Universidad de las Ciencias Informáticas

Facultad de Software Libre

Trabajo de diploma para optar por el título de Ingeniero Informático

Título: DBAnalyzer 2.0, sistema para analizar bases de datos libres.

Autor(a): Yenisel Valdés Hernández

Tutor: Lic. Daynel Marmol Lacal

Ciudad de la Habana, Junio de 2008

"Año 50 de la Revolución"

“Quality is free. It's not a gift, but it is free. What costs money are the unquality things, all the actions that involve not doing jobs right the first time.

Quality is not only free; it is an honest-to-everything profit maker. Every penny you don't spend on doing things wrong, over, or instead, becomes half a penny right on the bottom line. If you concentrate on making quality certain you can probably increase your profit by an amount equal to 5 to 10 percent of your sales. That is a lot of money for free.”

Philip B. Crosby, Quality Is Free, New York: Penguin Group, 1979, p. 1.

Agradecimientos

A mis padres por haber confiado siempre en mí, por haberme guiado en la vida y por haberme dado todo lo que un hijo pudiera desear, su amor y su cariño.

A mi hermano, porque deseo que sea tan o más exitoso que yo.

A mi hermana, porque se que ella me apoya siempre.

A mi familia, por mantenerse siempre unida, y ser ejemplo para mí de firmeza, de unidad y de amor.

A Daynel, que con paciencia y amor, ha sabido alentarme a seguir adelante y me ha apoyado en todas mis decisiones.

A todos mis amigos, a Zulima, a Arelis, a Leticia, a Dunia, en fin a todos aquellos que en mi vida de estudiante universitaria estuvieron ahí para mí.

Dedicatoria

Este trabajo va especialmente dedicado a mi mamá y a mi papá porque son los mejores papás del mundo.

A mi hermanito, porque como él, ninguno.

A Daynel, porque sin su ayuda me hubiera resultado mucho mas difícil.

Resumen

Bien es conocida la necesidad que existe hoy en día de tener acceso a datos confiables que aporten una buena calidad al desarrollo de las operaciones en las empresas y a la toma de decisiones, por lo que es de vital importancia contar con herramientas que ayuden en este proceso tan complicado que es la limpieza de datos.

Como resultado de una intensa investigación sobre el tema surge este trabajo, el cual se basa principalmente en el análisis estadístico de la información de las Bases de Datos (BD). También es descrito detalladamente el proceso de investigación, análisis y diseño de una herramienta que ayude en el proceso de limpieza de datos.

Con el nombre de DBAnalyzer 2.0 se presenta una herramienta capaz de analizar bases de datos sobre gestores libres, en este caso, PostgreSQL y MySQL. Sucesora de otra realizada en software propietario, brindará información sobre los datos contenidos en las tablas de las BD agrupándola por campos, de los cuales según su tipo se brindará determinada información, con el objetivo de que los analistas de este tipo de sistemas puedan identificar los errores fácilmente y así poder corregirlos con mayor rapidez.

Índice

Introducción

Desde un inicio, las bases de datos se convirtieron en una herramienta fundamental de control y manejo de información. Sin embargo no se tomaba en cuenta que el volumen de la misma iba en aumento y que más tarde sería imposible tener el control y hacer un uso eficiente de la información histórica almacenada en una base de datos, para la ayuda a la toma de decisiones; por lo que se hizo de vital importancia garantizar que estos datos fueran fáciles de obtener, estandarizados y confiables. En la actualidad las empresas generalmente no cuentan con aplicaciones únicas para cada parte operativa del negocio, sino que pueden tener replicaciones y distintos sistemas para atender un mismo conjunto de operaciones; por lo que un tema reconocido universalmente en Bases de Datos, pero la mayoría de las veces ignorado, es la limpieza de datos. De ahí que se hayan identificado tres temas fundamentales (PADRON 2006) como los mayores problemas en el manejo de los datos, estos son: el acceso a datos; las herramientas de consulta y por último la integridad de los datos. De estos problemas solo se atacan con fuerza los dos primeros, mientras que el tercero es un tema muchas veces ignorado.

El problema de la integridad o la limpieza de datos es poco tratado o evadido por muchas empresas, lo que conlleva a no considerar adecuadamente el impacto negativo para el negocio de tener almacenada información deficiente. En esos casos es probable que las bases de datos de los sistemas operacionales contengan datos duplicados, a veces erróneos, superfluos o incompletos; y a esto se le suman los posibles errores a la hora de la entrada de datos a los sistemas de datos operacionales.

La existencia de “datos sucios” como también se le suele llamar a los errores en las bases de datos suele acarrear grandes dificultades en las instituciones que no se preocupan por la confiabilidad de sus datos, lo que se refleja en el alto costo operacional, la toma de decisiones inadecuadas, el incremento de la inseguridad y una desviación de la atención de las direcciones de las instituciones (MALETIC and MARCUS 2002).

La **situación problémica** que se presenta es la siguiente:

En la actualidad hay diferentes aplicaciones que tienen como objetivo la limpieza de datos. Un problema de estas es el dominio de los datos, debido a que están implementadas para usarse en determinado contexto por lo que no pueden ser utilizadas para “limpiar” cualquier tipo de información, a ello se suma la dificultad que trae el que estas herramientas estén implementadas sobre software propietario. En Cuba todavía no se ha desarrollado ningún trabajo de este tipo, o no hay mucha información al respecto, aunque existe una herramienta llamada DBAnalyzer (MARMOL 2005), que fue creada con la intención de clasificar errores en las bases de datos. Esta herramienta fue creada para ser utilizada en sistemas de software propietario, específicamente Windows, para analizar bases de datos sobre gestores de licencia privativa.

De lo anterior surge el siguiente **problema científico:** “En los sistemas de software libre no existen herramientas para el análisis de las bases de datos libres, por lo que no se tiene el suficiente control en los datos manejados dentro de los Almacenes de Datos".

El **objeto de investigación** se centra en las bases de datos, su estudio y análisis, mientras que el **campo de acción** estaría enmarcado en el estudio y análisis de las bases de datos diseñadas en Software Libre.

Como **objetivo principal** de este trabajo de diploma se encuentra la migración de la herramienta DBAnalyzer a Software Libre, para así extender el análisis de los sistemas de Bases de Datos a gestores libres.

Como **objetivos específicos** se pueden citar los siguientes:

- Realizar una amplia investigación en internet sobre el tema de la limpieza de datos, y dejar una constancia que sirva para planes futuros.
- Migración de la herramienta DBAnalizer 1.0 a Software Libre para el análisis de la información contenida en las bases de datos libres.
- Hacer un aporte en el proceso de mejora de la calidad de datos de la Universidad de las Ciencias Informáticas.

Para cumplir el objetivo del presente trabajo serán necesarias varias **tareas de investigación**, estas son:

- Realizar un estudio profundo sobre el desarrollo de los métodos de limpieza de datos en sistemas de bases de datos, así como de las posibles clasificaciones de errores.
- Realizar la implementación de la herramienta en software libre.
- Analizar algunas Bases de Datos de la Universidad de las Ciencias Informáticas (UCI) en gestores libres, tales como: PostgreSQL y MySQL.

Para el desarrollo completo del trabajo y su total entendimiento se hizo necesario emplear métodos de investigación tales como: el **histórico lógico**, que se utilizó para el estudio de la evolución y desarrollo del tema de la limpieza de datos en los almacenes de datos. Éste se llevó a cabo con una profunda investigación en Internet de todo lo relacionado con los sistemas de bases de datos y la limpieza de datos sucios o errores que en ellas se pueden encontrar; el método **analítico-sintético** se utilizó para el procesamiento de información y elaboración de conclusiones. Este método ha servido para analizar y comprender la teoría y documentación relacionada con el tema de investigación, permitiendo así extraer los elementos más importantes relacionados con el objeto de

estudio; el método empírico de **análisis de documentos**, porque se han analizado documentos, la mayoría de ellos trabajos investigativos relacionados con la limpieza de datos y con las bases de datos en general.

El presente documento se estructura en tres capítulos y las correspondientes conclusiones, recomendaciones, referencias bibliográficas, bibliografía y glosario de términos.

Capítulo 1: Estado del arte de la Limpieza de Datos.
En este primer capítulo es ampliamente descrita la situación actual del problema de la limpieza de datos. En él son definidos varios conceptos y características que facilitan la comprensión del proceso en cuestión. Además se describen algunas de las herramientas y métodos más utilizados en este medio.

Capítulo 2: Características del sistema.
En este capítulo son analizadas las características que debe poseer un sistema de este tipo, son planteados los procesos principales que debe seguir además de un modelo de dominio y se modelan en forma de casos de uso los requisitos no funcionales del sistema.

Capítulo 3: Ciclo de vida del sistema.
En este capítulo es analizada la solución del problema, poniendo al descubierto todos los procedimientos a seguir durante todo el ciclo de vida del sistema, comenzando desde el análisis hasta la implementación de una propuesta de analizador, proceso que se evidenciará mediante diagramas y conceptos de los flujos de trabajo de análisis, diseño e implementación.

1 Estado del arte de la limpieza de datos.

La introducción de errores en los datos se hace cada vez más frecuente, aún cuando se realizan grandes esfuerzos por parte de los desarrolladores de sistemas para evitar los errores en los datos, la razón de error[1] es aproximadamente de un 5% (REDMAN 1998). Para solucionar este problema se han planteado diversas soluciones, coincidiendo en la mayoría de los casos en que hacer este proceso de forma manual es muy engorroso, lento y que se pueden de esta forma introducir nuevos errores, de ahí que la automatización de este proceso se ha convertido en una tarea de primer orden debido a la importancia que reviste.

En Cuba aún no se cuenta con herramientas de este tipo, aunque ya se aprecian los primeros pasos en este proceso que involucra grandes investigaciones y un trabajo en equipo, tampoco se cuenta con ningún procedimiento con este fin en la universidad, por lo que son detectados grandes cantidades de errores en la información contenida en las bases de datos.

1.1 Limpieza de datos.

El proceso de Limpieza de Datos, o *Data Cleaning* como también se le conoce, no es más que corregir o remover información incorrecta, con formato inapropiado o duplicada en una base de datos a través de métodos computarizados, el cual se ha convertido en elemento clave debido al gran volumen de información que actualmente se maneja dentro de cualquier contexto en las diferentes empresas a nivel mundial.

En general, el campo de trabajo de la limpieza de datos se centra en:

- definir y determinar los tipos de errores
- buscar y determinar 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

Los principios de la limpieza de datos deben estar presentes en todas las etapas del proceso de manipulación de los datos (captura, digitalización, almacenamiento, análisis, presentación y uso). Existen dos elementos claves a tener en cuenta para mejorar la calidad de los datos, que son prevención y corrección. A pesar de los esfuerzos que se puedan hacer para la prevención, se

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

puede dar por hecho que en casos de grandes cantidades de datos a entrar siempre van a existir errores (MALETIC and MARCUS 2000b), por lo que la validación de datos y la corrección no pueden ser ignorados.

Tener una buena documentación es un principio fundamental en la gestión de información, por lo que sin ella el usuario no podría determinar la idoneidad de los datos para el uso que tiene en mente y por lo tanto no puede determinar la calidad de la información para ese propósito.

Algunos de los principios de la limpieza de datos son:

- **Retroalimentación**

  La retroalimentación funciona como un camino de dos vías. Es importante que los responsables del cuidado de la información alienten a los usuarios a dar su opinión acerca de los datos y que las tomen en serio, pues es muy cierto que los usuarios tienen más posibilidades de detectar errores en los datos por la combinación de varias fuentes que las que tiene el administrador trabajando en solitario.

  A veces suele ser difícil establecer la comunicación entre los administradores de datos y el cliente, pero siempre debe haber un canal para que esta retroalimentación se desarrolle y así poder reducir la incidencia de futuros errores y mejorar la calidad de los datos. De hecho los canales de retroalimentación eficaces con los usuarios son la mejor forma de mejorar la calidad de los datos.

- **Educación y Entrenamiento**

  La educación y formación en todos los niveles de la cadena de información, puede llevar a mejorar sustancialmente la calidad de los datos.

  Todo comienza con la educación y la capacitación de los recolectores en el buen uso de los procedimientos de recolección e implementación de las necesidades de los usuarios, a través del entrenamiento de los operadores de entrada de datos y el personal técnico responsable de la gestión diaria de información en las bases de datos, a través de la educación de los usuarios finales, así como de la naturaleza de los datos, sus limitaciones y usos potenciales. Los aspectos de la educación y capacitación de la calidad de datos son en gran medida dependientes de la buena documentación. En la deficiente capacitación se encuentra la raíz de muchos de los problemas de la calidad de los datos.

- **Rendición de cuentas**

  La rendición de cuentas por la calidad de datos puede ayudar a las organizaciones a lograr un nivel de control de calidad, provee un punto de referencia para la retroalimentación sobre la información de errores y un punto de contacto para la documentación y las consultas.

- **Responsabilidad, transparencia, habilidad en la auditoria**

La responsabilidad, transparencia y la habilidad en la auditoría, son elementos esenciales en la limpieza de datos, pues ejercicios de limpieza de datos casuales y no planeados son ineficientes y generalmente improductivos. Dentro de las políticas de calidad de datos y las estrategias, deben ser establecidas líneas de responsabilidad concretas. Para mejorar la aptitud para el uso de los datos, además de su calidad, el proceso de limpieza de datos debe 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.

- **Planificación**

La planificación es esencial, desarrollar una política y una estrategia para la manipulación de datos constituye un principio fundamental de la limpieza de datos. La cadena de gestión de la información (Figura 1) incluye a la limpieza de datos como una parte central que necesita ser incorporada dentro de la organización de la visión y la política de la calidad de datos. Una estrategia para implementar la depuración de datos y la validación en una cultura de organización mejorará la calidad general de la organización de los datos y su reputación con los usuarios y proveedores por igual.

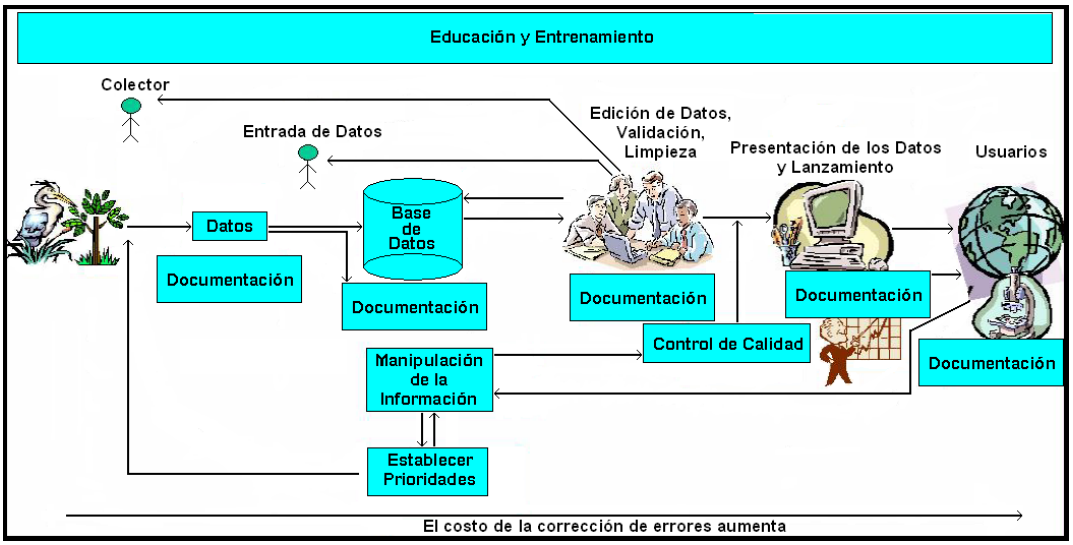


Figura 1 Cadena de Manipulación de la Información, muestra como el costo de la corrección de errores aumenta a medida que se avanza en la cadena. Educación y Entrenamiento son necesarios en todos los pasos (CHAPMAN, ARTHUR D. 2005b).

- **Organización de los datos**

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

- **Prevenir**

Es más barato y más eficiente prevenir que curar. Según esto es mejor prevenir que ocurra un error que detectarlo y corregirlo después, y aquí juega un papel importante la retroalimentación del administrador del sistema, pues se hace necesario que entre el administrador y el usuario se establezcan los canales apropiados de intercambio de información, para asegurar que no ocurran los mismos errores en el proceso de entrada de datos nuevamente, o que sea más baja la probabilidad de que estos ocurran.

- **La responsabilidad**

La responsabilidad de la limpieza de datos pertenece a todos, colectores, administradores y usuarios. La responsabilidad principal recae sobre el administrador, aunque el colector también tiene responsabilidades, ya que debe responder al administrador cuando éste encuentre algún error o ambigüedad en los datos recolectados por él. El usuario también tiene una responsabilidad importante, que es la de comunicar al administrador cualquier error u omisión que pudiera ocurrir, así como cualquier error en la documentación relacionado con la información, poniendo en práctica los mecanismos de retroalimentación.

- **La asociación**

Las asociaciones pueden ser un mecanismo muy eficiente de manipulación de la limpieza de datos, pues como mencionamos anteriormente el usuario es la persona que generalmente está en la mejor posición de identificar errores en los datos. Si los administradores pueden realizar asociaciones con estos usuarios claves, entonces los errores no serán ignorados. Con el desarrollo de estas asociaciones no será necesario duplicar muchos de los procesos de validación, además los errores serán más fáciles de documentar y corregir.

- **Priorizar**

El establecimiento de prioridades junto a la organización y el ordenamiento ayudan a reducir los costos y mejorar la eficiencia, además de reducir la duplicación. Para ello es importante concentrarse en la limpieza de aquellos registros en los que se almacena gran cantidad de datos, que pueden ser limpiados con un costo mucho menor, y que además son los más usados por los usuarios. Es en estos registros en los que se pueden detectar y corregir la mayor cantidad de errores, siendo esto una mejora en la relación entre el usuario y el proveedor, lo cual incentiva tanto a proveedores como usuarios a mejorar la calidad de los datos.

- **Establecer objetivos y desarrollar medidas**

El desarrollo de medidas es una adición valiosa al procedimiento de control de calidad y son usadas extensivamente con los metadatos espaciales. Éstas ayudan a la organización del proceso de limpieza de datos y proveen a los usuarios con información sobre los datos y su calidad, lo cual puede ser usado por los administradores para detectar aquellas partes de la base de datos que pueden necesitar atención. El desarrollo de medidas puede incluir el chequeo

estadístico de los datos, en el nivel de control de calidad e integridad (CHAPMAN, ARTHUR D 2005a).

- **Minimizar la repetición del proceso de limpieza de datos**

La duplicación es un factor importante en la organización de la limpieza de datos que muchas veces provoca el encarecimiento del proceso. La documentación también es importante en el proceso de validación para reducir 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 (CHAPMAN, ARTHUR D 2005a).

- **Documentación**

La documentación es la clave para una buena calidad de datos, pues a través de ella es más fácil para los administradores saber quién realizó el chequeo de la calidad de los datos y es más fácil para los usuarios determinar el grado de corrección de los datos. La documentación generalmente es de dos tipos, el primero está unido a cada registro, registros en los cuales ya se ha llevado a cabo el chequeo de los datos y que se han hecho cambios, el segundo tipo es el metadato que registra la información al nivel de grupo de datos. Ambos tipos de documentación son importantes, sin ellos una buena calidad de los datos se vería comprometida.

1.3 Calidad de los datos.

La existencia de datos pobres o erróneos causan el llamado problema de la basura *(Garbage – in, Garbage - out)*. Para cualquier análisis crítico o aplicación la principal tarea es la mejora de la calidad de los datos, sin embargo como las fuentes son diversas, los formatos heterogéneos y el volumen de los datos crece rápidamente, el mantenimiento y la mejora de los datos se hace más difícil (REDMAN 2001).

A pesar de las diferentes definiciones, los estudios y los acercamientos a la calidad de los datos *(Data Quality (DQ))*, existe un consenso al plantear que la calidad de los datos no puede ser evaluada independientemente de la persona que va a utilizarlos (STRONG *et al.* 1997), además que el concepto de calidad solamente puede ser aplicado en el momento en que los datos van a utilizarse, pues éstos no tienen valor (e incluso calidad), solamente un valor potencial, ya que adquieren su verdadero valor a la hora de ser usados para un fin determinado (ENGLISH 1999).

En la literatura la calidad de los datos es presentada como un concepto multidimensional (WANG and STRONG 1996). Una de las definiciones más completas está expresada en una colección de 15 dimensiones organizadas dentro de 4 categorías (Tabla 1).

Categorías de la Calidad de los Datos	Dimensiones de la Calidad de los Datos
Intrínseco	Precisión, Objetividad, Credibilidad, Reputación
Accesibilidad	Accesibilidad, Seguridad de acceso
Contextual	Relevancia, Valor añadido, Oportuno, Integridad, Cantidad de datos
Representativo	Interpretable, Fácil de entender, Representación concisa, Representación consistente

Tabla 1 Categorías y dimensiones de la calidad de datos.

- **Calidad de datos intrínseca**: no solo incluye a la precisión y objetividad (esto es evidente para profesionales de sistemas de información), sino también credibilidad y reputación. Esto sugiere que, contrario a la vista de desarrollo tradicional, los consumidores de datos también ven la reputación y la credibilidad como parte integral de la calidad de datos intrínseca, debido a que la precisión y la objetividad por sí solas no son suficientes para que los datos tengan una buena calidad.
- **Calidad de datos accesible**: los usuarios finales ven esta categoría como un importante aspecto en la calidad de los datos. No obstante, existe una diferencia entre tratar a la accesibilidad como una categoría de la calidad de datos global, o separándola de otras categorías de la calidad de datos, en cualquiera de los casos debe ser tomada en cuenta.
- **Calidad de datos contextual**: no está explícitamente reconocida en la literatura sobre la calidad de los datos. Esta agrupación de las dimensiones de la calidad de datos contextual revela que la calidad de los datos debe ser considerada dentro del contexto de la tarea a mano.
- **Calidad de datos representativa**: incluye aspectos relacionados con el formato de los datos (representación concisa y consistente) y su significado (interpretables y fáciles de entender). Problemas relacionados al significado y formato surgen en los sistemas de base de datos en los cuales el formato es visto como parte de la sintaxis y el significado como parte de la conciliación semántica.

1.4 Taxonomía de errores.

Los errores que se encuentran mediante el proceso de detección y limpieza de datos se presentan agrupados en varios grupos. Aunque no resulta sencilla esta clasificación, con los métodos computacionales resulta mucho más fácil agrupar dichos errores para su posterior análisis y corrección.
Entre las clasificaciones más comunes están:

- **Anomalías sintácticas.**

Las anomalías sintácticas son aquellas relativas a las alteraciones que se pueden producir en las gramáticas, que impiden la correcta formación de oraciones y conceptos, o que afectan a las reglas que definen las secuencias correctas de los elementos de un lenguaje. Este tipo de anomalías se pueden dar en tres formas (MARMOL 2005).
Errores léxicos: Son aquellos errores que se dan en el vocabulario, conjunto de palabras que pertenecen a un campo semántico dado en los que la estructura de los datos difiere del formato especificado para dichos datos, 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: Los errores en el formato o las características de un valor son aquellos donde el valor dado por un atributo A no se ajusta con el dominio u orden determinado del formato previsto anteriormente G(dom(A)).
Irregularidades: Son los errores relacionados 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 de su representación para poder interpretarlos. Otro ejemplo es el uso o el diferente uso de las abreviaturas.

- **Anomalías semánticas.**

Las anomalías semánticas son aquellas discrepancias de una regla relativas a la significación de las palabras, de ellas se dan cuatro casos:
Violaciones de las restricciones de integridad: Son aquellas en las que se describen tuplas (o grupos de tuplas) que no satisfacen una o más de las restricciones de integridad. Las restricciones de integridad son usadas para describir el 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: Las contradicciones son violaciones de la dependencia funcional que pueden ser representadas como restricciones de integridad o duplicados con valores inexactos, 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.

Duplicados: Es cuando dos o más tuplas representan 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: Éstas representan la anomalía más complicada encontrada en colecciones de datos. Por tuplas 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.

- **Anomalías de alcance.**

Valores que faltan: Son el resultado de omisiones en el proceso de colección de los datos.

Tuplas que faltan: Es el resultado de las omisiones de entidades completas existentes en el mini-mundo que no son representadas por tuplas en la colección de datos (REDMAN 2001).

Existen otras taxonomías o clasificaciones de errores que en esencia se resumen en la descrita anteriormente, ejemplo de estas son las dadas en (QUASS 1999) (GREENFIELD 2000), (AHO 1979).

1.5 Métodos utilizados para la limpieza de datos.

Las bases de datos en la actualidad son muy susceptibles a los errores, las pérdidas e incluso a la inconsistencia de los datos, debido a la gran cantidad de información almacenada en las mismas. Para ello existen numerosos métodos o técnicas para el procesamiento de los datos que pueden ser aplicadas para eliminar los errores y corregir las inconsistencias. La integración de los datos combina la información proveniente de distintas fuentes en un almacén de datos coherente.

La transformación de los datos como la normalización puede ser aplicada para mejorar la precisión y la eficiencia de los algoritmos en la minería, involucrando así grandes dimensiones de datos. Por ejemplo, se pueden reducir los datos adicionando la eliminación de características redundantes o el

agrupamiento. Estas técnicas de procesamiento, cuando son aplicadas antes de la minería, pueden mejorar substancialmente la calidad y el tiempo requerido en la minería de datos.
Entre los métodos más famosos, están los siguientes:

- **Análisis gramatical.** *(Parsing)*

El análisis gramatical es desarrollado para la detección de errores sintácticos mediante analizadores gramaticales, los cuales deciden si una cadena dada es un elemento del lenguaje definido. En el proceso de limpieza de datos las cadenas son tuplas completas de una instancia relacional o valores de 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 se usa una gramática derivada de la estructura del fichero. Los datos son manejados por sistemas de administración de bases de datos, que 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 (AHO 1979), (RAMAN 2001).

- **Transformación de datos.** *(Data transformation)*

La transformación de los datos propone transformar los datos del formato dado a un formato esperado por la aplicación, esto involucra el esquema de tuplas y el dominio de sus valores. El esquema de transformación es desarrollado en el proceso de limpieza de datos, los cuales son transformados en un esquema común de acuerdo a las necesidades de la aplicación. La corrección de los valores debe ser desarrollada solo en casos donde los datos de entrada no se corresponden con el esquema y puedan llevar a fallos posteriores en el proceso de transformación (MARMOL 2005). Esto hace que la limpieza de datos y el esquema de transformación se vean como tareas suplementarias. Puede involucrar las siguientes técnicas (HAN and KAMBER 2000):

  - **Smoothing**: Facilitar el trabajo de eliminación de los datos con errores, que puede incluir agrupamiento y regresión.
  - **Agregación** (Agregation): Son aplicadas operaciones de resumen o agregación de los datos. Este paso es usado generalmente en la construcción de cuadros de datos para el análisis de los datos en múltiples granularidades.
  - **Generalización** (Generalization): Donde los datos de bajo nivel o primitivos son sustituidos por conceptos de alto nivel a través del uso de conceptos jerárquicos. Por ejemplo tenemos los atributos categóricos tales como *calle* que puede ser generalizado en un concepto de más alto nivel como *ciudad* o *provincia*, también en valores de atributos numéricos como *edad* que puede ser generalizado en conceptos de alto nivel como: *joven, adulto* o *anciano* (HAN and KAMBER 2000).

- **Normalización** *(Normalization)*: La normalización es una transformación en el nivel de instanciación utilizada 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 (ABITEBOUL *et al.* 1999), (SATTLER, KU 2001), (MARMOL 2005), donde los atributos de los datos están en una escala dentro de un rango determinado, por ejemplo, -1.0 a 1.0, 0 a 1.0.
- **Construcción de atributos** *(Attribute construction***:** or feature construction): Cuando los atributos son construidos y adicionados desde un conjunto de atributos dado, para ayudar en el proceso de minería de datos.

- **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, ya sea 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 (MAYOL 1999).

- **Eliminación de duplicados.** *(Duplicate Elimination)*

El algoritmo de eliminación de duplicados ordena todos los atributos en grupos de tuplas similares; donde cada tupla debe ser comparada con el resto mediante métodos de detección de duplicados, algunos de los cuales se explican a continuación:

Sorted Neighbourhood Method (HERNÁNDEZ 1995): es un método rápido, que 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 se comparan las tuplas que están en un mismo grupo. La identificación de tuplas duplicadas se hace usando reglas basadas en el conocimiento específico del dominio. Para mejorar la precisión, los resultados de varios pases de la detección de duplicados pueden ser combinados con el cálculo de cercanía transitiva de todos los pares de tuplas duplicadas encontradas.

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. Éstas 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.

- **Métodos estadísticos.** *(Statistical Methods)*

Los métodos estadísticos pueden ser usados para la verificación de los datos, así como en la detección y 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 minoría 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 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 (MALETIC and MARCUS 2000a).

1.6 Herramientas de limpieza de datos.

- **AJAX**: 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 más 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: asignación *(mapping)*, visión *(view)*, correspondencia *(matching)*, agrupación *(clustering)* y fusión *(merging)* (GALHARDAS *et al.* 2000), (GALHARDAS *et al.* 2001).
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.

- **Arktos:** 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 en 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 especifica el comportamiento (la acción que será desarrollada) en caso de la ocurrencia de un error (VASSILIADIS *et al.* 2001).
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.

- **IntelliClean:** 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 situación particular, esto habilita la especificación de las reglas de puesta en vigor de las restricciones de integridad, para cada restricción de integridad se define una regla de actualización que 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 (LEE and LOW 2000), (LOW 2001).
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.

- **Potter´s Wheel:** 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 interfaz. 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 (RAMAN 2001).
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.

- **FraQL:** 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 (SATTLER, K U and SAAKE 2000), (SATTLER, KU 2001).

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 éstos 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.

- **FuzzyDupes 2007:** Es una herramienta con licencia *Shareware* y su autor es *Kroll Software-Development;* esta herramienta tiene como objetivo la búsqueda de duplicados borrosos*.* Para este propósito se utilizan algoritmos que comparan cadenas de caracteres y detectan patrones recurrentes dentro de esas cadenas, más conocidos como algoritmos de similitudes de patrones (*pattern matching algorithms*), un ejemplo de ellos es la métrica de similitud (también conocido como Levenshtein distance metric). Este algoritmo muestra el número fundamental de pasos (insertar, modificar, eliminar) necesarios para convertir una cadena de caracteres A en una cadena de caracteres B. Como el número de comparaciones a hacer se incrementa gradualmente se intenta agrupar todos los registros antes de iniciar la búsqueda, para así poder hacer las comparaciones incluso en grandes bases de datos. Otro paso a realizar es la normalización de los datos, donde se reemplazan los caracteres especiales por abreviaturas comunes para facilitar la búsqueda de resultados desde afuera. Estas modificaciones son triviales y tendrán un impacto secundario en los resultados.

- **WinPure Clean & Match 2007:** Es un software de limpieza de datos, que puede importar varias listas a la misma vez, incluye opciones avanzadas para la búsqueda de similitudes, provee cinco módulos de limpieza de datos que ayudan a asegurar que las listan están totalmente llenas, que son precisas y que tienen las direcciones correctas, también utiliza algoritmos de similitudes de patrones y utiliza métodos para eliminar el problema de mezcla/limpieza *(merge/purge)*. Es utilizada para limpiar listas de correo electrónico, bases de datos de comercialización, hojas de cálculo y correos electrónicos.

- **ListCleaner Pro:** Es un software para la limpieza de datos y duplicados y para la corrección de bases de datos, hojas de cálculo, correos electrónicos, etc. Puede manejar listas de contactos electrónicos, listas de direcciones electrónicas, listas de categorías y precios de productos, detalles de ventas y nombres de estudiantes, entre otras. Para buscar duplicados el programa analiza las cadenas de datos y sus comparaciones lógicas, también elimina signos de puntuación indeseados y errores ortográficos e identifica errores faltantes.

2 Características del sistema.

En este capítulo se presentan las características del sistema que se propone para dar solución al problema de la limpieza de datos, el cual tiene una gran influencia en las bases de datos. Se describen elementos del proceso de limpieza de datos, para lo cual se hace necesario definir conceptos y agruparlos en un modelo de dominio donde se capturen los requerimientos funcionales y no funcionales que debe cumplir el sistema, de forma tal que al finalizar se haya comprendido el proceso de análisis de bases de datos.

Para cumplir el objetivo de este trabajo se hizo necesaria una gran investigación sobre el proceso de limpieza de bases de datos para poder automatizarlo, de ahí que se decidiera hacer una segunda versión de una herramienta (DBAnalyzer 1.0) desarrollada sobre software propietario y en lenguaje de programación Delphi. La segunda versión de la herramienta tomará como nombre DBAnalyzer 2.0 y será desarrollada en Software Libre con la herramienta NetBeans y el JDK 1.6, además de que será programada en lenguaje Java. Esta herramienta deberá realizar un análisis estadístico de los datos presentes en la base de datos, el cual mostrará en un reporte con detalles de esta información.

2.1 Modelo de Dominio.

Como lo que se espera de este proyecto es una segunda versión de un software se realizará un modelo de dominio (Figura 2) para obtener una mejor visión de los principales conceptos que se manejan en el dominio del sistema en desarrollo provocando así que los usuarios, clientes, desarrolladores e interesados utilicen un vocabulario común para entender el contexto en que se enmarca el sistema.

El primer paso en el modelo de dominio es identificar todos los conceptos que se utilizarán en el diagrama mediante un **glosario de términos**:

Administrador: Persona encargada de ejecutar la aplicación y de realizar todas las operaciones necesarias para obtener los resultados esperados.

Base de Datos (BD): Conjunto de datos relacionados entre sí.

Campo: Es la unidad menor de información sobre un objeto (almacenada en la base) y representa una propiedad de un objeto (por ejemplo, el color).

Dato: Hechos conocidos, que pueden registrarse y que tienen significado implícito. Ejemplo: Nombre, Número telefónico, etc.

Reporte: Documento generado por la aplicación que contiene los resultados del análisis de la base de datos.

***Sistema Gestor de Base de Datos (SGBD)**:* conjunto de programas que permite a los usuarios crear y mantener una BD, es un software de propósito general que facilita el proceso de definir, construir y manipular la BD para diversas aplicaciones.

Tabla: Archivo simple dentro de una base de datos, compuesto por filas y columnas. Las filas representan tuplas y las columnas los atributos o campos.

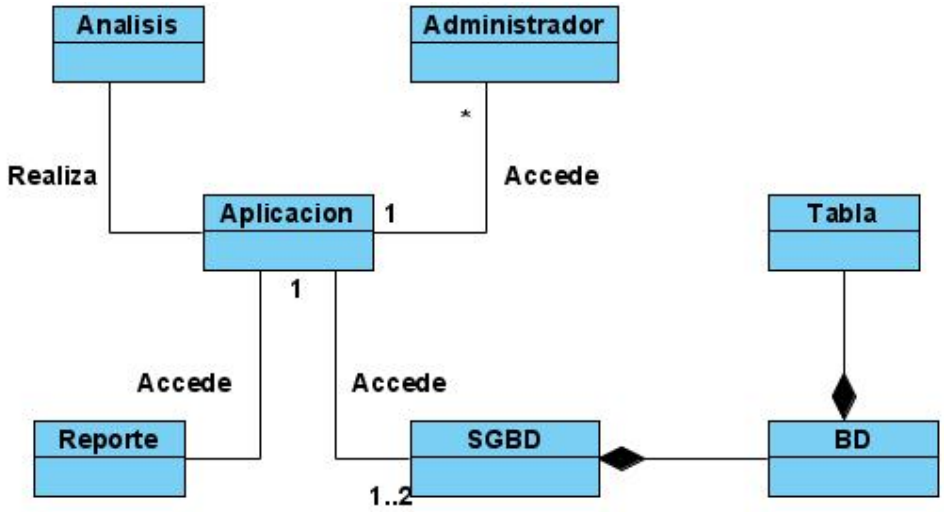


Figura 2 Diagrama del modelo de dominio.

2.2 Requerimientos del sistema.

En el levantamiento de requisitos se define que es lo que debe hacer el sistema que se construye, estos requerimientos tienen que ser aceptados y aprobados por los usuarios finales, de ahí que sea un flujo de trabajo muy importante en el proceso de creación del producto. Los requisitos pueden ser de dos tipos, funcionales y no funcionales.

Requerimientos funcionales.

Los requisitos funcionales son aquellos que definen condiciones o capacidades que el sistema debe cumplir. Los requisitos funcionales que identifican este sistema son:

R1 Permitir conectarse a las Bases de Datos.

- **R1.1** Permitir conectar al SGBD PostgreeSQL.
- **R1.2** Permitir conectar al SGBD MySQL.
- **R1.3** Conectar utilizando dirección (URL) del servidor y nombre de la BD.

R2 Permitir autenticar usuario.

- **R2.1** Permitir autenticar utilizando usuario y contraseña.

R2.2 Permitir autenticar utilizando usuario solamente.

R2.3 Permitir autenticar utilizando usuario anónimo.

R3 Analizar tablas.

R3.1 Mostrar tablas de la BD.

R3.2 Permitir escoger una tabla

R3.3 Permitir escoger algunas tablas.

R3.4 Permitir escoger todas las tablas.

R3.5 Permitir analizar una tablas.

R3.6 Permitir analizar algunas tablas.

R3.7 Permitir analizar todas las tablas.

R4 Gestionar reporte.

R4.1 Generar reporte.

R4.2 Mostrar reporte.

R4.3 Buscar reporte.

Requisitos no funcionales.

Los requerimientos no funcionales son propiedades o cualidades que el sistema debe cumplir, los que se identifican en este sistema son:

Usabilidad

Para usar el sistema se deben tener conocimientos elementales de computación y bases de datos.

Soporte

Para usar el sistema se necesita el JDK 1.6, una conexión de red hacia un servidor de base de datos.

Portabilidad

El sistema deberá ser multiplataforma.

Seguridad

El sistema no almacenará ni guardará las contraseñas proporcionadas mediante la autenticación a las bases de datos.

Legales

El sistema deberá poseer mecanismos para brindar respuesta ante posibles fallas.

Interfaz Externa

El diseño de la interfaz deberá ser claro y preciso para guiar al usuario en los procesos a realizar. Para una mejor visualización debe utilizarse una resolución de 1024 x 768 píxeles.

Restricciones de la implementación

Se programará en lenguaje Java, en la herramienta NetBeans, solo permitirá conexiones a BD en PostgreSQL y MySQL.

Confiabilidad

Garantizar la veracidad y rigor estadístico de los reportes.

Ayuda

El sistema contará con un manual de ayuda al usuario.

2.3 Casos de uso del sistema.

Los actores presentes en el sistema son:

Actor	Descripción
Administrador o Usuario	Es el encargado de ejecutar la aplicación y realizar todos los pasos necesarios para el análisis de los datos.

Diagrama de casos de uso del sistema.

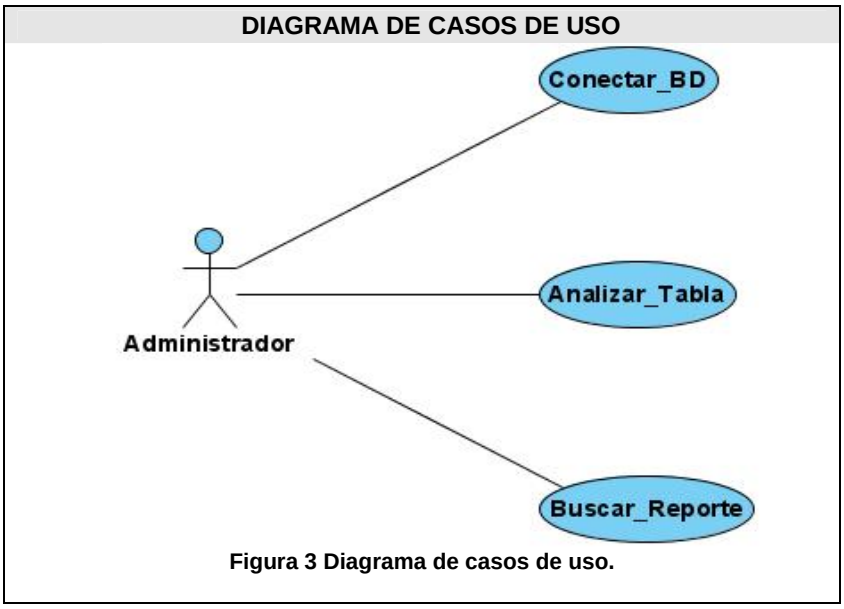


Figura 3 Diagrama de casos de uso.

Casos de uso del sistema.

CU-1	Conectar_BD
Actor	Administrador

Descripción	El administrador selecciona el gestor de base de datos apropiado, introduce la dirección (URL) del servidor donde se encuentra la base de datos y el nombre de la misma, autenticándose con los permisos adecuados.
Referencia	R1, R2

CU-2	Analizar_Tabla
Actor	Administrador
Descripción	Después que el sistema muestre las tablas de la base de datos el administrador selecciona las tablas que desee y pulsa el botón analizar, tras lo cual el sistema analizará las tablas y guardará un reporte en el disco duro en la dirección que el usuario especifique.
Referencia	R3, R4

CU-3	Buscar_Reporte
Actor	Administrador
Descripción	El administrador va al menú archivo y selecciona Abrir Reporte, le aparecerá una nueva ventana que tiene un menú en el cual seleccionará Buscar Reporte y mediante una ruta de acceso abrirá el reporte que será mostrado en la nueva ventana.
Referencia	R4

2.4 Descripción extendida de los casos de uso del sistema.

Caso de uso	
CU-1	Conectar a BD
Propósito	Establecer una conexión con la base de datos.
Actores: Administrador	
Resumen: El caso de uso comienza cuando el administrador selecciona el gestor de base de datos adecuado, introduce la dirección (URL) del servidor donde se encuentra dicha base de datos y el nombre de la misma, y por último se autentica según los permisos que posea en la BD a la cual está intentando conectarse.	
Referencias	R1, R2

Precondiciones	El administrador debe conocer dirección del servidor, el nombre de la base de datos y saber los datos de autenticación.
Postcondiciones	El administrador puede acceder a la base de datos y ver sus tablas.

Acción del actor	**Respuesta del sistema**
1. El administrador selecciona el gestor de base de datos en el cual se encuentra la base de datos deseada. 2. Introduce la dirección (URL) del servidor y el nombre de la base de datos. 3. Se autentica.	3.1 El sistema verifica que toda la información sea correcta. 3.2 Si la información es correcta accede a la base de datos y muestra las tablas.
Flujo alternativo 1	
Acción del actor	**Respuesta del sistema**
	3.2 Si la información es incorrecta o hay campos vacíos, el sistema muestra un mensaje de error.

Caso de uso	
CU-2	Analizar_Tabla
Propósito	Analizar la(s) tabla(s) seleccionada(s) por el usuario.
Actores: Administrador	
Resumen: Después que el sistema muestre las tablas de la base de datos el administrador selecciona las tablas que desee y pulsa el botón analizar, tras lo cual el sistema analizará las tablas y mostrará un cuadro de búsqueda para que el usuario seleccione la ruta donde desea guardar el reporte.	
Referencias	R3, R4
Precondiciones	Debe haberse conectado exitosamente a una base de datos y esta debe contener menos una tabla.
Postcondiciones	Se tiene un documento (.txt) a modo de reporte guardado en el disco duro de la máquina que contiene toda la información sobre el análisis efectuado en la base de datos.

Acción del actor	**Respuesta del sistema**
1. El administrador selecciona la(s) tabla(s) que desea analizar.	

2. Solicita Analizar. 3. Introduce ruta del reporte.	2.1 El sistema analiza la(s) tabla(s) seleccionadas y genera un reporte. 2.2 Solicita ruta para guardar el reporte en el disco duro de la computadora. 3.1 Guarda el reporte.
Flujo alternativo 1	
Acción del actor	**Respuesta del sistema**
1. No selecciona ninguna tabla. 2. Pulsa el botón Analizar.	 2.1 El sistema muestra el mensaje de que debe seleccionar al menos una tabla.

Caso de uso	
CU-3	Buscar_Reporte
Propósito	Buscar un reporte ya hecho.
Actores: Administrador	
Resumen: El sistema da la oportunidad de que el usuario mediante una nueva ventana busque un reporte existente en el disco duro de la computadora.	
Referencias	R4
Precondiciones	
Postcondiciones	Se muestra en una ventana el reporte solicitado.
Acción del actor	**Respuesta del sistema**
1. El administrador va al menú Archivo y solicita Abrir Reporte. 2. Solicita Buscar Reporte. 3. El administrador señala la ruta de acceso del reporte que desea ver.	1.1 El sistema muestra una nueva ventana con un menú. 2.1 Muestra un cuadro de búsqueda. 3.1 El sistema muestra el reporte en la nueva ventana.
Flujo alternativo	
Acción del actor	**Respuesta del sistema**

Luego de realizada la representación de las funcionalidades del sistema a través del Diagrama de Casos de Uso y de haber sido descritas las acciones de los actores del sistema con los casos de uso que interactúan, es posible comenzar a desarrollar la propuesta de solución del sistema, teniendo en cuenta el cumplimiento de los requerimientos especificados.

3 Ciclo de vida del sistema

En este capítulo se forma la estructura de la herramienta DBAnalyzer 2.0 mediante los flujos de trabajo Análisis y Diseño e Implementación. Con el objetivo de brindar una herramienta sencilla capaz de brindar un análisis confiable de la información contenida en una base de datos. A continuación se muestra el proceso detallado de lo que ocurre en estos flujos de trabajo, como se modela el análisis, como se lleva a cabo el diseño de la herramienta, tomando las soluciones más sencillas y eficientes, e incluso se muestra la situación física de los componentes lógicos desarrollados a través del modelo de despliegue.

3.1 Modelo de Análisis

El análisis consiste en obtener una visión del sistema con el propósito de conseguir una comprensión más precisa de los requisitos, refinarlos y estructurarlos y utilizar el lenguaje de los desarrolladores para analizar con profundidad los requisitos funcionales. El objetivo de este flujo de trabajo es traducir los requisitos a una especificación que describe cómo implementar el sistema.

Diagramas de clases del análisis

DIAGRAMA DE CLASES DE ANÁLISIS (CU-2 Analizar_Tabla)

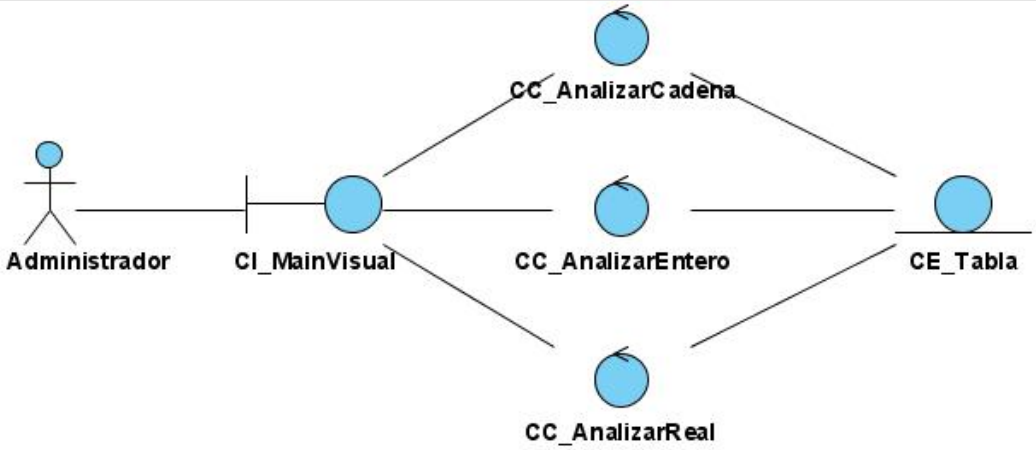
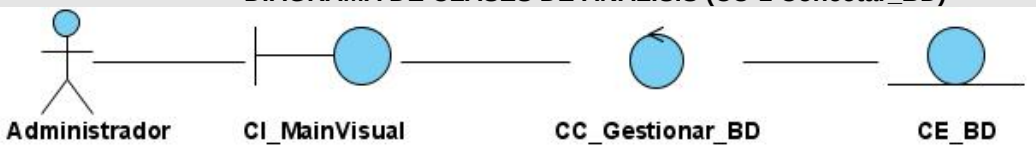
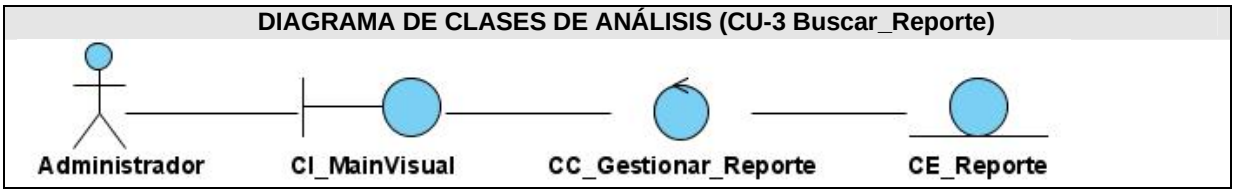


DIAGRAMA DE CLASES DE ANÁLISIS (CU-1 Conectar_BD)

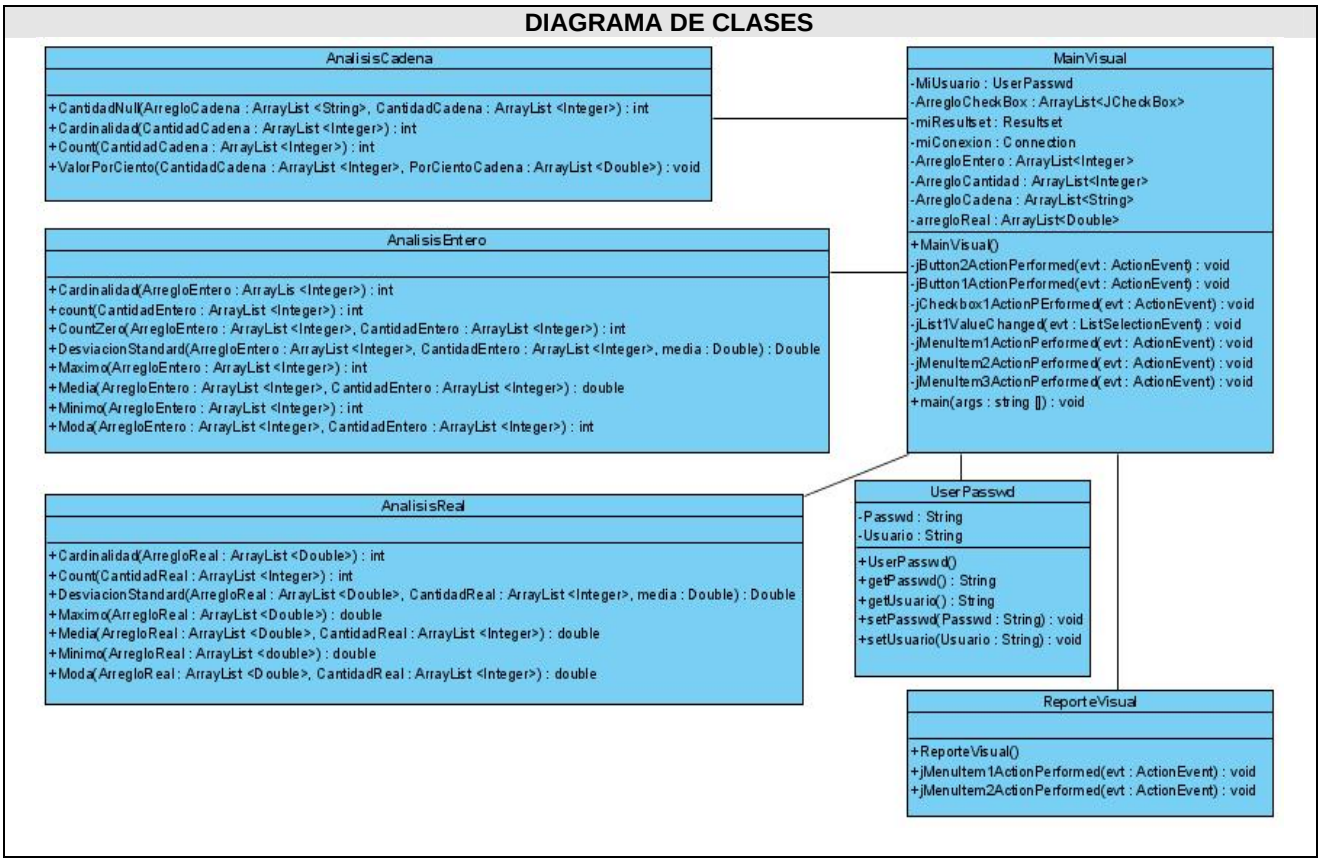




3.2 Diseño

El diseño es un refinamiento del análisis que tiene en cuenta los requisitos no funcionales para saber cómo el sistema debe ser implementado en forma tal que pueda cumplir sus objetivos y que no contenga ambigüedades.

Diagrama de clases



Diagramas de interacción

DIAGRAMA DE INTERACCION (CU-1 Conectar_BD)

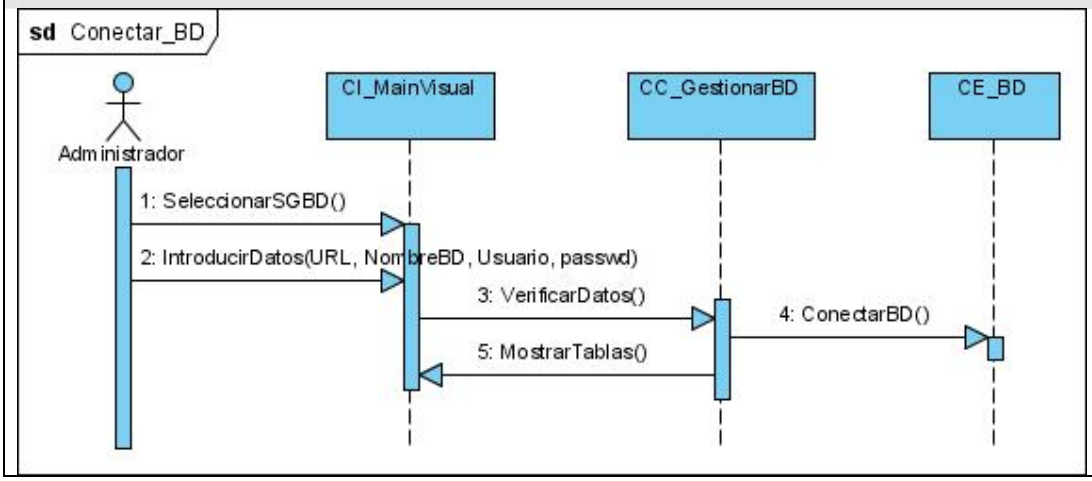


DIAGRAMA DE INTERACCION (CU-2 Analizar_Tabla, Escenario: AnalizarCadena)

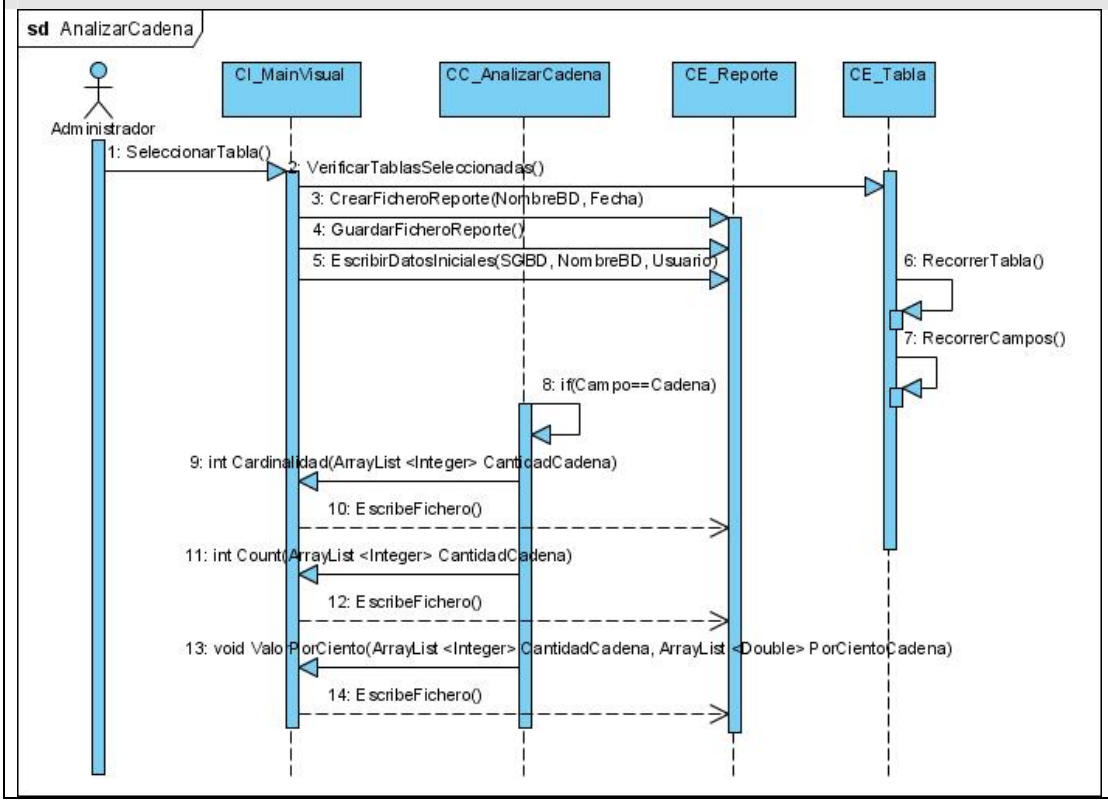


DIAGRAMA DE INTERACCION (CU-2 Analizar_Tabla, Escenario: AnalizarReal)

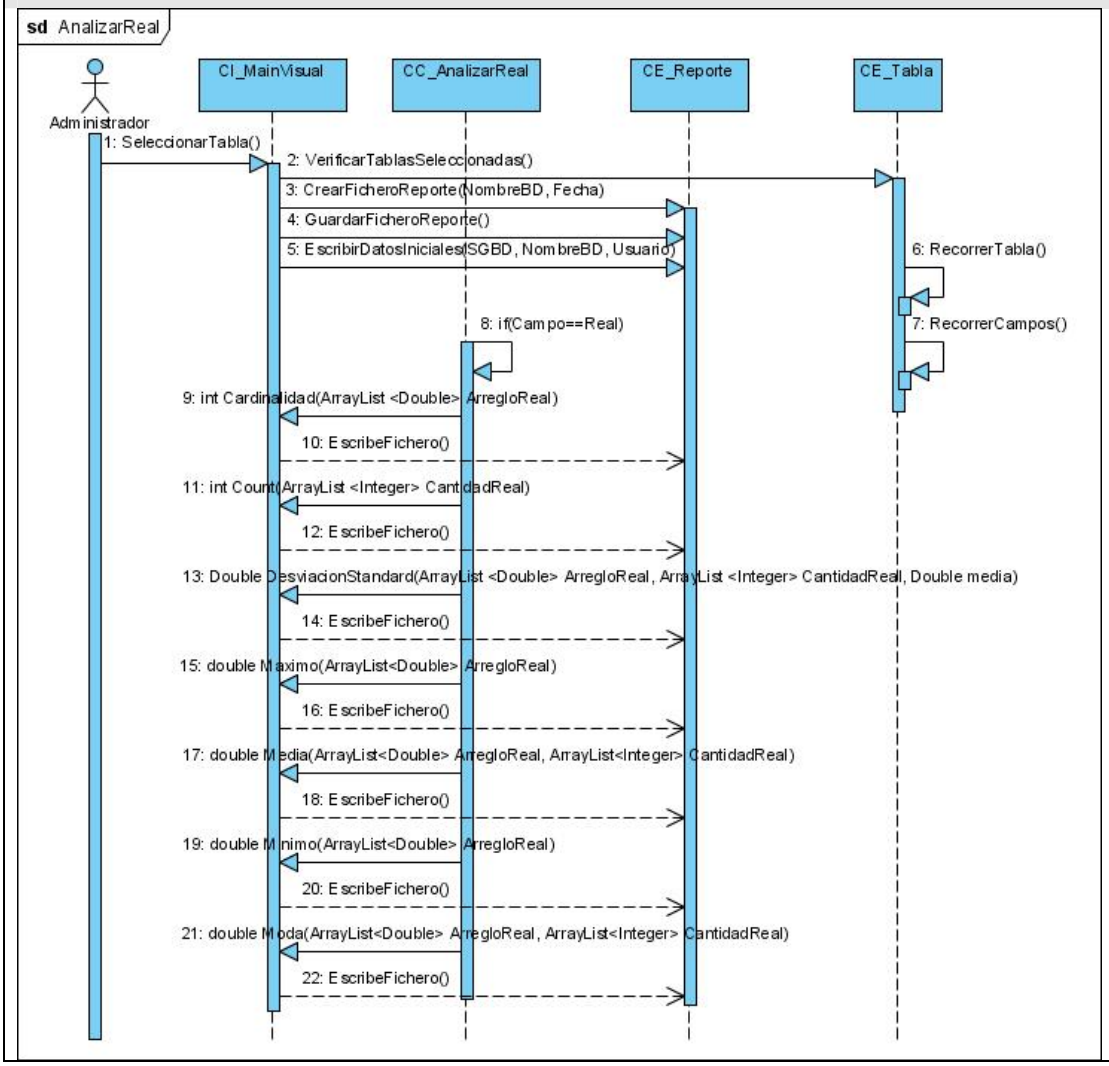


DIAGRAMA DE INTERACCION (CU-2 Analizar_Tabla, Escenario: AnalizarEntero)

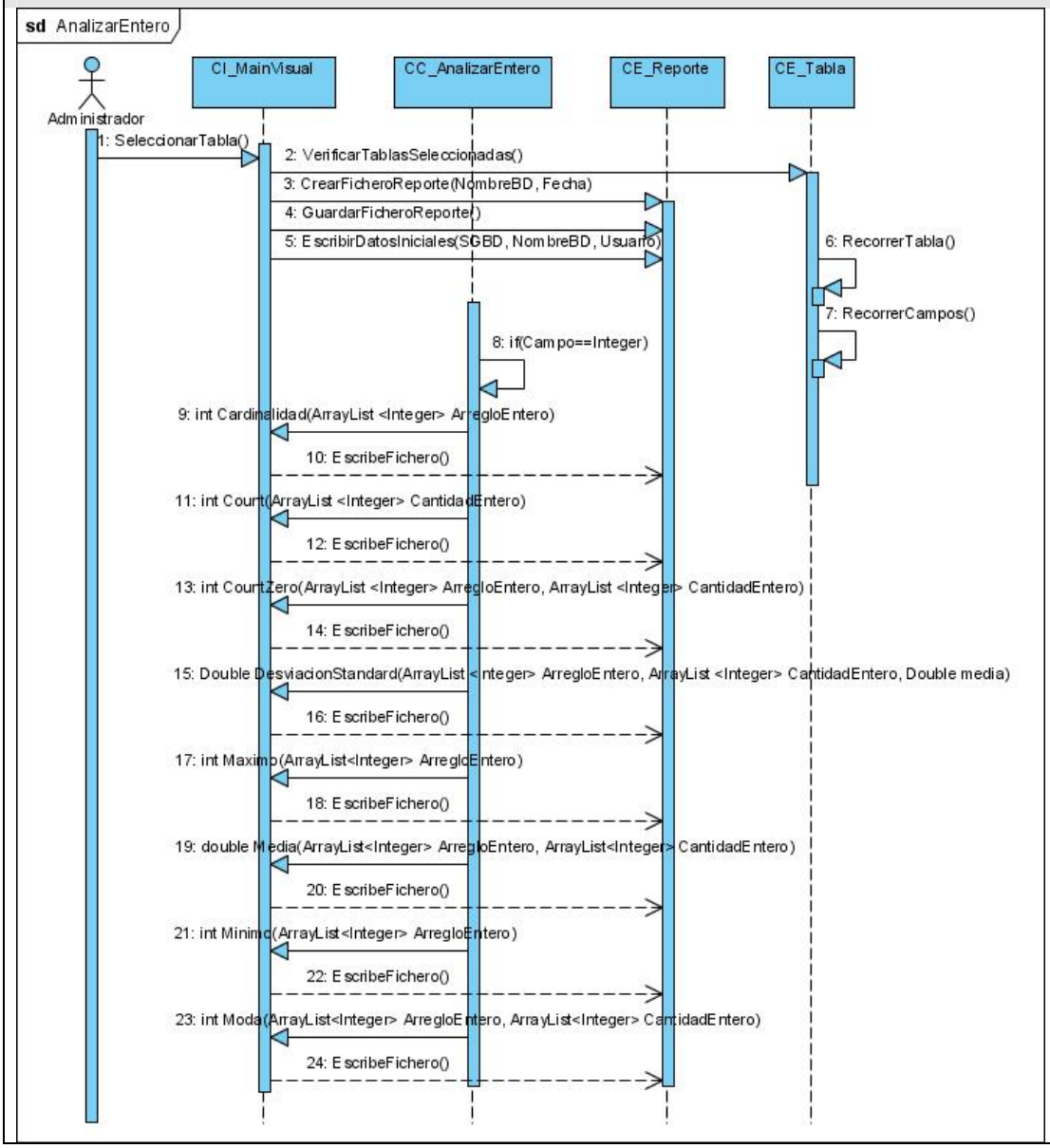
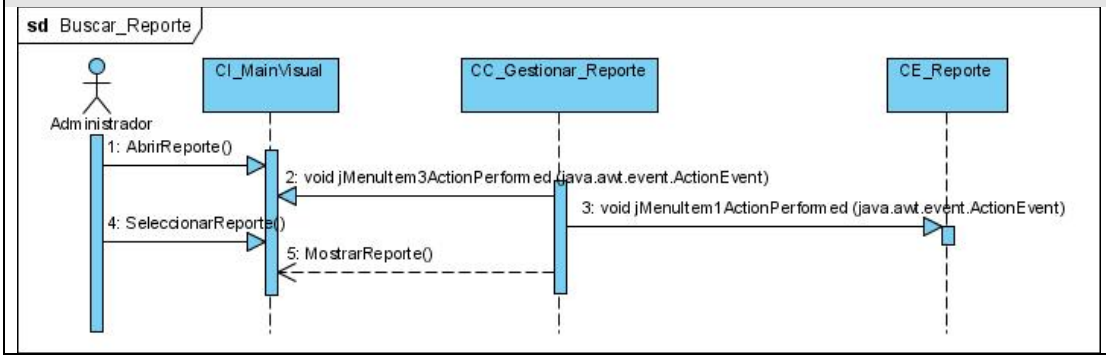


DIAGRAMA DE INTERACCION (CU-3 Buscar_Reporte)



Descripción de clases

A continuación se realiza la descripción detallada de las clases que conforman el sistema, especificando los atributos y las operaciones que las mismas realizan.

Nombre: AnalisisCadena	
Tipo de clase: Controladora	
Atributo	**Tipo**
Para cada responsabilidad:	
Nombre:	int Cardinalidad(ArrayList <Integer> CantidadCadena)
Descripción:	Recibe como parámetro el arreglo de cadenas y devuelve la cantidad de elementos del arreglo.
Nombre:	int Count(ArrayList <Integer> CantidadCadena)
Descripción:	Recibe como parámetro el arreglo de enteros con la cantidad de ocurrencias y devuelve la suma de los elementos de ese arreglo.
Nombre:	void ValorPorCiento(ArrayList <Integer> CantidadCadena, ArrayList <Double> PorCientoCadena)
Descripción:	Recibe como parámetros el arreglo de enteros con la cantidad de ocurrencias y un arreglo de reales que inicialmente está vacío, este método no devuelve nada, modifica por referencia el arreglo de reales pues lo llena con el porciento de ocurrencia de cada elemento según la cantidad de repeticiones de cada elemento.

Nombre: AnalisisEntero	
Tipo de clase: Controladora	
Atributo	**Tipo**
Para cada responsabilidad:	
Nombre:	int Cardinalidad(ArrayList <Integer> ArregloEntero)
Descripción:	Recibe como parámetro el arreglo de enteros que almacena los elementos de la BD y devuelve la cantidad de elementos diferentes del arreglo.
Nombre:	int Count(ArrayList <Integer> CantidadEntero)
Descripción:	Recibe como parámetro el arreglo de enteros con la cantidad de ocurrencias y devuelve la suma de los elementos de ese arreglo.

Nombre:	int CountZero(ArrayList <Integer> ArregloEntero, ArrayList <Integer> CantidadEntero)
Descripción:	Recibe como parámetro los arreglos de enteros y devuelve la cantidad de elementos que son cero.
Nombre:	Double DesviacionStandard(ArrayList <Integer> ArregloEntero, ArrayList <Integer> CantidadEntero, Double media)
Descripción:	Recibe como parámetro los arreglos de enteros y la media, devuelve el cálculo de la desviación estándar de los elementos del primer arreglo de enteros.
Nombre:	int Maximo(ArrayList <Integer> ArregloEntero)
Descripción:	Recibe como parámetro el arreglo de enteros que almacena los elementos de la BD y devuelve el mayor elemento del arreglo.
Nombre:	double Media(ArrayList <Integer> ArregloEntero, ArrayList <Integer> CantidadEntero)
Descripción:	Recibe como parámetros los arreglos de enteros, devuelve el cálculo de la media de los elementos del primer arreglo.
Nombre:	int Minimo(ArrayList <Integer> ArregloEntero)
Descripción:	Recibe como parámetro el arreglo de enteros que almacena los elementos de la BD y devuelve el menor elemento del arreglo.
Nombre:	int Moda(ArrayList <Integer> ArregloEntero, ArrayList <Integer> CantidadEntero)
Descripción:	Recibe como parámetros los arreglos de enteros, devuelve el elemento del primer arreglo que más se repite.

Nombre: AnalisisReal	
Tipo de clase: Controladora	
Atributo	**Tipo**
Para cada responsabilidad:	
Nombre:	int Cardinalidad(ArrayList <Double> ArregloReal)
Descripción:	Recibe como parámetro el arreglo de reales y devuelve la cantidad de elementos del arreglo.
Nombre:	int Count(ArrayList <Integer> CantidadReal)
Descripción:	Recibe como parámetro el arreglo de enteros y devuelve la suma de los elementos de ese arreglo.
Nombre:	Double DesviacionStandard(ArrayList <Double> ArregloReal, ArrayList

	<Integer> CantidadReal, Double media)
Descripción:	Recibe como parámetro el arreglo de real, el arreglo de enteros y la media, devuelve el cálculo de la desviación estándar de los elementos del arreglo de reales.
Nombre:	double Maximo(ArrayList<Double> ArregloReal)
Descripción:	Recibe como parámetro el arreglo de reales y devuelve el mayor elemento del arreglo.
Nombre:	double Media(ArrayList <Double> ArregloReal, ArrayList <Integer> CantidadReal)
Descripción:	Recibe como parámetros el arreglo de reales y el arreglo de enteros, devuelve el cálculo de la media de los elementos del primer arreglo.
Nombre:	double Minimo(ArrayList <Double> ArregloReal)
Descripción:	Recibe como parámetro el arreglo de reales y devuelve el menor elemento del arreglo.
Nombre:	double Moda(ArrayList <Double> ArregloReal, ArrayList <Integer> CantidadReal)
Descripción:	Recibe como parámetros el arreglo de reales y el arreglo de enteros, devuelve el elemento del arreglo de reales que más se repite.

Nombre: MainVisual	
Tipo de clase: Interfaz	
Atributo	**Tipo**
MiUsuario	UserPasswd
ArregloCheckBox	JCheckBox
miResultset	Resultset
miConexion	Connection
ArregloEntero	Integer
ArregloCantidad	Integer
ArregloCadena	String
ArregloReal	Double
Para cada responsabilidad:	
Nombre:	jButton2ActionPerformed(java.awt.event.ActionEvent evt)
Descripción:	Método que se ejecuta al dar clic sobre el botón “Analizar” y que a cada elemento del jList que esté seleccionado (que corresponde a una tabla de la BD) le hace el análisis campo a campo en dependencia del tipo de dato que

	almacene cada campo, si no hay ningún elemento del jList seleccionado se muestra un mensaje para que se seleccione al menos uno.
Nombre:	jButton1ActionPerformed(java.awt.event.ActionEvent evt)
Descripción:	Método que se ejecuta al dar clic sobre el botón “Conectar”, donde se chequea que esté seleccionado el gestor de BD al cual se va a conectar, y que los jTextField donde va la dirección donde se encuentra el servidor y el nombre de la BD no estén en blanco, si lo anterior no se cumple se lanza una excepción, luego se trata de conectar (con el nombre de usuario y la contraseña) si no se puede conectar se lanza una excepción, si se conecta se actualiza el jList con la lista de tablas que tiene la BD a la cual se conectó.
Nombre:	jCheckBox1ActionPerformed(java.awt.event.ActionEvent evt)
Descripción:	Método que se ejecuta al dar clic en el checkbox “Seleccionar todas las tablas”, que cuando está marcado selecciona todos los elementos del jList, y si se desmarca des-selecciona todos los elementos del jList.
Nombre:	jListValueChanged(javax.swing.event.ListSelectionEvent evt)
Descripción:	Método que se ejecuta al dar clic sobre un elemento dentro del jList y que modifica al checkbox que está marcado cuando todos los elementos del jList están seleccionados y desmarcado en caso contrario
Nombre:	jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt)
Descripción:	Método que se ejecuta al dar clic en el menú “Analizar BD” y que inicializa los componentes visuales para realizar un nuevo análisis.
Nombre:	jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt)
Descripción:	Método que se ejecuta al dar clic en el menú “Salir” y que cierra la aplicación.
Nombre:	jMenuItem3ActionPerformed(java.awt.event.ActionEvent evt)
Descripción:	Método que se ejecuta al dar clic en el menú “Abrir Reporte” y que abre la ventana donde se muestra un reporte.
Nombre:	void main(string args[])
Descripción:	Es el punto de entrada al proyecto.
Nombre:	MainVisual ()
Descripción:	Constructor de la clase, que inicializa los componentes visuales.

Nombre: UserPasswd	
Tipo de clase: Entidad	
Atributo	**Tipo**

Passwd	String
Usuario	String
Para cada responsabilidad:	
Nombre:	UserPasswd()
Descripción:	Constructor de la clase, que crea un objeto de tipo UserPasswd con sus atributos vacíos.
Nombre:	String getPasswd()
Descripción:	Método que devuelve el valor del atributo Passwd.
Nombre:	String getUsuario()
Descripción:	Método que devuelve el valor del atributo Usuario.
Nombre:	void setPasswd(String Passwd)
Descripción:	Método que recibe como parámetro una cadena y que modifica el valor del atributo Passwd.
Nombre:	void setUsuario(String Usuario)
Descripción:	Método que recibe como parámetro una cadena y que modifica el valor del atributo Usuario.

Nombre: ReporteVisual	
Tipo de clase: Entidad	
Atributo	**Tipo**
Para cada responsabilidad:	
Nombre:	ReporteVisual()
Descripción:	Método que inicializa todos los componentes de la clase.
Nombre:	void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt)
Descripción:	Método donde se busca un fichero texto y luego se muestra en una componente visual.
Nombre:	void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt)
Descripción:	Método que cierra la ventana.

Tratamiento de errores

La confiabilidad del sistema constituye un punto fundamental en la interacción con el usuario, para ello se lleva a cabo el tratamiento de errores para los posibles fallos que se pueden encontrar en el sistema durante el trabajo con la aplicación. Siempre que ocurra un error o se lance una excepción el sistema notificará inmediatamente al usuario y le propondrá las posibles soluciones. La

notificación de los errores se realiza mediante ventanas de error que genera el sistema, de tal modo que el usuario entienda la naturaleza del error y pueda corregirlo.
Mediante una barra de progreso el usuario podrá saber cuando el proceso ha sido completado, y así poder continuar.

Seguridad

Para una mayor seguridad el sistema cuenta con un proceso para borrar automáticamente las contraseñas después de haber accedido a la base de datos.

Interfaz.

La interfaz del sistema es sencilla y fácil de entender. En la ventana principal cuenta con dos checkbox para seleccionar el gestor de base de datos al cual se desea conectar; también posee cuatro cuadros de texto, los dos primeros para especificar la dirección del servidor y el nombre de la base de datos, el tercero para poner el nombre de usuario y el cuarto para introducir el password si es necesario, por último tiene un botón que es el que da la instrucción de conectar a la base de datos.
Cuando el botón de conectar es presionado, el sistema verifica que la información anterior es válida y se conecta a la base de datos, mostrando todas las tablas que esta posee, dando la posibilidad de seleccionar cualquier número de tablas para analizar, instrucción que es dada al sistema por otro botón nombrado Analizar. Mientras el sistema va analizando las tablas aparece en la parte inferior de la aplicación una barra de progreso que indica el estado de la operación, así cuando esté al 100% significará que ya terminó de ejecutarse.
La ventana principal tiene además en la parte superior un menú Archivo y otro Ayuda, el segundo con un manual de ayuda al usuario, y el primero con las opciones de Salir y Abrir Reporte. Al seleccionar Abrir Reporte se abrirá una nueva ventana que contiene un campo de texto y un menú en el cual se selecciona la opción Buscar Reporte para buscar un reporte que se encuentre en el disco duro de la máquina.

Concepción de la ayuda.

El sistema contará con una ayuda que se mantendrá accesible como parte del menú principal de la aplicación, la cual tendrá como objetivo ofrecer al usuario la información sobre como interactuar con el sistema, así como datos generales acerca de la creación y objetivos que persigue la aplicación en cuestión además de una explicación detallada de cada funcionalidad permitiéndole al usuario una mejor comprensión y facilitando así su trabajo.

3.3 Implementación

En el flujo de trabajo de implementación el propósito fundamental es dejar el producto listo en su versión inicial, donde se empieza con el resultado del diseño y se implementa el sistema en término de componentes describiendo así cómo estos se organizan de acuerdo a los nodos del despliegue.
Debido a la simplicidad de la aplicación así como a sus características que la definen como una aplicación de escritorio, el sistema estará montado en la misma máquina cliente del usuario, donde se llevarán a cabo todas las operaciones y se guardarán igualmente sus resultados

.

Diagrama de despliegue

El diagrama de despliegue muestra las relaciones físicas entre los componentes de hardware y software en el sistema final, es decir, la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes visuales.
La máquina cliente estará conectada a servidores de bases de datos MySQL y PostgreSQL mediante una conexión TCP/IP.

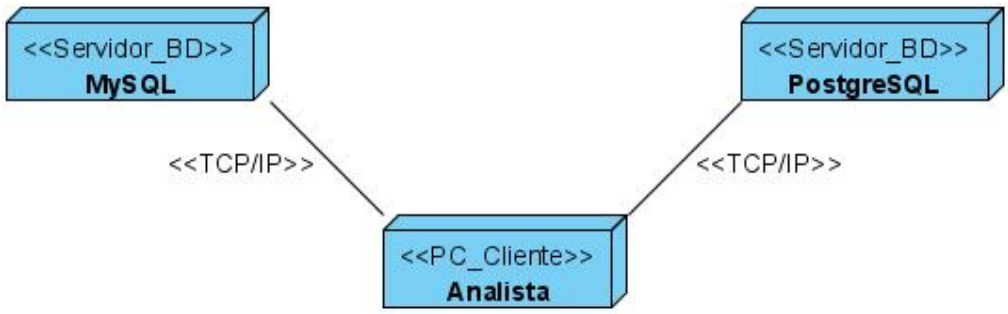
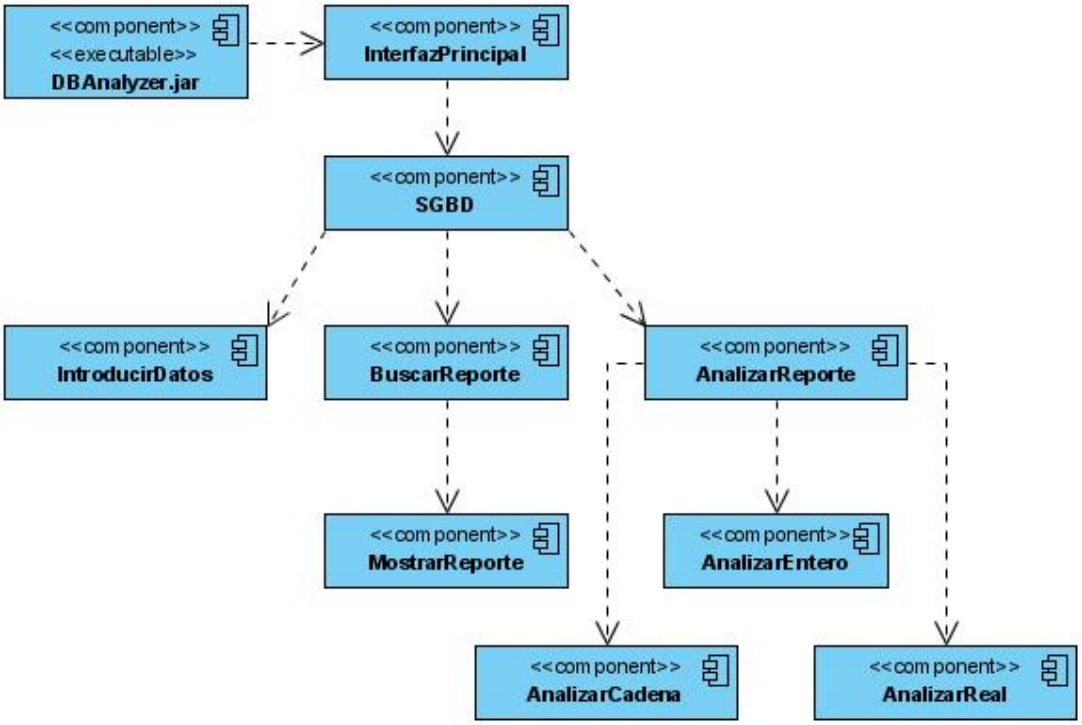


Diagrama de componentes.

Los diagramas de componentes son usados para estructurar el modelo de implementación en términos de subsistemas de implementación y mostrar las relaciones entre los elementos de implementación. Es un grafo de componentes unidos a través de relaciones que pueden ser de compilación o de ejecución, y además se pueden representar las interfaces de esos componentes. Es una forma de representar una vista estática del sistema, que representa la organización y dependencia que existe entre los componentes físicos que se necesitan para ejecutar la aplicación.


<<component>>
<<executable>>
DBAnalyzer.jar
<<component>>
InterfazPrincipal
<<component>>
SGBD
<<component>>
IntroducirDatos
<<component>>
BuscarReporte
<<component>>
AnalizarReporte
<<component>>
MostrarReporte
<<component>>
AnalizarEntero
<<component>>
AnalizarCadena
<<component>>
AnalizarReal

Conclusiones

El análisis de las bases de datos se ha convertido en una tarea de vital importancia en nuestros días, el aumento del volumen de información almacenada hace que se de más importancia a la automatización de los procesos de corrección y detección de errores.

Con este trabajo se presenta una herramienta capaz de analizar estadísticamente la información almacenada en las bases de datos construidas en PostgreSQL y MySQL. Pudiendo así concluir que las tareas y objetivos propuestos al inicio de la investigación han sido cumplidos.

Recomendaciones

De forma general los objetivos trazados en la apertura de este trabajo han sido alcanzados, pero, a lo largo del proceso de desarrollo, ha quedado claro que la propuesta es sólo la primera etapa de un proyecto que puede ser mucho más ambicioso. Por tanto se proponen las siguientes recomendaciones:

- Extender el desarrollo de la aplicación adicionándole nuevas funcionalidades.
- Ampliar el análisis a otros gestores de bases de datos.
- Continuar la investigación de este tema para poder construir una herramienta eficaz en el proceso de detección y corrección de errores.

Referencias Bibliográficas

ABITEBOUL, S.; T. MILO*, et al.* Tools for Data Translation and Integration *Bulletin of the Technical Committee on Data Engineering*, 1999, 22(1): 3-8.

AHO, A. V. *Principles of Compiler Design*, Addison-Wesley Publishing Company, 1979.

CHAPMAN, A. D. *Principles and Methods of DataCleaning- Primary Species and Species Occurrence Data*. *Report of the Global Biodiversity Information Facility*. Copenhagen, 2005a.

---. *Principles of Data Quality*. *Report for the Global Biodiversity Information Facility*. Copenhagen, 2005b.

ENGLISH, L. P. *Improving Data Warehouse and Business Information Quality: Methods for Reducing Costs and Increasing Profits*. New York: Wiley, 1999.

GALHARDAS, H.; D. SHASHA*, et al. AJAX: An extensible data cleaning tool*. *Proceedings of the ACM SIGMOD on Management of Data*. Dallas, TX, USA, 2000.

---. *Declarative data cleaning: Language model and algorithms*. *Proceedings of the 27th VLDB Conference*. Roma, Italy, 2001.

GREENFIELD, L. *An (Informal) Taxonomy of Data Warehouse. Data Errors*, 2000.

HAN, J. and M. KAMBER. *Data Mining Concepts and techniques*. Simon Fraser University, Morgan Kaufman Publishers, 2000.

HERNÁNDEZ, M. A. *The merge/purge problem for large databases*. *Proceedings of the ACM SIGMOD Conference*, 1995.

LEE, M. L. and W. L. LOW. *Intelliclean: A knowledge- based inteligent data cleaner*. *Proceedings of the ACM SIGMOD*. Boston, USA, 2000.

LOW, W. L. A knowledge- based approachfor duplicate elimination in data cleaning *Information Systems*, 2001: pp.585-606.

MALETIC, J. I. and A. MARCUS. *Automated Identification of Errors in DataSets*. Memphis, The University of Memphis, 2002.

---. *Data Cleansing: Beyond Integrity Analysis*. *Proceedings of the Conference on Information Quality*, 2000a.

---. *Data Cleansing: Beyond Integrity Analysis. Information Quality (IQ) 2000*, The University of Memphis, 2000b.

MARMOL, D. *Determinacion de una taxonomía de errores en los sistemas operacionales de nuestro entorno*. Villa Clara, Universidad Central de las Villas, 2005.

MAYOL, E. A Survey of Current Methods for Integrity Cnstraint Maintenance and View Updating *ER Workshops*, 1999: 62-73.

PADRON, L. *Estudio de Herramientas para limpiar Direcciones Postales*. Villa Clara, Universidad Central de las Villas, 2006.

QUASS, D. *A Framework for Research in Data Cleaning*, Brinham Young University, 1999.

RAMAN, V. *Potter's Wheel: An Interactive Framework for Data transformation and cleaning*. *Proceedings of the 27th VLDB Conference* Roma, Italy, 2001.

REDMAN, T. C. *Data Quality: The Field Guide*, Boston: MA: Digital Press, 2001.

---. *The Impact of Poor Data Quality on the Typical Enterprise*. *Communications of the ACM*, 1998. pp.79-82.

SATTLER, K. *A Data Preparation Framework based on a Multidatabase Languaje*. *International Database Engineering Applications Symposium (IDEAS)*. Grenoble, France, 2001.

SATTLER, K. U. and G. SAAKE. *Adding Conflict Resolution Features to a Query Language for Database Federations*. *Proc 3 nd Int Workshop on Engineering Federated Information Systems*. Dublin, Ireland, 2000.

STRONG, D. M.; Y. W. LEE*, et al. Communications of the ACM*, 1997. 40: 103-110.

VASSILIADIS, P.; S. SKIADOPOULOS*, et al.* ARKTOS: towards the modeling, design control and execution of ETL processes *Information Systems*, 2001: pp.537-561.

WANG, R. Y. and D. M. STRONG Journal of Management Information Systems (JMIS), 1996, 12(4): 5.

Bibliografía

(CABALLERO 2007-2008) CABALLERO, D. I. Introduccion a la Calidad de los Datos y de la Información. en: Calidad y Medición de Sistemas de Información. Universidad de Castilla La Mancha, 2007-2008.p.

(CASARES) CASARES, C. Data Warehousing. http://www.programacion.com/bbdd/tutorial/warehouse/

(COHEN et al. 2003) COHEN, W. W.; P. RAVIKUMAR, et al. A Comparison of String Distance Metrics for Name-Matching Tasks, 2003.

(COLE et al. 1996) COLE, R. A.; J. MARIANI, et al., Eds. Survey of the State of the Art in Human Language Technology, 1996. http://cslu.cse.ogi.edu/HLTsurvey/HLTsurvey.html

(CORTIZO) CORTIZO, J. C. Preprocesado de Datos,

(DACCACH 1997-2000) DACCACH, J. C. Limpieza de Datos (data Scrubbing o Data Cleansing), 1997-2000. [Disponible en: http://www.gestiopolis.com/delta/term/TER245.html

(PADRÓN 2006) PADRÓN, L. Almacenes de datos: importancia de la estandarización de las direcciones para las empresas de hoy en día, 2006. [Disponible en: http://www.monografias.com/trabajos31/almacenes-datos/almacenes-datos.shtml

(PADRÓN 2006) --- Almacenes de datos: importancia en el estandar, 2006. http://www.mailxmail.com/curso/empresa/almacenesdedatos/capitulo7.htm

(PADRÓN 2006) ---. Similitud entre cadenas y estadarización de direcciones postales, ilustrados.com, 2006. [Disponible en: http://www.ilustrados.com/publicaciones/EEuVAAVupVgtCqaRMG.php

(POCO et al. 2006) POCO, J. L.; F. INCAHUANACO, et al. Modelo de un gestor de búsquedas por similitud, 2006.

(RAHM and DO) RAHM, E. and H. H. DO. Data Cleaning: Problems and Current Approaches. Germany, University of Leipzig.http://dbs.uni-leipzig.de

(ROSA et al.) ROSA, F. D. L.; S. POZO, et al. Análisis y Visualización de Comunidades Científicas con Información Extraída de la Web,

(ROSSEL 2003) ROSSEL, R. Mejoramiento de la Calidad de los Datos en un Data Warehouse, 2003.

(WINKLER 2006) WINKLER, W. E. Overview or Record Linkage and Current Research Directions. Washington, 2006.

Glosario de términos.

Anomalía: Discrepancia de una regla o de un uso.

Campo semántico: conjunto de unidades léxicas de una lengua que comprende términos ligados entre si por referirse a un mismo orden de realidades o ideas.

Dominio de los datos: Orden determinado de ideas materias o conocimientos.

Intrínseco: Íntimo, esencial.

Léxico: Perteneciente o relativo al léxico (II Vocabulario de un idioma o una región). 2. Diccionario de una lengua. 3. Vocabulario, conjunto de palabras de un idioma, o de las que pertenecen al uso de una región, a una actividad determinada, a un campo semántico dado, etc.

Semántica: Perteneciente o relativo a la significación de las palabras. 2. Estudio del significado de los signos lingüísticos y de sus combinaciones, desde un punto de vista sincrónico a diacrónico.

Servidor de Bases de Datos: Un servidor de bases de datos es un sistema que proporciona servicios de gestión, administración y protección de la información (datos) a través de conexiones de red, gobernadas por unos protocolos definidos y a los que acceden los usuarios a través de aplicaciones clientes (bien sean herramientas del propio sistema como aplicaciones de terceros).

Sintáctico: Perteneciente o relativo a la sintaxis.

Sintaxis: Parte de la gramática que enseña a coordinar y unir las palabras para formar las oraciones y expresar conceptos.

Superfluo: No necesario, que está de más.

Taxonomía: Ciencia que trata de los principios, métodos y fines de la clasificación. Se aplica para la organización jerarquizada y sistemática. 2. Acción y efecto de clasificar.