 y 119343.02, el valor que más se repite es el 5210 y tiene 253 posibles valores fuera de rango.

En varios de los campos anteriores se aprecia la aparición del 0, representado como valor por defecto y se sigue apreciando la dispersión.

- La tabla “Decena_Molida” tiene 7 campos y 167 tuplas:
  - “CMolida_Dec” es de tipo moneda, con cardinalidad 144 y valores entre 0 y 140453.15, 0 es el valor que más se repite y tiene 51 posibles valores fuera de rango.
  - “CMolida_Hf” es de tipo moneda, con cardinalidad 167, valores entre 5.85 y 284893.22, el valor que más se repite es el 4535.05 y tiene 65 posibles valores fuera de rango.
- La tabla “Decena_Variedad” tiene 7 campos y 2048 tuplas:
  - “CCortada_Dec” es de tipo moneda, con cardinalidad 819, valores entre 0 y 24458.49, el valor que más se repite es el 200 y tiene 621 posibles valores fuera de rango.
  - “CCortada_Hf” es de tipo moneda, con cardinalidad 953, valores entre 0 y 137854.55, el valor que más se repite es el 20345.75 y tiene posibles 731 valores fuera de rango.
- La tabla “Diario” tiene 6 campos y 249008 tuplas:
  - “CAI_Codigo” es de tipo entero, con cardinalidad 14, valores entre 401 y 999, el valor que más se repite es el 449 y tiene un posible valor fuera de rango (999).
  - “Valor_Dia” es de tipo real, con cardinalidad 161233, valores entre -12419.63 y 720487.351, el valor que más se repite es el 51.25 y tiene 161181 posibles valores fuera de rango.

- "Valor_HF" es de tipo real, con cardinalidad 181131, valores entre -35103.78 y 49973594.394, el valor que más se repite es el 112.125 y tiene 181092 posibles valores fuera de rango.
- "Valor_DecDia" es de tipo real, con cardinalidad 167580, valores entre -1078269.675 y 6269944.129, el valor que más se repite es el 19.375 y tiene 167532 posibles valores fuera de rango.

- La tabla "Diario_Ajuste" tiene 5 campos y 8353 tuplas:
  - "Valor_DecDia" es de tipo real, con cardinalidad 1091, valores entre -693.926 y 5774712, 0 es el valor que más se repite y todos los valores son posibles fuera de rango.
- La tabla "Diario_CalInfo" tiene 9 campos y 17136 tuplas:
  - "canthf" es de tipo real, con cardinalidad 204, valores entre 0 y 857, el valor que m´s se repite es el 101 y tiene 145 posibles valores fuera de rango.
  - "incuhf" es de tipo real, con cardinalidad 532, valores entre 0 y 757, 0 es el valor que más se repite y tiene 468 posibles valores fuera de rango.
- La tabla "Diario_CalInfoRdto" tiene 4 campos y 1224 tuplas:
  - "CAI_Codigo" es de tipo entero, con cardinalidad 14, valores entre 401 y 999, el 999 es el valor que más se repite y tiene un posible valor fuera de rango (999).
  - "rdto3dias" es de tipo real, con cardinalidad 1082, valores entre 0 y 17.913, 0 es el valore que más se repite y tiene 48 posibles valores fuera de rango.
  - "rdtorepo" es de tipo real, con cardinalidad 1025, valores entre 0 y 13.598, 0 es el valor que más se repite y tiene 5 posibles valores fuera de rango (13.301, 13.34, 13.304, 13.315, 13.598).
- La tabla "Diario_Energia" tiene 6 campos y 46102 tuplas:
  - "CAI_Codigo" es de tipo entero, con cardinalidad 14, valores entre 401 y 999, el 999 es el valor que más se repite y tiene un posible valor fuera de rango (999).
  - "Valor_Dia" es de tipo real, con cardinalidad 30835, valores entre -642 y 844677, 0 es el valor que más se repite y tiene 30317 posibles valores fuera de rango.
  - "Valor_HF" es de tipo real con cardinalidad 36191, valores entre 0 y 56122413, 0 es el valor que más se repite y tiene 35880 posibles valores fuera de rango.

- “Valor_DecDia” es de tipo real, con cardinalidad 33626, valores entre -642 y 7613565, 0 es el valor que más se repite y tiene 33269 posibles valores fuera de rango.

- La tabla “Diario_Equipo” tiene 26 campos y 20247 tuplas:
  - “ct_verde_da” es de tipo moneda, con cardinalidad 13276, valores entre 0 y 4603.75, el valor que más se repite es el 455.75 y tiene 7719 posibles valores fuera de rango.
  - “ct_quemada_da” es de tipo moneda, con cardinalidad 1019, valores entre 0 y 2222.39, 0 es el valor que más se repite y tiene 934 posibles valores fuera de rango.
  - “ct_total_da” es de tipo moneda, con cardinalidad 13426, valores entre 0 y 4603.75, el valor que más se repite es 455.75 y tiene 7836 posibles valores fuera de rango.
  - “proto_dd” es de tipo entero, con cardinalidad 65, valores entre 0 y 76, 0 es el valor que más se repite (17179 veces) y tiene 62 posibles valores fuera de rango.
  - “ftransporte_dd” es de tipo entero, con cardinalidad 69, valores entre 0 y 127, 0 es el valor que más se repite (18391 veces) y tiene 66 posibles valores fuera de rango.
  - “ct_verde_dd” es de tipo moneda, con cardinalidad 14927, valores entre 0 y 25400.86, el valor que más se repite es 985.75 y tiene 10 492 posibles valores fuera de rango.
  - “ct_quemada_dd” es de tipo moneda, con cardinalidad 1680, valores entre 0 y 8948.06, 0 es el valor que más se repite y tiene 1045 posibles valores fuera de rango.
  - “ct_total_dd” es de tipo moneda, con cardinalidad 14974, valores entre 0 y 25400.86, el valor que más se repite es 985.75 y tiene 10478 posibles valores fuera de rango.
  - “ftransporte_hf” es de tipo entero, con cardinalidad 265, valores entre 0 y 407, 0 es el valor que más se repite (16404 veces) y tiene 248 posibles valores fuera de rango.
  - “ct_verde_hf” es de tipo moneda, con cardinalidad 16289, valores entre 0 y 225054.35, el valor que más se repite es 53648 y tiene 12439 posibles valores fuera de rango.

- “ct_quemada_hf” es de tipo moneda, con cardinalidad 2737, valores entre 0 y 32743.61, 0 es el valor que más se repite y tiene 2488 posibles valores fuera de rango.
- “ct_total_hf” es de tipo moneda, con cardinalidad 16411, valores entre 0 y 225054.35, el valor que más se repite es 53648 y tiene 11708 posibles valores fuera de rango.

Se aprecia en los campos de tipo entero, la aparición del 0 como valor por defecto y se sigue viendo la dispersión en los campos de tipo moneda.

- La tabla “Diario_Fuerza” tiene 22 campos y 2200 tuplas:
  - “ccortada_da” es de tipo moneda, con cardinalidad 1587, valores entre 0 y 1635.87, el valor que más se repite es 35 y tiene 513 posibles valores fuera de rango.
  - “ct_verde_da” es de tipo moneda, con cardinalidad 1414, valores entre 0 y 1436.6, el valor que más se repite es 91.5 y tiene 638 posibles valores fuera de rango.
  - “ct_quemada_da” es de tipo moneda, con cardinalidad 989, valores entre 0 y 1430.87, 0 es el valor que más se repite y tiene 580 posibles valores fuera de rango.
  - “ct_total_da” es de tipo moneda, con cardinalidad 1718, valores entre 0 y 1542.27, el valor que más se repite es 34 y tiene 535 posibles valores fuera de rango.
  - “plantilla_dd” es de tipo entero, con cardinalidad 826, valores entre 0 y 4832, el valor que más se repite es786, tiene 330 elementos igual a 0 y 357 posibles valores fuera de rango.
  - “asistcorte_dd” es de tipo entero, con cardinalidad 932, valores entre 0 y 4423, el valor que más se repite es 63, tiene 343 elementos igual a cero y 408 posibles valores fuera de rango.
  - “ccortada_dd” es de tipo moneda, con cardinalidad 1811, valores entre 0 y 11786.12, el valor que más se repite es 155 y tiene 890 posibles valores fuera de rango.
  - “ct_verde_dd” es de tipo moneda, con cardinalidad 1606, valores entre 0 y 9409.02, el valor que más se repite es 6 y tiene 955 posibles valores fuera de rango.
  - “ct_quemada_dd” es de tipo moneda, con cardinalidad 1299, valores entre 0 y 8205, 0 es el valor que más se repite y tiene 963 posibles valores fuera de rango.

- “ct_total_dd” es de tipo moneda, con cardinalidad 1849, valores entre 0 y 113009.21, el valor que más se repite es 84.5 y tiene 924 posibles valores fuera de rango.
- “plantilla_hf” es de tipo entero, con cardinalidad 1646, valores entre 0 y 40596, el valor que más se repite es 23922, tiene 139 elementos igual a 0 y 827 posibles valores fuera de rango.
- “asistcorte_hf” es de tipo entero con cardinalidad 1586, valores entre 0 y 31291, el valor que más se repite es 1813, tiene 139 elementos igual a 0 y 795 posibles valores fuera de rango.
- “ccortada_hf” es de tipo moneda, con cardinalidad 2006, valores entre 0 y 80345.84, el valor que más se repite es 14326.75 y tiene 1080 posibles valores fuera de rango.
- “ct_verde_hf” es de tipo moneda, con cardinalidad 1860, valores entre 0 y 60668.73, el valor que más se repite es1860 y tiene 862 posibles valores fuera de rango.
- “ct_quemada_hf” es de tipo moneda, con cardinalidad 1534, valores entre 0 y 36016.79, 0 es el elemento que más se repite y tiene 1256 posibles valores fuera de rango.
- “ct_total_hf” es de tipo moneda, con cardinalidad 1964, valores entre 0 y 78698.63, el valor que más se repite es 13851 y tiene 1058 posibles valores fuera de rango.

- La tabla “Diario_MPGen” tiene 6 campos y 98358 tuplas:
  - “CAI_Codigo” es de tipo entero, con cardinalidad 14, valores entre 401 y 999, el 999 es el valor que más se repite y tiene un posible valor fuera de rango (999).
  - “Valor_Dia” es de tipo real, con cardinalidad 63511, valores entre -82.814 y 55748.35, 0 es el valor que más se repite y tiene 62135 posibles valores fuera de rango.
  - “Valor_HF” es de tipo real, con cardinalidad 79574, valores entre 0 y 2517402.52, 0 es el valor que más se repite y tiene 78864 posibles valores fuera de rango.
  - “Valor_DecDia” es de tipo real, con cardinalidad 72346, valores entre 0 y 313675.49, 0 es el valor que más se repite y tiene 71316 posibles valores fuera de rango.
- La tabla “Diario_Parametro” tiene 8 campos y 60711 tuplas:

- "CAI_Codigo" es de tipo entero, con cardinalidad 14, valores entre 401 y 999, el 999 es el valor que más se repite y tiene un posible valor fuera de rango (999).
- "Valor_Dia" es de tipo real, con cardinalidad 36309, valores entre -4 y 3009.419, el valor que más se repite es 389.5 y tiene 19887 posibles valores fuera de rango.
- "Valor_DecDia" es de tipo real, con cardinalidad 41669, valores entre -1.632 y 23737.472, el valor que más se repite es 523 y tiene 31716 posibles valores fuera de rango.
- "Valor_HF" es de tipo real, con cardinalidad 49277, valores entre -1.026 y 178676.399, el valor que más se repite es 49277 y tiene 12860 posibles valores fuera de rango.

El análisis de esta Base de Datos nos brindó los siguientes resultados: posible mal diseño de la base de datos debido a tablas vacías, presencia de valores por defecto que pueden ser considerados valores faltantes en muchas tuplas, mucha dispersión en los valores de los campos de tipo moneda y real.

### 3.3.2 Access

La base de datos Estipendio tiene un total de 19 tablas

- La tabla "Conversion Errors" tiene 3 campos y 1 tupla, 2 de sus campos están vacíos, uno de ellos es de tipo cadena y el otro es de un tipo no manejado por la herramienta.
- La tabla "Errores de pegado" tiene1 campo y 131 tuplas, el campos es de tipo cadena, tiene dos posibles errores, uno de ellos es que mantiene una tupla con la cadena '#Eliminado' de un elemento que ya fue borrado, y el otro es el numero de un carné, donde la cadena que aparece es '00000000100'.
- La tabla "Estipendiosanoanterior" tiene 2 campos y 3769 tuplas:
  - "CI" es de tipo cadena, con cardinalidad 3769, y este campo tiene 33 números de identidad representados de forma incorrecta (10000000000, 11000000000, 20000000000, 30000000000, 40000000000, 50000000000, 70000000000, 73121200000, 75010200000, 75051500000, 75081400000, 76070600000, 77010300000, 77061900000, 77070700000, 77090700000, 77102000000, 78012200000, 78012700000, 78052500000, 78070600000, 78092600000, 79021200000, 79071400000, 79101000000, 80000000000, 80063000000,

80081400000, 80112300000, 81062400000, 81110600000, 90000000000, 99030812337).

- La tabla “estudiantesanoanterior” tiene 11 campos y 3776 tuplas:
  - “CI” es de tipo cadena, con cardinalidad 3776, se encontraron 38 números de identidad representados de forma incorrecta (10000000000, 11000000000, 20000000000, 30000000000, 40000000000, 50000000000, 68122700000, 70000000000, 71021200000, 72091800000, 73061700000, 73080800000, 73121200000, 75010200000, 75051500000, 75081400000, 76070600000, 77010300000, 77070700000, 77090700000, 77102000000, 78012200000, 78012700000, 78052500000, 78070600000, 79021200000, 79021200000, 79101000000, 80000000000, 80031300000, 80050500000, 80063000000, 80081400000, 80112300000, 81062400000, 81110600000, 90000000000, 99030812337).
  - “Obsv” es de tipo no manejado por el sistema.
  - “Foto” es de tipo no manejado por el sistema.
- “Nomina” tiene 6 campos y 4260 tuplas:
  - CI” es de tipo cadena, con cardinalidad 4260, se encontraron 26 números de identidad representados de forma incorrecta (00000000001, 00000000006, 00000000008, 00000000009, 00000000011, 00000000017, 00000000019, 00000000023, 00000000024, 00000000025, 00000000031, 00000000035, 00000000037, 00000000039, 00000000041, 00000000043, 00000000044, 00000000045, 00000000049, 00000000051, 00000000052, 00000000061, 00000000064, 00000000065, 00000000090, 82062700000).
- La tabla “Nominasanoanterior” tiene 6 campos y 3776 tuplas:
  - “CI” es de tipo cadena, con cardinalidad 3776, se encontraron 35 números de identidad representados de forma incorrecta (10000000000, 11000000000, 20000000000, 30000000000, 40000000000, 50000000000, 68122700000, 68122700000, 71021200000, 73080800000, 75010200000, 75051500000, 75081400000, 76070600000, 77010300000, 77061900000, 77070700000, 77102000000, 78012200000, 78012700000, 78052500000, 78070600000, 78092600000, 79021200000, 79071400000, 79101000000, 80000000000,

80031300000, 80050500000, 80063000000, 80081400000, 80112300000, 81062400000, 90000000000, 99030812337).

- La tabla “TEstNuevos” tiene 11 campos y 3244 tuplas:
  - “CI” es de tipo cadena, con cardinalidad 3244, se encontraron 38 números de identidad representados de forma incorrecta (10000000000, 11000000000, 20000000000, 30000000000, 40000000000, 50000000000, 68122700000, 70000000000, 71021200000, 73080800000, 73061700000, 73080800000, 73121200000, 75010200000, 75051500000, 75081400000, 76070600000, 77010300000, 77061900000, 77070700000, 77090700000, 77102000000, 78012200000, 78012700000, 78052500000, 78070600000, 78092600000, 79021200000, 79101000000, 80000000000, 80031300000, 80050500000, 80063000000, 80081400000, 80112300000, 81110600000, 90000000000, 99030812337).
  - “Foto” es de tipo no manipulado por la herramienta.
- La tabla “TEstudiantes” tiene 11 campos y 4260 tuplas:
  - CI” es de tipo cadena, con cardinalidad 4260, se encontraron 25 números de identidad representados de forma incorrecta (00000000001, 00000000006, 00000000008, 00000000009, 00000000011, 00000000017, 00000000019, 00000000023, 00000000024, 00000000025, 00000000031, 00000000035, 00000000037, 00000000039, 00000000041, 00000000043, 00000000044, 00000000045, 00000000049, 00000000051, 00000000052, 00000000061, 00000000064, 00000000065, 00000000090).
  - “Nombre” es de tipo cadena, con cardinalidad 4258, a continuación se muestran los nombres repetidos:

| Cadena | Veces |
|---|---|
| PEREZ RODRIGUEZ MICHAEL | 2 |
| GONZALEZ GARCIA ANISLEY | 2 |

  - “Obsv” es de tipo no analizado por la herramienta.
  - “Foto” es de tipo no analizado por la herramienta.
- La tabla “TEstudiantesOld” tiene 11 campos y 4081 tuplas:

- CI” es de tipo cadena, con cardinalidad 4081, se encontraron 39 números de identidad representados de forma incorrecta (10000000000, 11000000000, 20000000000, 30000000000, 40000000000, 50000000000, 68122700000, 70000000000, 71021200000, 72091800000, 73061700000, 73080800000, 73121200000, 75010200000, 75051500000, 75081400000, 76070600000, 77010300000, 77061900000, 77070700000, 77090700000, 77102000000, 78012200000, 78012700000, 78052500000, 78070600000, 78092600000, 79021200000, 79071400000, 79101000000, 80000000000, 80031300000, 80063000000, 80081400000, 80112300000, 81062400000, 81110600000, 90000000000, 99030812337).
- “Obsv” es de tipo no manejado por la herramienta.
- “Foto” es de tipo no manejado por la herramienta.

El análisis realizado a esta base de datos nos brindó los siguientes resultados: repetición de nombres, números de identidad incorrectos (teniendo en cuenta el significado de cada uno de sus dígitos) y una cadena vacía (borrada).

### 3.3.3 FoxPro

La base de datos defunc99 tiene un total de 2 tablas.

- La tabla “cie9” tiene 2 campos y 5314 tuplas:
  - “dicc_codig” es de tipo cadena, con cardinalidad 5309, para este campo la herramienta detectó 362 posibles errores, debido a la ocurrencia de la letra “X” en lugar de un dígito que sería lo correcto ya que los valores de este campo se componen por cuatro dígitos, así como 5 casos en que aparece repetido dos veces el campo identificador lo cual significa que se enumeraron dos enfermedades con el mismo identificador, siendo los mismos los que a continuación relacionamos:

| **Código** | **Descripción 1** | **Descripción 2** |
|---|---|---|
| 2391 | “Tumor de naturaleza no especificada del aparato respiratorio” | Neoplasia pulmón |
| 2981 | “Otras psicosis no orgánicas, tipo agitado” | “Psicosis tipo agitado” |
| 3019 | “Transtornos de la personalidad, | “Transtornos de la |

| | | |
|---|---|---|
| | sin especificación” | personalidad no especificado” |
| 2390 | “Tumor de naturaleza no especificada del aparato digestivo” | “Neoplasia base lengua” |
| 7907 | “Bacteriemia no especificada” | “Bacteriemia por hemodialisis” |

- “dicc_desc” es de tipo cadena, con cardinalidad 5279, para este campo se encontraron 33 descripciones que están relacionadas a dos identificadores diferentes y 1 descripción a 3 identificadores distintos, a continuación se relaciona la lista:

| **Descripción** | **Códigos** |
|---|---|
| “Laceracion y contusion cerebrales” | 8510 y 851X |
| “Politraumatismo” | 8691 y 8692 |
| “Mola hidatiforme” | 630X y 6300 |
| “Envenenamiento por agentes psicotropicos” | 9699 y 969X |
| “Efectos de otras causas externas” | 994X y 9949 |
| “Traumatismo de otros organos intraabdominales” | 868X y 8680 |
| “Desarreglo interno de la rodilla” | 717X y 7179 |
| “Otros desarreglos articulares” | 718X y 7189 |
| “Otros trastornos articulares y el no especificado” | 719X y 7198 |
| “Deformidades adquiridas de los dedos del pie” | 735X y 7359 |
| “Otras deformidades adquiridas de los miembros” | 736X y 7369 |
| ”Otros hallazgos anormales no especificos” | 796X y 7969 |
| “Fractura base craneo” | 801X y 8019 |
| “Fractura de la rotula” | 822X y 8229 |
| “Esguinces y desgarros del hombro y del brazo” | 840X y 8409 |
| “Esguinces y desgarros del codo y del antebrazo” | 841X y 8419 |
| “Esguinces y desgarros de la cadera y del muslo” | 843X y 8439 |
| “Esguinces y desgarros de la rodilla y de la pierna” | 844X y 8449 |
| “Esguinces y desgarros de la region sacroiliaca” | 846X y 8469 |

| | |
|---|---|
| “Herida de los anexos del ojo” | 870X y 8709 |
| “Traumatismo superficial del ojo y sus anexos” | 918X y 9180 |
| “Quemaduras” | 949X y 9499 |
| “Envenenamiento por antibioticos” | 960X y 9609 |
| “Envenenamiento por sedantes e hipnoticos” | 967X y 9679 |
| “Efecto toxico del alcohol” | 980X y 9809 |
| “Efectos de la radiacion” | 990X y 9909 |
| “Efectos del frio” | 9910 y 9919 |
| “Efectos de la presion atmosferica” | 993X y 9939 |
| “Otras deficiencias nutricionales” | 2698 y 2699 |
| “Otras anomalias congenitas del corazon” | 7468 y 7469 |
| ”Fractura del tobillo,” | 8248 y 8249 |
| “Amputacion traumatica unilateral,nivel no especificado,sin complicacio” | 8874 y 8974 |
| “Efecto toxico de otros gases emanaciones o vapores” | 9878 y 9879 |
| “Efecto toxico de otros metales” | 9859, 985X y 9858 |

- La tabla “defu99” tiene 32 campos y 79499 tuplas:
  - Tiene 10 campos vacíos, todos de tipo cadena.
  - Tiene 11 campos de tipo real con todos sus elementos igual a 0.
  - “nombre” es de tipo cadena, con cardinalidad 13951, hay una serie de nombres que se repiten, los cuales son:

| **Cadena** | **Veces** |
|---|---|
| MARIA HDEZ GARCIA | 2 |
| JOSEFA ROMERO MTNEZ | 2 |
| CLARA DIAZ PEREZ | 2 |
| ELPIDIO REYES TAMAYO | 2 |
| LORENTE SOCARRAS ROBERTO | 2 |
| LUIS RGUEZ RGUEZ | 2 |
| GREGORIO E.PEREZ REGALADO | 2 |

| | |
|---|---|
| ALEIDA RDGUEZ RDGUEZ | 2 |
| DESCONOCIDO | 4 |

- “sexo” es definido aquí como un real y los valores que este campo toma son 1 y 2, cosa que es errónea, puesto que este campo debiera ser de tipo cadena, donde los valores fueran o bien 1 y 2 ó M y F, para que después se pudieran hacer correctamente los cálculos estadísticos correspondientes, y no como en este caso que al ser tratado como real, lo que devuelve es el valor máximo, el mínimo y la media cuando lo que realmente nos interesa es la cantidad de veces que se repite cada valor, y el valor en sí.
- “peso” que es de tipo real el valor máximo es de 9999 y el mínimo de 0, la moda es 0, lo que quiere decir que ese valor es el que más se repite, indicando que se está utilizando el 0 para especificar un valor que falta.
- “fecpan” es de tipo cadena, con cardinalidad 31, tiene 13876 elementos vacíos.
- “embar” es de tipo cadena, con cardinalidad 26, tiene 13876 elementos vacíos.
- “ccir” es de tipo cadena, con cardinalidad 17, tiene 13876 elementos vacíos.

De este análisis podemos concluir que como principales problemas en esta BD están:

- Violación de la regla de la Entidad.
- Problemas de diseño, que generan una gran cantidad de campos vacíos.
- Tipos de datos incorrectos.
- Datos fuera de rango.

La base de datos defunc03 tiene un total de 2 tablas:

- La tabla “cie10” tiene 2 campos y 12428 tuplas:
  - “descrip” es de tipo cadena, con cardinalidad 12426, este campo presentó la repetición de 2 descripciones, las cuales mostramos:

| **Descripción** | **Códigos** |
|---|---|
| “Sífilis cardiovascular” | A520 y I980 |
| “Asfixia” | R090 y T71 |

- La tabla “defu03” tiene 56 campos y 78434 tuplas:

- “apellido1” es de tipo cadena, con cardinalidad 2967, en este campo podemos apreciar falta de unicidad a la hora de representar el mismo apellido:

| **Apellido** | **Veces** |
|---|---|
| “Gonzalez” | 2522 |
| “Glez” | 135 |
| “Hernandez” | 2013 |
| “Hdez” | 109 |
| “Martinez” | 1436 |
| “Mtnez” | 116 |
| “Rodriguez” | 3004 |
| “Rdguez” | 157 |
| “Rguez” | 47 |
| “Dominguez” | 368 |
| “Dguez” | 4 |
| “Fernandez” | 927 |
| “Fdez” | 65 |

Así como errores ortográficos, a continuación los listamos:

| **Aparece** | **Veces** | **Debiera ser** |
|---|---|---|
| Hernnandez | 3 | Hernandez |
| Hernanadez | 3 | Hernandez |
| Ernandez | 2 | Hernandez |
| Hernande | 1 | Hernandez |
| Hernanez | 1 | Hernandez |
| Hernandze | 1 | Hernandez |
| Hernandezq | 1 | Hernandez |
| Henandez | 1 | Hernandez |
| Heernandez | 1 | Hernandez |
| Fernadez | 5 | Fernandez |
| Feernandez | 4 | Fernandez |
| Gozalez | 2 | Gonzalez |
| Glonzalez | 1 | Gonzalez |
| Gionzalez | 1 | Gonzalez |

| | | |
|---|---|---|
| Gez | 1 | Glez (Gonzalez) |
| Rodriguiez | 1 | Rodriguez |
| Lopezq | 1 | Lopez |
| Garcias | 19 | Garcia |
| Grcia | 3 | Garcia |
| Qcosta | 1 | Costa o Acosta |
| Cahapman | 1 | Chapman |
| Aabreu | 1 | Abreu |
| Aalfonso | 2 | Alfonso |
| Mmendez | 2 | Mendez |
| Menndez | 1 | Mendez |
| Perz | 2 | Perez |
| Peres | 2 | Perez |
| Perez< | 2 | Perez |
| Peerz | 1 | Perez |
| Riverro | 1 | Rivero |
| Villarq | 1 | Villar |
| Moralesn | 1 | Morales |
| Barrueto | 4 | Barreto |
| Caralero | 1 | Carralero |
| Cerralero | 1 | Carralero |
| Beltron | 1 | Beltran |
| Ssierra | 2 | Sierra |
| Escabar | 2 | Escobar |
| Ramorez | 1 | Ramirez |
| Herreria | 1 | Herrera |
| Batitista | 1 | Batista |

- “apellido2” es de tipo cadena, con cardinalidad 2750, en este campo podemos apreciar falta de unicidad a la hora de representar el mismo apellido:

| **Apellido** | **Veces** |
|---|---|
| “Gonzalez” | 2447 |

| | |
|---|---|
| “Glez” | 152 |
| “Hernandez” | 2118 |
| “Hdez” | 112 |
| “Martinez” | 1358 |
| “Mtnez” | 102 |
| “Rodriguez” | 2966 |
| “Rdguez” | 149 |
| “Rguez” | 47 |
| “Dominguez” | 307 |
| “Dguez” | 9 |
| “Fernandez” | 799 |
| “Fdez” | 57 |
| “Izquierdo” | 109 |
| “Izqdo” | 5 |

Así como posibles errores ortográficos:

| **Aparece** | **Veces** | **Debiera ser** |
|---|---|---|
| Hernadez | 12 | Hernandez |
| Hernnadez | 4 | Hernandez |
| Herandez | 3 | Hernandez |
| Heernandez | 3 | Hernandez |
| Hrnandez | 1 | Hernandez |
| Herhandez | 1 | Hernandez |
| Rodeiguez | 2 | Rodriguez |
| Rodrguez | 1 | Rodriguez |
| Rodrigez | 1 | Rodriguez |
| Peres | 3 | Perez |
| Peez | 1 | Perez |
| Frenandez | 4 | Fernandez |
| Gozalez | 4 | Gonzalez |
| Gonzelez | 1 | Gonzalez |
| Gonzaalez | 1 | Gonzalez |

| | | |
|---|---|---|
| Garcias | 13 | Garcia |
| Gacia | 1 | Garcia |
| Mertinez | 3 | Martinez |
| Maratinez | 1 | Martinez |
| Dminguez | 2 | Dominguez |
| Jiunco | 1 | Junco |
| Quindelan | 4 | Kindelan |
| Arsuaga | 1 | Arzuaga |
| Vilavicencio | 1 | Villavicencio |

- “noident” es de tipo cadena, con cardinalidad 12880, aquí aparecen repetidos una serie de números de identidad:

| **Cadena** | **Veces** |
|---|---|
| 72999999999 | 3 |
| 68999999999 | 3 |
| 50999999999 | 3 |
| 34999999999 | 3 |
| 64999999999 | 2 |
| 76011399999 | 2 |
| 34091899999 | 2 |
| 20021999999 | 2 |
| 52129999999 | 2 |
| 52099999999 | 2 |
| 22122699999 | 2 |
| 40101799999 | 2 |
| 59999999999 | 2 |
| 02112999999 | 2 |
| 20999999999 | 2 |
| 27999999999 | 2 |
| 39030999999 | 2 |
| 03011199999 | 2 |

También por el significado de los dígitos que componen el número de identidad de cada persona (los seis primeros dígitos significan la fecha de nacimiento de la persona, así los dos primeros dígitos significan el, el tercer y cuarto dígito el mes y el quinto y el sexto es el día) encontramos 52 números de identidad incorrectos (72999999999, 68999999999, 49999999999, 42099999999, 65999999999, 30999999999, 50999999999, 64999999999, 34999999999, 02019999999, 01999999999, 57119999999, 45129999999, 83129999999, 22129999999, 60099999999, 90129999999, 67119999999, 66099999999, 32119999999, 52129999999, 52099999999, 12099999999, 42999999999, 24999999999, 59999999999, 73999999999, 55999999999, 51999999999, 32999999999, 19999999999, 12999999999, 20999999999, 27999999999, 33999999999, 21999999999, 23999999999, 09999999999, 29999999999, 35999999999, 71999999999, 37999999999, 33019999999, 89099999999, 75099999999, 42129999999, 47129999999, 16129999999, 46999999999, 44999999999, 43999999999, 03019999999).

- "sexo" es de tipo real y los valores que este campo toma son 1 y 2, cosa que es errónea, puesto que este campo debiera ser de tipo cadena, donde los valores fueran o bien 1y 2 ó M y F, para que después se pudieran hacer correctamente los cálculos estadísticos correspondientes, y no como en este caso que al ser tratado como real, lo que devuelve es el valor máximo, el mínimo y la media cuando lo que realmente nos interesa es la cantidad de veces que se repite cada valor, y el valor en sí.
- "rmadre" es de tipo cadena, con cardinalidad 1, tiene 15 elementos vacíos, y 77759 elementos con el valor "0".
- "causa2" es de tipo cadena, con cardinalidad 128, tiene 11917 elementos vacíos.
- "causa3" es de tipo cadena, con cardinalidad 15, tiene 12748 elementos vacíos.
- "causa4" es de tipo cadena, con cardinalidad 15, tiene 10671 elementos vacíos.
- "causa5" es de tipo cadena, con cardinalidad 140, tiene 7619 elementos vacíos
- "causa6" es de tipo cadena, con cardinalidad 92, tiene 12066 elementos vacíos.
- "causa7" es de tipo cadena, con cardinalidad 43, tiene 12793 elementos vacíos.
- "causa8" es de tipo cadena, con cardinalidad 12, tiene 12884 elementos vacíos.
- "causa9" es de tipo cadena, con cardinalidad 2, tiene 12896 elementos vacíos.

- “peso” es de tipo real, donde el valor máximo es de 9999, y el mínimo es 0, la media es 19.56368 y la moda es 0, en este campo las mediciones están tomadas de forma incorrecta, pues no hay peso corporal que llegue al valor de 9999, ni nadie tiene como peso mínimo 0, si además añadimos que ese es el valor que más se repite.

El análisis a esta base de datos nos brinda como resultados, que la misma tiene un mal diseño, al tener tantos campos vacíos y con sus elementos igual a 0 (que también nos indica que está vacío, implica valores por defecto), así como inconsistencia a la hora de representar información y faltas de ortografía.

## 3.4 Taxonomía de errores de los datos en nuestro entorno.

A continuación presentamos un resumen que permite definir una primera versión que resume los errores más frecuentes en los datos en los sistemas de bases de datos operacionales en nuestro entorno.

Incompleto

- Registros que faltan
- Campos que faltan

Incorrecto

- Códigos incorrectos
- Registros duplicados
- Información incorrecta entrada al sistema

Incomprensible

- Múltiples campos dentro de uno
- Códigos desconocidos

Inconsistente

- Uso y significado inconsistente de diversos códigos
- Diferentes códigos con el mismo significado
- Uso inconsistente de nulos, vacíos y espacios
- Falta de integridad referencial

# Conclusiones

El proceso de limpieza de datos es aplicado con intensiones diferentes y dentro de áreas diferentes de la integración de datos y procesos de manipulación. Está definido como la secuencia de operaciones que tienen la intención de mejorar al máximo la calidad de datos de una colección de datos.

Tiene un inconveniente, y es que este proceso es muy dependiente del dominio y es exploratorio. Los software existentes de limpieza de datos sobre todo se centran en la transformación de datos y la eliminación de duplicados. Algunos de estos permiten la especificación declarativa de un proceso de limpieza más completo, dejando todavía la mayor parte de los detalles de la operación limpiadora al usuario.

Todavía existen muchos problemas abiertos y desafíos en la limpieza de datos, principalmente dirigidos en la dirección de la manipulación de valores múltiples, alternativos, y la documentación de operaciones limpiadoras realizadas. Así como la especificación y desarrollo de un framework apropiado que apoye el proceso de limpieza de datos.

El trabajo realizado en el marco de esta tesis cumplió los objetivos propuestos, se desarrolló una herramienta de ayuda al análisis de los datos, primera y fundamental etapa para lograr la detección de errores, y en consecuencia del estudio realizado con las Bases de datos analizadas, pudimos establecer una Taxonomía inicial de los errores de los datos en nuestro entorno.

## Recomendaciones

Las recomendaciones fundamentales que planteamos se tengan en cuenta para darle continuidad al trabajo comenzado en este proyecto de tesis están orientadas a:

- Adicionar otros cálculos estadísticos (unicidad, varianza y mediana) en el análisis de los valores numéricos que ayuden a la agilización del proceso de limpieza y a una mejor comprensión de los datos.
- Analizar un conjunto mayor de bases de datos que permitan enriquecer la taxonomía propuesta.
- Usar hilos en la programación de la herramienta, para mejorar el control sobre la aplicación, esto se traduce en: poder continuar procesando los diferentes mensajes que lleguen a la aplicación aún cuando se estén analizando bases de datos muy grandes, con lo que se ganaría más control sobre la aplicación, permitiendo detener, reiniciar, y salvar el proceso en momentos que el usuario desee.
- Analizar la posibilidad de la migración de esta herramienta y de otras que se desarrollen en el entorno de esta investigación a plataformas de software libre.

# Referencias Bibliográficas


[1] Redman TC. "The Impact of Poor Data Quality on the Typical Enterprise". Communications of the ACM, 1998. pp. 79-82.

[2] Jonathan I. Maletic AM. Automated Identification of Errors in Data Sets. Memphis: The University of Memphis, 2002.

[3] H. Galhardas DF, D. Shasha, E. Simon. “An Extensible Framework for Data Cleaning". France: Institute National de Recherche en Informatique ét en Automatique,, 1999.

[4] M. A. Hernandez JSS. “Real-world Data is Dirty: Data Cleansing and The Merge/Purge Problem”, Journal of Data Mining and Knowledge Discovery 1998(2):9-37.

[5] A. Levitin TR. "A Model of the data (life) cycles with application to quality". Information and Software Technology, 1995. pp. 217-223.

[6] U. Fayyad GP-S, P. Smyth. “From Data Mining to Knowledge Discovery: An Overview”, Advances in Knowledge Discovery and Data Mining 1996:1-36.

[7] Chapman AD. Principles of Data Quality. Report for the Global Biodiversity Information Facility. Copenhagen: GBIF, 2004.

[8] Redman TC. Data Quality: The Field Guide. Boston: MA: Digital Press. Rios, 2001.

[9] Heiko Müller J-CF. Problems, Methods, and Challenges in Comprehensive Data Cleansing. Berlin, Germany, 2003.

[10] Quass D. A Framework for Research in Data Cleaning. Brinham Young University, 1999.

[11] Greenfield L. An (Informal) Taxonomy of Data Warehouse Data Errors. 2000.

[12] A.V. Aho JDU. Principles of Compiler Design. Addison-Wesley Publishing Company, 1979.

[13] V. Raman JMH. Potter’s Wheel: An Interactive Framework for Data Transformation and Cleaning., Proceedings of the 27th VLDB Conference,2001, Roma, Italy.

[14] S. Abiteboul SC, T. Milo, P. Mogilevsky, J. Siméon, S. Zohar. Tools for Data Translation and Integration., Bulletin of the Technical Committee on Data Engineering 1999;22(1):3-8.

[15] K. U. Sattler ES. A Data Preparation Framework based on a Multidatabase Language., International Database Engineering Applications Symposium (IDEAS),2001, Grenoble, France.

[16] Enric Mayol ET. A Survey of Current Methods for Integrity Constraint Maintenance and View Updating., ER Workshops,1999, 62-73.

[17] M.A. Hernandez SJS. The merge/purge problem for large databases., Proceedings of the ACM SIGMOD Conference,1995.

[18] J.I. Maletic AM. Data Cleansing: Beyond Integrity Analysis., Proceedings of the Conference on Information Quality,2000.

[19] H. Galhardas DF, D. Shasha, E. Simon. AJAX: An extensible data cleaning tool., Proceedings of the ACM SIGMOD on Management of data,2000, Dallas, TX, USA.

[20] H. Galhardas DF, D. Shasha, E. Simon, C.-A. Saita. Declarative data cleaning: Language, model, and algorithms., Proceedings of the 27th VLDB Conference,2001, Roma, Italy.

[21] K. U. Sattler SC, G. Saake. Adding Conflict Resolution Features to a Query Language for Database Federations., Proc. 3nd Int. Workshop on Engineering Federated Information Systems,2000, Dublin, Ireland.

[22] P. Vassiliadis ZaV, S. Skiadopoulos, N. Karayannidis, T. Sellis. ARKTOS: towards the modeling, design, control and execution of ETL processes. Information Systems, 2001. pp. 537-561.

[23] Mong Li Lee TWL, Wai Lup Low. IntelliClean: A knowledge-based intelligent data cleaner., Proceedings of the ACM SIGKDD,2000, Boston, USA.

[24] Wai Lup Low MLLaTWL. A knowledge-based approach for duplicate limination in data cleaning. Information Systems, 2001. pp. 585-606.