UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 6

EXTENSIÓN DE ALGORITMOS DE REGLAS DE ASOCIACIÓN PARA MINERÍA DE DATOS EN POSTGRESQL

"TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS"

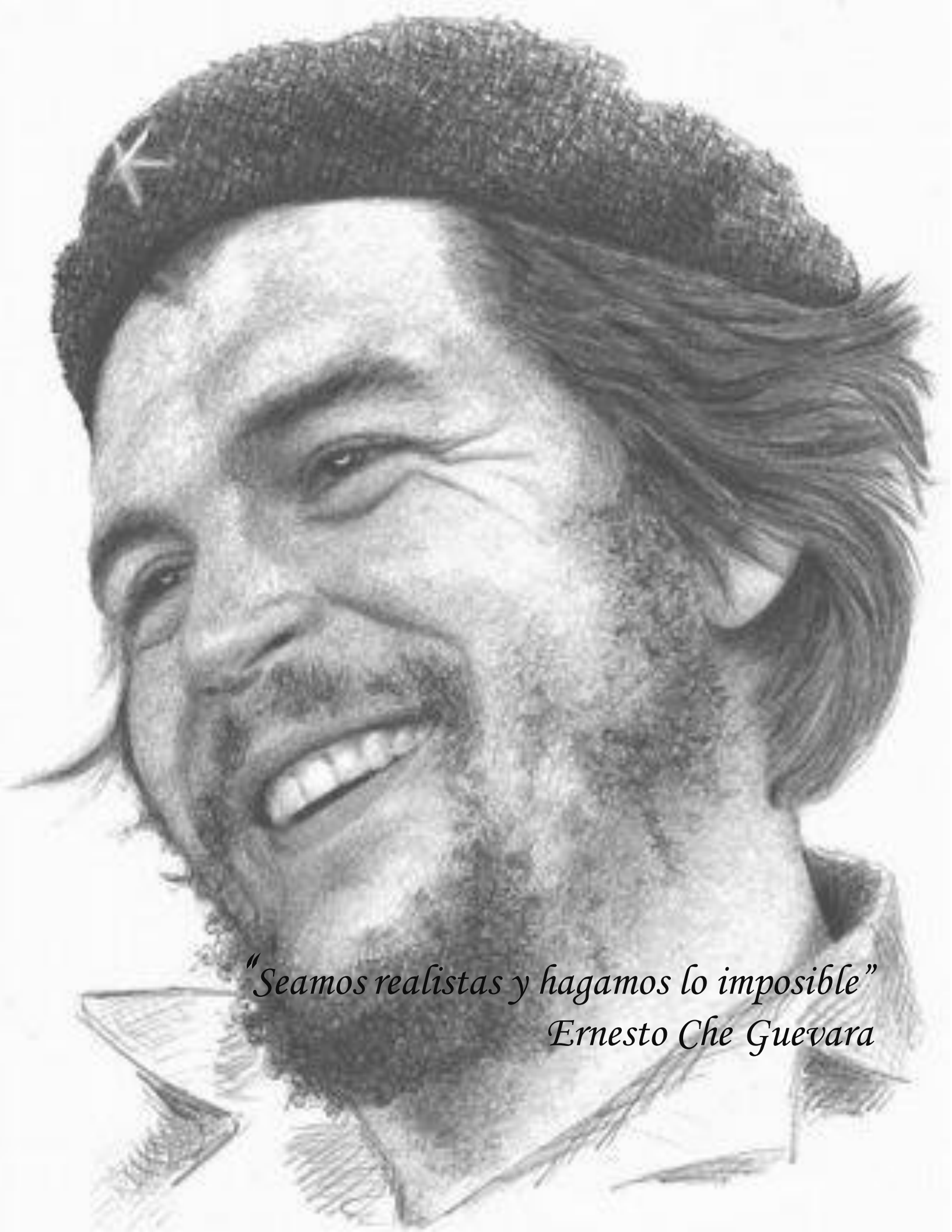
Autor: Audrey Cordero Sánchez

Tutores: MSc. Yadira Robles Aranda

Ing. Juan Manuel Ruiz Godoy

La Habana, 2013

"Año 55 de la Revolución"


"Seamos realistas y hagamos lo imposible"
Ernesto Che Guevara

DECLARACIÓN DE AUTORÍA

Declaro que soy la única autora de este trabajo y autorizo a la Facultad 6 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Audrey Cordero Sánchez

_____

Firma del autor

MSc. Yadira Robles Aranda

_____

Firma del tutor

Ing. Juan Manuel Ruiz Godoy

_____

Firma del tutor

DATOS DE CONTACTOS

Tutora:

MSc. Yadira Robles Aranda

Universidad de las Ciencias Informáticas, La Habana, Cuba

Correo electrónico: yrobles@uci.cu

Tutor:

Ing. Juan Manuel Ruiz Godoy

Universidad de las Ciencias Informáticas, La Habana, Cuba

Correo electrónico: jmruiz@uci.cu

AGRADECIMIENTOS

Mis primeros agradecimientos van para mis padres a los que admiro y respeto. Gracias por su dedicación día a día, por guiarme por el mejor camino, por estar siempre a mi lado y sobre todo por confiar en mí.

A mi mamá: por saber ser amiga y madre, por estar disponible siempre para mí, porque en todo momento puedo contar con ella, por saber entenderme cuando nadie más lo puede hacer, por su infinito amor y sobre todo por la confianza que siempre ha depositado en mí.

A mi papá quererme como soy y por sus consejos, que me han sido muy útiles durante este tiempo en la universidad.

A Raylith: por estar apoyándome en los momentos más difíciles, por ser amiga y compañera, por haber estado a mi lado en el transcurso más difícil en esta etapa en la universidad, porque pude contar contigo cada vez que necesité tú ayuda. Gracias por todo.

A Osbel y Yainelis pues han sido para mí como mis hermanos, por ayudarme en los momentos difíciles y en los que más los he necesitado, pues siempre han estado ahí dispuestos a ayudarme. Gracias por hacerme saber que puedo contar con ustedes siempre.

A Carlos pues fue quien me alentó cuando pensé que no lo iba a lograr, por estar apoyándome y dándome las fuerzas para poder estudiar, por brindarme su mano cada vez que me caía y ayudarme a levantar.

A Aliuva por estar atenta de mí en el tiempo que estuvo conmigo aquí en la universidad, por ayudarme cuando no sabía qué hacer, por estar siempre atenta de mis problemas y buscando la manera de solucionarlos.

A Yanet que me encanta joderla y ha sabido ganarse un lugar bien lindo en mi corazón, eres una niña muy dulce.

A la Iraidis y Dayana por aguantar cada vez que las he ido a molestar y saber sobrellevarme.

A Leidanys por siempre estar atenta de mis estudios y tratando de que salga de todo sin problemas.

A Marcos, Osman, Raúl, Wilson, Darlys y Hack por siempre ayudarme a con mis dudas y estudiar conmigo.

A los amigos de Raylith pues cada vez que los necesitaba siempre estuvieron dispuestos a ayudarme.

A mis tutores, Yadira y Manolo pues la verdad que me ayudaron mucho y siempre estuvieron apoyándome durante el trascurso de este trabajo.

A todos los que de una manera u otra han formado parte de mi superación todos estos años en la universidad.

DEDICATORIA

Este trabajo que es parte de todo lo que me va a preparar el futuro, va dedicado a mis abuelos, a Pullo que no se encuentra a mi lado, pero para mí es como si lo estuviera pues siempre va a estar presente en mi corazón. A mi abuela Olga por el amor que me ha dado todos estos años, por siempre estar pensando en mí. A ustedes van dedicados todos mis logros, no solo estos, todos los que a partir de este momento sea capaz de alcanzar.

RESUMEN

El desarrollo de las tecnologías y la informatización de las empresas ha incrementado el cúmulo de información almacenada en las bases de datos dificultando el proceso de análisis de éstas manualmente, por lo cual es necesario utilizar técnicas como la Minería de Datos que faciliten obtener conocimientos a partir de los datos recopilados.

En la presente investigación se analizó de las técnicas de Minería de Datos la de reglas de asociación para integrar varios de sus algoritmos al Sistema Gestor de Base de Datos (SGBD) PostgreSQL, debido a las deficiencias de las herramientas libres existentes para realizar el análisis de la información. En ella se desarrollaron mecanismos para optimizar el rendimiento de los algoritmos implementados con el objetivo de aprovechar las ventajas de PostgreSQL.

Además se comprobó la solución de la investigación mediante la aplicación de las técnicas al almacén de datos de ensayos clínicos Racotumumab del Centro de Inmunología Molecular y se realizó un experimento para comprobar que los algoritmos integrados al gestor permiten aprovechar las potencialidades de PostgreSQL para disminuir el tiempo del análisis de los datos.

Palabras claves: Minería de Datos, Postgre SQL, Reglas de Asociación, Sistema Gestor de Base de Datos

TABLA DE CONTENIDO

Tabla de Contenido

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Con el acelerado avance de la sociedad moderna en el siglo XXI y el desarrollo empresarial alcanzado en esta etapa, surge la necesidad de almacenar los datos, interactuar con estos y clasificarlos en pequeños grupos que describan sus características principales, basándose en la similitud o diferencia entre ellos. Un gran porcentaje de los datos generados representan “hechos” que diariamente se están registrando, tales como: transacciones financieras, operaciones de compra y venta, préstamos o devoluciones y movimientos de almacén.

Convertir los datos en información limpia, útil para el análisis y el apoyo en la toma de decisiones, es una tarea compleja, de ahí la creciente necesidad de una nueva generación de técnicas computacionales y sistemas informáticos para apoyar la extracción de conocimiento útil. Estas técnicas y sistemas son el centro del emergente y dinámico campo de investigación denominado: Descubrimiento de conocimiento en Bases de Datos (KDD, en inglés Knowledge Discovery in Data Bases). Dentro de sus etapas hay una que resulta fundamental, la Minería de Datos (DM, en inglés Data Mining).

La Minería de Datos es una de las etapas de lo que se ha venido llamando el proceso de extracción de conocimientos a partir de los datos. Esta incluye diferentes tipos de técnicas como las redes neuronales artificiales, los árboles de decisión y las reglas de asociación, estas son formas de representación del conocimiento utilizadas en sistemas de expertos.

La Minería de Datos es una técnica fundamental para la toma de decisiones. El proceso de aprendizaje de los datos juega un papel muy importante en muchas áreas de la ciencia, las finanzas y la industria. Este proceso consta de varias fases e incorpora diferentes técnicas de Aprendizaje Automático, la Estadística, las Bases de Datos, los Sistemas de Toma de Decisiones, la Inteligencia Artificial y otras áreas de la informática y de la gestión de información, dónde las entidades o empresas han de minimizar los riesgos en la toma de decisiones estratégicas.

Las herramientas de Minería de Datos son utilizadas para resolver situaciones donde el volumen de datos es muy grande o complejo por la cantidad de variables que se manipulan. Normalmente, estos patrones no se pueden detectar mediante la exploración tradicional de los datos porque las relaciones son

demasiado complejas o porque hay demasiados datos. Por lo que es necesario hacer uso de los diferentes paquetes de software para aplicar la Minería de Datos.

La Universidad de las Ciencias Informáticas (UCI), se adentra cada vez más en la producción de software empresarial con el reto de convertir a la industria de software cubana en un renglón fundamental de la economía del país. Son varias empresas las que solicitan los servicios de la UCI con vistas a informatizar y posteriormente analizar todo un cúmulo de datos que poseen las mismas. Estos datos crecen día a día y resulta de vital importancia conocer cuál es el conocimiento oculto en ellos de manera que sea posible encontrar patrones y asociaciones o correlaciones entre los datos que se almacenan y que pueden ser usados para realizar predicciones a través de técnicas de reglas de asociación.

Para la aplicación de estas técnicas de reglas de asociación existen numerosas herramientas tanto libres como propietarias que permiten aplicarlas a grandes volúmenes de datos, ejemplo de las herramientas libres son Yale/Rapid Miner y Weka, como propietarias se encuentran Oracle y SQL Server. En el caso de las herramientas libres mencionadas anteriormente, las mismas necesitan conectarse al Sistema Gestor de Base de Datos y si en estos existe un gran volumen de datos para analizar, el proceso se vuelve engorroso y lento. Por su parte Oracle y SQL Server han desarrollado módulos que tienen incluido las técnicas de minería de datos, lo cual permite ganar en rapidez en los tiempos de respuesta ya que no sería necesario transformar datos desde un archivo el cual puede no ser reconocido por el gestor de base de datos. Por otro lado con ello se evitaría tener que contar con personal calificado que domine aquellas herramientas que permitan la transformación de los datos para que sean entendibles al gestor de base de datos usado. A pesar de contar con la implementación en el propio gestor, de los algoritmos de minería de datos, son herramientas cuyas licencias de uso y el soporte las hacen altamente costosas. A todo esto se le suma que PostgreSQL como gestor de base de datos no presenta implementación alguna de algoritmos de reglas de asociación y resultaría esto una desventaja dada las potencialidades que posee el mismo.

Después de analizar la problemática, se identifica como ***problema a resolver:***
¿Cómo lograr la independencia del sistema gestor de Base de Datos PostgreSQL para aplicar la Minería de Datos utilizando reglas de asociación?

Se plantea como ***objeto de estudio*** las técnicas de reglas de asociación enmarcadas en el ***campo de acción*** de las técnicas de reglas de asociación en PostgreSQL.

Para darle solución al problema científico anterior planteado se definió como ***objetivo general*** integrar algoritmos de las técnicas de reglas de asociación al Sistema Gestor de Base de Datos PostgreSQL para analizar el comportamiento de los datos almacenados en las tablas.

Desglosándose en los siguientes ***objetivos específicos:***

1. Analizar las técnicas de reglas de asociación.
2. Implementar los algoritmos de Minería de Datos.
3. Integrar los algoritmos implementados al SGBD PostgreSQL.
4. Validar la solución propuesta.

Para dar cumplimiento a los objetivos específicos se plantearon las siguientes ***tareas de la investigación:***

a) Caracterización de las técnicas de reglas de asociación.
b) Caracterización de las herramientas de Minería de Datos.
c) Análisis de los lenguajes de programación y herramientas para la implementación en PostgreSQL.
d) Implementación de los algoritmos de reglas de asociación.
e) Creación de la extensión Minería de Datos.
f) Incorporación de la extensión creada al SGBD PostgreSQL.
g) Aplicación del diseño de pruebas de caja negra.
h) Validación del correcto funcionamiento de los algoritmos implementados.

El presente documento se encuentra estructurado en tres capítulos.

En el Capítulo 1 se realiza un estudio sobre las herramientas y metodologías usadas en el proceso de Minería de Datos. Se define el lenguaje de programación y la herramienta a utilizar para implementar los algoritmos así como la metodología seleccionada para aplicar el proceso de Minería de Datos. También se seleccionaron los algoritmos a implementar de las reglas de asociación y se presenta una descripción de los mismos.

En el Capítulo 2 se presenta una descripción de los algoritmos implementados incluyendo los pseudocódigos así como una breve descripción de todas las funciones implementadas. Además se muestra como fueron integradas al SGBD PostgreSQL 9.1.

En el Capítulo 3 se realiza la aplicación de las técnicas de Minería de Datos al Mercado de Datos de Ensayos Clínicos del Centro de Inmunología Molecular y se validará haciendo uso de las técnicas de caja negra.

También contiene las conclusiones donde se exponen los principales resultados de la investigación y se proponen recomendaciones para continuar desarrollando el objeto de la investigación. Además se exponen todos los materiales consultados y referenciados, quedando organizados en referencias bibliográficas y la bibliografía según corresponda en cada caso.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

Introducción

En el presente capítulo se abordarán definiciones referentes al tema de la Minería de Datos para llegar a una mejor comprensión de la misma. Se seleccionarán los algoritmos a implementar pertenecientes a las reglas de asociación de la Minería de Datos. Se incluirá un estudio sobre las herramientas y metodologías usadas en el proceso de Minería de Datos con el fin de seleccionar la más adecuada para dar solución al problema. También se realizará un previo análisis de las metodologías de desarrollo de software hasta seleccionar la más adecuada a utilizar. Además se abordará sobre las herramientas de programación y sobre PL/PgSQL, el lenguaje procedural de PostgreSQL.

1.1 MINERÍA DE DATOS

Para que las empresas trabajen eficientemente en su competitividad, se necesita contar con el máximo de información necesaria y presentarla de la forma más fácil de manipular. El Descubrimiento de Conocimientos en Bases de Datos abarca todas aquellas tecnologías que surgen de la capacidad de procesar, analizar y aprovechar la información encubierta en grandes volúmenes de datos. Dentro de las múltiples áreas que se agrupan alrededor del Descubrimiento de Conocimientos en Bases de Datos, surge la Minería de Datos como una de las disciplinas que más influyen en nuestros días dentro del medio de análisis de datos.

El surgimiento de técnicas como la Minería de Datos está asociado con la necesidad de procesar y analizar grandes volúmenes de datos. En los últimos años, la Minería de Datos ha experimentado un gran auge como soporte de la gestión de la información y el conocimiento, así como para el descubrimiento del significado que poseen los grandes volúmenes de datos almacenados. Esta permite explorar y analizar las bases de datos disponibles con el fin de obtener información mediante la consolidación de los datos y conocimientos útiles a la toma de decisiones; además de facilitar la extracción de la información existente en los textos, así como crear sistemas inteligentes capaces de entenderlos. (1) (2)

1.1.1 Definiciones de Minería de Datos

La Minería de Datos consiste en la extracción de información que reside de manera implícita, desconocida o previamente ignorada, que puede ser potencialmente útil, de un conjunto de datos. Se puede considerar a la Minería de Datos como una colección de diferentes técnicas que sirven para inducir el conocimiento e información de una manera estructurada de un gran conjunto de datos. (2)(3)

A continuación se muestran dos conceptos de Minería de Datos:

- ✓ La Minería de Datos no trivial es el proceso de identificación válida, novedosa, potencialmente útil y en última instancia comprensible en los patrones de tiempo. Se utiliza para descubrir patrones y relaciones de datos, con énfasis en las grandes bases de datos de observación. (4)

- ✓ La Minería de Datos es el proceso de descubrir conocimientos interesantes, como patrones, asociaciones, cambios, anomalías y estructuras significativas a partir de grandes cantidades de datos almacenadas en bases de datos, Data Warehouses, o cualquier otro medio de almacenamiento de información. (5)

De manera general, puede afirmarse que la Minería de Datos es un proceso apoyado en técnicas y herramientas que descubre a partir de datos almacenados nuevos conocimientos válidos y potencialmente útiles para ayudar a la toma de decisiones más seguras, que propicien algún beneficio a las organizaciones.

1.1.2 Técnicas de Minería de Datos

Las técnicas de Minería de Datos provienen de la inteligencia artificial y de la propia estadística. Dichas técnicas, no son más que algoritmos sofisticados que se aplican sobre un conjunto de datos para obtener resultados, patrones o modelos a partir de los datos recopilados. Las técnicas de Minería de Datos se clasifican en dos grandes categorías: supervisadas o predictivas y no supervisadas o descriptivas.

Tabla 1: Técnicas de Minería de Datos

Técnicas de Minería de Datos	
No supervisadas (Descriptivas)	**Supervisadas (Predictivas)**

Clustering	Asociación	Predicción	Clasificación
✓ Numérico ✓ Conceptual ✓ Probabilístico	✓ A-Priori ✓ Partition ✓ Eclat	✓ Regresión ✓ Árboles de Predicción ✓ Estimador de Núcleos	✓ Tabla de Decisión ✓ Árboles de Decisión ✓ Inducción de Reglas ✓ Basado en Ejemplares ✓ Redes de Neuronas ✓ Lógica Borrosa ✓ Técnicas Genéticas ✓ Bayesiana

La Minería de Datos contiene diversos tipos de técnicas, entre las que se destacan las técnicas de reglas de asociación, que permiten obtener correlaciones, patrones frecuentes, asociaciones o estructuras casuales entre los repositorios de series temporales. Las reglas de asociación en la Minería de Datos representan una de las técnicas más valiosas e interesantes para la extracción de conocimiento oculto en grandes volúmenes de datos. Las relaciones inter-transaccionales (con restricciones temporales y presencia del parámetro tiempo en la definición de las reglas asociativas) representan en sí los nuevos retos y novedades dentro del campo de la minería de reglas asociativas aplicadas a series temporales y datos secuenciales. (2)

Una técnica constituye el enfoque conceptual para extraer la información de los datos, y, en general es implementada por varios algoritmos. Cada algoritmo representa, en la práctica, la manera de desarrollar una determinada técnica. (2)

Las predicciones se utilizan para prever el comportamiento futuro de algún tipo de entidad mientras que una descripción puede ayudar a su comprensión. De hecho, los modelos predictivos pueden ser descriptivos y emplearse para realizar predicciones, de esta forma, hay algoritmos o técnicas que pueden servir para distintos propósitos. (2) (3)

1.2 SELECCIÓN DE LOS ALGORITMOS A IMPLEMENTAR

Las Reglas de Asociación es una técnica que permite la búsqueda automática de reglas que relacionan conjuntos entre sí, encontrando las asociaciones interesantes en forma de relaciones de implicación entre los valores de los atributos. Estas reglas son algoritmos no supervisados ya que no existen relaciones

conocidas con las que comprobar la validez de los resultados, sino que se evalúa si esas reglas son estadísticamente significativas mediante los valores del soporte y la confianza.

Esta técnica emergió en la década de los 90 con una aplicación práctica, el análisis de información de ventas para el mercadeo. Mediante ella se descubrían las relaciones entre los datos recopilados a gran escala por los sistemas de terminales de punto de venta de supermercados. Los datos consistían en colecciones de transacciones, también conocidas como bases de datos transaccionales, donde cada transacción expresa qué productos compró un cliente. (6)

Este tipo de técnicas se emplea para establecer las posibles relaciones o correlaciones entre los sucesos aparentemente independientes; pudiendo reconocer como la ocurrencia de un suceso o acción puede inducir o generar la aparición de otros. Son utilizadas cuando el objetivo es realizar análisis exploratorios, buscando relaciones dentro del conjunto de datos. Por lo general esta forma de extracción de conocimiento se fundamenta en técnicas estadísticas, como los análisis de correlación y de variación. (2)

En el presente trabajo de diploma se comienza el estudio del análisis de datos con la propuesta de implementación de los algoritmos de Minería de Datos A-Priori y Partition, estos algoritmos pertenecen a las reglas de asociación. Se decidió comenzar con la implementación de estos algoritmos pues son los más conocidos y de los que existen más documentación, también en las herramientas existentes para trabajar con Minería de Datos estos son los más utilizados.

1.2.1 A-Priori

El algoritmo A-Priori dado un conjunto de datos comienza a formar y a analizar pequeños subconjuntos de datos sobre los cuales se impone una condición definida por el usuario. Esta condición está basada en un soporte el cual debe ser menor o igual que aquel que define el usuario y su objetivo es ir descartando aquellos subconjuntos que no cumplan con ella. Con esto se garantiza que el número de conjuntos considerados se reduzcan así como el espacio de búsqueda.

Definición de las Reglas de Asociación:
Forma general: $X \Rightarrow Y$, donde X e Y son conjuntos de items (X es denominado el antecedente de la regla e Y su consecuente).

Soporte: $(A \Rightarrow B) = P(A \cap B)$
El soporte para X⇒Y es el porcentaje de las transacciones que contienen todos los items de X e Y.

Confianza: $(A \Rightarrow B) = P(B \mid A) = (P(A \cap B)) / (P(A))$
La confianza para X⇒Y es el porcentaje de transacciones que contienen Y, entre las transacciones que contienen X.

Se considera que una regla es interesante si su soporte y su confianza son mayores o iguales que ciertos umbrales de mínimo soporte y mínima confianza especificados, por lo que se buscan, independientemente de en qué lado aparezcan, pares atributo-valor que cubran una gran cantidad de ejemplos. A cada par atributo-valor se le denomina item, mientras que a un conjunto de items se les denomina item-sets. Para la formación de item-sets no se pueden unir item referidos al mismo atributo con distinto valor. Se buscan item-sets con un máximo soporte, para lo que se comienza con item-sets con un único item. Se eliminan los item-sets cuyo valor de soporte sea inferior al mínimo establecido, y se combinan el resto formando item-sets con dos items. A su vez se eliminan aquellos nuevos item-sets que no cumplan con la condición del soporte, y al resto se le añadirá un nuevo item, formando item-sets con tres items. El proceso continuará hasta que ya no se puedan formar item-sets con un item más. Para generar los item-sets de un determinado nivel, sólo es necesario emplear los item-sets del nivel inferior (con n-1 coincidencias, siendo n el número de items del nivel). Una vez se han obtenido todos los item-sets, se pasará a la generación de reglas. Se tomará cada item-set y se formarán reglas que cumplan con la condición de confianza. Debe tenerse en cuenta que un item-set puede dar lugar a más de una regla de asociación, al igual que un item-set también puede no dar lugar a ninguna regla. (2)

El algoritmo A-Priori se utiliza para la generación de ítems-set frecuentes. Este algoritmo se resume en dos pasos (6):

- ✓ Paso # 1: Generación de todos los item-sets que contienen un solo elemento, utilización de estos para generar item-sets que contengan dos elementos, y así sucesivamente. Se toman todos los posibles pares de items que cumplen con las medidas mínimas de soporte inicialmente preestablecidas; esto permite ir eliminando posibles combinaciones: aquellas que no cumplan con los requerimientos de soporte no entrarán en el análisis.

- ✓ Paso # 2: Generación de las reglas revisando que cumplan con el criterio mínimo de confianza. Es interesante observar que si una conjunción de consecuentes de una regla cumple con los niveles mínimos de soporte y confianza, sus subconjuntos (consecuentes) también los cumplen; en el caso contrario, si algún item no los cumple no tiene caso considerar sus súper conjuntos.

En la siguiente figura se describe el algoritmo A priori (7):

```
Algoritmo: Apriori
 1)  L_1 = {large 1-itemsets};
 2)  for ( k = 2; L_{k-1} ≠ ∅; k++ ) do begin
 3)    C_k = apriori_gen(L_{k-1}); // New candidates
 4)    forall transactions t ∈ D do begin
 5)      C_t = subset(C_k, t);      // Candidates contained in t
 6)      forall candidates c ∈ C_t do
 7)        c.Support ++ ;
 8)    end
 9)    L_k = {c ∈ C_k | c.Support ≥ minsup};
10)  end
11)  Answer = ⋃_k L_k;
```

Figura 1: Pseudocódigo del algoritmo A-Priori.

1.2.2 Partition

La ventaja de mantener la Base de Datos en memoria y evitar las operaciones de Entrada/Salida en disco, motivaron los trabajos de Savasere, Omiecinski y Navathe (7). Con tales objetivos, propusieron un algoritmo, al que denominaron Partition, el cual divide las transacciones de la Base de Datos en tantas partes disjuntas (o sea, en particiones no necesariamente de iguales tamaños) como fueran necesarias para que todas las transacciones en cada partición pudieran almacenarse en la memoria operativa.

En contraste con el algoritmo A-Priori clásico, este recorre la Base de Datos sólo dos veces. En la primera fase, cada partición es minada independientemente para encontrar todos los ítem-sets frecuentes localmente. Al final de esta fase se toman los ítem-sets localmente frecuentes como ítem-sets candidatos globalmente. En la segunda fase, se determinan los soportes de los ítem-sets candidatos y se identifican

los ítem-sets frecuentes. En la generación del conjunto de ítem-sets candidatos se considera la siguiente propiedad:

Propiedad: Particionamiento del soporte de un ítem-set o conjunto de items. Si un ítem-set no es localmente frecuente en ninguna parte, o subconjunto, de una partición de una Base de Datos, entonces no será frecuente globalmente.

De esta propiedad se infiere que los ítem-sets que no sean localmente frecuentes en ninguna parte entonces no deben considerarse como candidatos globales. Esta propiedad garantiza que en el conjunto de ítem-sets candidatos que se obtiene en la primera fase pudieran existir falsos positivos, pero nunca existirían falsos negativos. En la Figura No. 2 se muestra el pseudocódigo del algoritmo Partition considerando una partición P de m subconjuntos p1 a pm (7):

```
Algoritmo: Partition
 1) for ( i = 1; i≤m ; i++ ) do begin // Phase I
 2)   "Load partition p_i ∈ P";
 3)   GenItemsets(p_i, L_i);
 4) end
 5) C = ⋃_{i=1..m} L_i ;
 6) for ( i = 1; i≤m ; i++ ) do begin // Phase II
 7)   "Load partition p_i ∈ P";
 8)   AccumulateSupport(p_i, C);
 9) end
10) L = {c ∈ C | c.Support ≥ minsup};
11) Answer = L
```

Figura 2: Pseudocódigo del algoritmo Partition

1.3 HERRAMIENTAS DE MINERÍA DE DATOS

Las herramientas de la Minería de Datos son utilizadas para resolver situaciones donde los volúmenes de datos son muy grandes por la cantidad de variables que se manipulan, por lo que la extracción de conocimiento se hace compleja. Estas herramientas exploran las bases de datos en busca de patrones

ocultos, encontrando información predecible que un experto no puede llegar a encontrar porque se encuentra fuera de sus expectativas. También predicen futuras tendencias y comportamientos que permiten hacer una toma de decisiones y pueden responder a preguntas que tradicionalmente consumen demasiado tiempo para poder ser resueltas. En la actualidad existe una gran cantidad de herramientas de software para el desarrollo de modelos de Minería de Datos. A continuación se realizará una caracterización de las herramientas de Minería de Datos más utilizadas.

1.3.1 Herramientas Libres

Yale / Rapid Miner:

Las iniciales Yale responden a Yet Another Learning Environment, fue desarrollada en el lenguaje de progrmacion Java, el cual integra completamente los códigos de Weka y además permite realizar Minería de Datos. Tiene dos tipos de licencia, la gratuita bajo la licencia GPL y la propietaria. YALE puede importar información a partir de sistemas de Bases de Datos como PostgreSQL y Microsoft SQL Server. Se requiere tener instalado con anterioridad el Java Runtime Environment de Sun.

Características de YALE / Rapid Miner:

- ✓ Es un sistema prototipado para el descubrimiento del conocimiento y Minería de Datos.
- ✓ Es un software de tipo código abierto con licencia GNU GPL, basado en java.
- ✓ Trabaja bajo las plataformas Windows y Linux.
- ✓ Posee alrededor de 400 operadores que pueden ser combinados.
- ✓ Usa el lenguaje de scripting XML para describir los operadores y su configuración.
- ✓ La característica más importante es la capacidad de jerarquizar cadenas del operador y de construir complejos árboles de operadores.
- ✓ El lenguaje de encriptación permite automáticamente una gran cantidad de experimentos.
- ✓ Posee una interfaz gráfica, línea comando, y API de Java para usar Rapid Miner desde tus propios programas.
- ✓ Una gran cantidad de extensiones (plugins).
- ✓ Las aplicaciones incluyen: Text Mining, Multimedia Mining, entre otras. (2)

Weka:

Las iniciales WEKA responden a Waikato Environment for Knowledge Analysis, se trata de una herramienta de libre distribución desarrollada en la Universidad de Waikato (Nueva Zelanda), escrita en lenguaje Java y permite realizar multitud de análisis. La herramienta está aplicada a procesos de Minería de Datos, por lo que agrupa diferentes técnicas: pre-procesado, agrupamiento o clustering, ajuste de clasificadores y generación de reglas de asociación. También incluye facilidades para la visualización de los datos.

La herramienta dispone de cuatro interfaces distintas:

1. **Interfaz en modo texto:** Permite la introducción de todo tipo de comandos, pero no es posible realizar representaciones gráficas, el interfaz en modo texto permite instanciar las distintas clases Java definidas en el programa Weka.
2. **Interfaz Explorer**: Es el interfaz gráfico básico, en él se pueden mostrar gráficamente tanto las características de los datos de partida como los resultados de los análisis. Permite introducir los comandos con ayuda del mouse, seleccionando los operadores adecuados en menús desplegables.
3. **Interfaz Experimenter:** Se trata de un interfaz gráfico más avanzado, en el que no solo se pueden realizar análisis sobre los datos, sino que además es posible comparar el funcionamiento de diferentes algoritmos (por ejemplo, diferentes clasificadores) o bien comparar distintos ficheros de datos.
4. **Interfaz KnowledgeFlow:** Este último interfaz permite representar como una red de operadores en cascada los procesos a realizar sobre los datos (pre-procesado, selección de características, ajuste de un clasificador y evaluación de los porcentajes de acierto esperables).

Además de estas cuatro interfaces, también caben otras dos alternativas: en primer lugar, es posible introducir el código Java de Weka (que es accesible libremente) en una aplicación Java propia, de modo que se puede crear un programa específico, que haga uso de las utilidades de Weka que se desee. En segundo lugar y por último, también es posible instanciar las clases Java desde la línea de comandos de MS-DOS, mediante la llamada al intérprete Java de que disponga el sistema (sea este Windows, Linux o Macintosh). (8)

1.3.2 Herramientas Propietarias

Oracle Data Mining:

Oracle es una herramienta potente que permite descubrir nuevos conocimientos ocultos en los datos. Ayuda a buscar nueva información valiosa, patrones en los datos, identificar los atributos clave, descubrir nuevos clusters y asociaciones, y revelar conocimientos valiosos. Esta herramienta permite buscar nueva información en sus datos utilizando una amplia gama de algoritmos de avanzada. Oracle proporciona múltiples algoritmos ya que diferentes algoritmos son efectivos para diferentes tipos de análisis y diferentes problemas de negocio.

La mayoría de los algoritmos de Minería de Datos pueden separarse en técnicas con "aprendizaje supervisado" y "aprendizaje no supervisado". La técnica de aprendizaje supervisado examina cuidadosamente los datos para buscar patrones y relaciones entre los otros atributos y el atributo objetivo. Los algoritmos de aprendizaje supervisado de Oracle incluyen Naive Bayes, Árbol de Decisión, Modelos Lineales Generalizados y Máquinas de Vectores Soporte. La otra gran categoría de los algoritmos de Minería de Datos es para el aprendizaje no supervisado, en la cual los algoritmos buscan encontrar asociaciones y clusters en los datos independientemente de cualquier objetivo de negocios definido. Estos algoritmos incluyen Clustering k-Means mejorado, Clustering de Partición Ortogonal, Reglas de Asociación y Factorización de Matrices No Negativas.

Oracle incluye una interface gráfica de usuarios para el análisis de datos que tiene el fin de crear, evaluar y aplicar modelos de Minería de Datos. Oracle guía al analista de datos a través del proceso Minería de Datos con total flexibilidad y presenta los resultados en formatos gráficos y tabulares. Oracle puede generar el código PL/SQL asociado con una Actividad de Recuperación de los Datos. (9) (10)

SQL Server Data Mining:

SQL Server Data Mining es una herramienta que contiene las características necesarias para crear complejas soluciones de Minería de Datos ya que permite:

- ✓ Aplicar soluciones de Minería de Datos utilizando Microsoft Excel.
- ✓ Entender cómo, cuándo y dónde aplicar los algoritmos que se incluyen en el servidor de SQL.

- ✓ Realizar la extracción de datos de procesamiento analítico en línea (OLAP).
- ✓ Utilizar SQL Server Management Studio para acceder y proteger los objetos de Minería de Datos.
- ✓ Utilizar SQL Server Business Intelligence Development Studio para crear y gestionar proyectos de Minería de Datos.

Algunas de las ventajas de la Minería de Datos de Microsoft es la integración estrecha con la plataforma de Base de Datos de clase mundial SQL Server, ya que aprovecha el desempeño, la seguridad y las características de optimización de SQL Server; extensibilidad ya que se puede extender la Minería de Datos de Microsoft para implementar algoritmos que no vienen incluidos en el producto.

Varios de los algoritmos implementados por Microsoft son: Árboles de Decisión, Bayes Naive, Clústeres, Redes Neuronales, Serie Temporal, Regresión Lineal, Clústeres de Secuencia y Asociación. (1)

1.3.3 Fundamentación de la herramienta seleccionada

Después de un estudio de las herramientas de la Minería de Datos, se puede llegar a la conclusión de que tanto YALE / Rapid Miner como Weka a pesar de ser de libres y poseer diversas ventajas tienen las desventajas de que el proceso es engorroso ya que requiere de tiempo para la preparación y la vinculación de los datos con el gestor, extendiendo así el tiempo de respuesta de los análisis de los datos.

Por otra parte ORACLE y SQL Server son herramientas muy potentes, y una de sus mayores fortalezas radica en la integración con el Sistema Gestor de Base de Datos, pero ambas son herramientas propietarias e implican altas inversiones por el uso de herramientas comerciales y es muy costoso para la economía del país.

Es por ello que resulta necesario implementar los algoritmos de Minería de Datos de reglas de asociación para integrarlos al Gestor de Base de Datos PostgreSQL y aprovechar al máximo sus potencialidades.

1.4 METODOLOGÍAS PARA APLICAR LA MINERÍA DE DATOS

Para implementar una tecnología en un negocio, se requiere de una metodología. La mayoría de las consultoras especializadas en alguna tecnología cuentan con una metodología, según los tipos de proyectos que aborden. Estos métodos son definidos a partir de sus experiencias y tomando lo mejor de

procedimientos más exitosos o populares. Contar con una metodología se ha convertido hoy en día en un eslabón importante y necesario para las empresas. A continuación se describen algunas de las metodologías usadas en la Minería de Datos.

1.4.1 CRISP-DM

Las iniciales CRISP-DM responden a Cross Industry Standard Process for Data Mining (según sus siglas en Inglés), esta metodología consiste en un conjunto de tareas descritas en cuatro niveles de abstracción: fase, tarea genérica, tarea especializada, e instancia de proceso, organizados de forma jerárquica en tareas que van desde el nivel más general hasta los casos más específicos.

CRISP-DM está dividida en 4 niveles de abstracción organizados de forma jerárquica en tareas que van desde el nivel más general, hasta los casos más específicos y organiza el desarrollo de un proyecto de Minería de Datos en una serie de 6 fases.

Niveles de abstracción:

- ✓ **Fase**: Se le denomina fase al asunto o paso dentro del proceso. CRISP-DM consta de 6 fases: comprensión del negocio, comprensión de los datos, preparación de los datos, modelación, evaluación y explotación.
- ✓ **Tarea genérica**: Cada fase está formada por tareas genéricas, o sea, la tarea genérica es la descripción de las actividades que se realizan dentro de cada fase. Por ejemplo, la tarea Limpiar los datos es una tarea genérica.
- ✓ **Tarea especializada**: La tarea especializada describe cómo se pueden llevar a cabo las tareas genéricas en situaciones específicas. Por ejemplo, la tarea Limpiar los datos tiene tareas especializadas, como limpiar valores numéricos, y limpiar valores categóricos.
- ✓ **Instancias de proceso**: Las instancias de proceso son las acciones y resultados de las actividades realizadas dentro de cada fase del proyecto. Las fases del proyecto de Minería de acuerdo a lo establecido por la metodología CRISP-DM interactúan entre ellas de forma iterativa durante el desarrollo del proyecto. La secuencia de las fases no siempre es ordenada, o en ocasiones si se determina al realizar la evaluación que los objetivos del negocio no se cumplieron se debe regresar y buscar las causas del problema para redefinirlo. (11)

A continuación se describen cada una de las fases en que se divide CRISP-DM:

- ✓ **Fase # 1 - Comprensión del negocio:**

La primera fase es probablemente la más importante, ya que es muy importante la capacidad de poder convertir el conocimiento adquirido del negocio, en un problema de Data Mining y en un plan preliminar cuya meta sea el alcanzar los objetivos del negocio. Las principales tareas que componen esta fase son las siguientes:

1. Determinar los objetivos del negocio.
2. Evaluación de la situación.
3. Determinación de los objetivos de la Minería de Datos.

- ✓ **Fase # 2 - Comprensión de los datos:**

Esta fase comprende la recolección inicial de datos, en esta junto a las próximas dos fases, son las que demandan el mayor esfuerzo y tiempo en un proyecto de Minería de Datos Las principales tareas que componen esta fase son las siguientes:

- ✓ Recolección de datos iniciales.
- ✓ Descripción de los datos.
- ✓ Exploración de datos.
- ✓ Verificación de la calidad de los datos.

- ✓ **Fase # 3 - Preparación de los datos:**

En esta fase una vez efectuada la recolección inicial de datos, se procede a su preparación para adaptarlos a las técnicas de Minería de Datos. Las principales tareas que componen esta fase son las siguientes:

- ✓ Selección de datos.
- ✓ Limpieza de los datos.
- ✓ Estructuración de los datos.
- ✓ Integración de los datos.
- ✓ Formateo de los datos.

- ✓ **Fase # 4 - Modelado:**

Las técnicas a utilizar en esta fase se eligen en función de los siguientes criterios:

- ✓ Ser apropiada al problema.
- ✓ Disponer de datos adecuados.
- ✓ Cumplir los requisitos del problema.
- ✓ Tiempo adecuado para obtener un modelo.
- ✓ Conocimiento de la técnica.

Las principales tareas que componen esta fase son las siguientes:

1. Selección de la técnica de modelado.
2. Generación del plan de prueba.
3. Construcción del Modelo.
4. Evaluación del modelo.

- ✓ **Fase # 5 - Evaluación:**

En esta fase se evalúa el modelo, teniendo en cuenta el cumplimiento de los criterios de éxito del problema. Las principales tareas que componen esta fase son las siguientes:

1. Evaluación de los resultados.
2. Proceso de revisión.
3. Determinación de futuras fases.

- ✓ **Fase # 6 - Implementación:**

Las tareas que se ejecutan en esta fase son las siguientes:

1. Plan de implementación.
2. Monitorización y Mantenimiento.
3. Informe Final.
4. Revisión del proyecto. (12)

1.4.2 SEMMA

SA Systems es el Instituto desarrollador de esta metodología, el cual la define como el proceso de selección, exploración y modelado de grandes cantidades de datos para descubrir patrones de negocio desconocidos.

La metodología SEMMA se caracteriza principalmente por priorizar sus fases desde un punto de vista técnico, es decir, dando prioridad a las prácticas usadas para su implementación y obtención de resultados. El nombre de esta terminología es el acrónimo correspondiente a las cinco fases básicas del proceso de Minería de Datos (Sample, Explore, Modify, Model, Assess). Estos cinco pasos son:

- ✓ **Muestreo:**

Es la primera etapa del proyecto, en ella se preparan los datos para su posterior exploración. En esta etapa es común la utilización del nodo de partición (especialmente si quieren realizarse árboles de decisión o redes neuronales). Normalmente se suele utilizar un porcentaje de 70 para la muestra de entrenamiento y uno de 30 para la validación.

- ✓ **Exploración:**

Se trata de la exploración de los datos, la cual es una de las partes más trabajosas. En este paso se tiene un nodo que ayuda a explorar gráficamente los datos, otro nodo de selección de variables que ayuda a eliminar aquellos inputs que no tienen relación con la variable objetivo, incluso se puede hacer un "clustering" o una segmentación.

- ✓ **Modificación:**

Cuando se llega a este paso hay que centrarse en la selección y transformación de variables y datos que servirán para la construcción de los modelos. Entre otras tareas a realizar destacan: la reducción de dimensión, imputación de valores "missing" y "outliers".

- ✓ **Modelización:**

En este paso se escogen los modelos, la elección del modelo va a depender esencialmente de los datos que se tienen, del tipo de variables y de obtener modelos fácilmente entendibles. Se pueden escoger varios modelos como regresión, regresión logística, árboles de decisión, análisis factorial discriminante y redes neuronales. También se pueden aplicar más de un modelo a la vez, y luego comparar los resultados obtenidos.

- ✓ **Estimación:**

Después de realizados todos los pasos se llega al momento de comparar los modelos. Para realizarlo se utiliza el análisis del diagrama Características Operativas del Receptor(ROC, en sus siglas en inglés Receiver Operating Characteristic). La curva ROC es útil para comparar el comportamiento global de un modelo. El gráfico ROC enfrenta dos variables: la sensibilidad y la especificidad. Lo ideal es que ambas categorías sean altas. (13)

Después de realizar un análisis de las principales características de las metodologías descritas anteriormente se puede observar que todas estas metodologías comparten la misma esencia. Se estructura el KDD en fases similares, que se encuentran inter-relacionadas entre sí; y lo describen de forma iterativa e interactiva. Se pudo comprobar que algunas (como SEMMA) se centran más en los aspectos técnicos del desarrollo del proceso; mientras que otras (como CRISP-DM) mantienen una perspectiva más amplia respecto a los objetivos empresariales del proyecto. CRISP-DM ve el proyecto de forma global, tomando en cuenta los aspectos relacionados con el campo de aplicación y mantiene una perspectiva más amplia.

Esta diferencia se establece ya desde la primera fase del proyecto de Minería de Datos donde la metodología SEMMA comienza realizando un muestreo de datos, mientras que la metodología CRISP-DM comienza realizando un análisis del problema empresarial para su transformación en un problema técnico. Otra diferencia significativa entre la metodología SEMMA y la metodología CRISP-DM radica en su relación con herramientas comerciales. La metodología SEMMA sólo es abierta en sus aspectos generales ya que está muy ligada a los productos SAS donde se encuentra implementada. Por su parte la metodología CRISP-DM ha sido diseñada como una metodología neutra respecto a la herramienta que se utilice para el desarrollo del proyecto de Minería de Datos siendo su distribución libre y gratuita.

Por estas características planteadas se ha seleccionado a la metodología CRISP-DM, pues desde un punto de vista más global se puede considerar que la metodología CRISP-DM está más cercana al concepto real de proyecto.

1.5 METODOLOGÍAS DE DESARROLLO DE SOFTWARE

Las metodologías de desarrollo de software no son más que un conjunto de pasos y procedimientos a seguir para desarrollar un software. Son las encargadas de estructurar, planificar y controlar el proceso de desarrollo para lograr una mejor calidad de las aplicaciones. Facilitan el trabajo haciéndolo más entendible mediante la división del proyecto en varias etapas. A continuación se explican de forma resumida algunas de las metodologías ágiles más conocidas:

- ✓ **SCRUM:** Está especialmente indicada para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en dos. El desarrollo de software se realiza mediante iteraciones, denominadas sprints, con una duración de 30 días. El resultado de cada sprint es un incremento ejecutable que se muestra al cliente. La segunda característica importante son las reuniones a lo largo del proyecto. Éstas son las verdaderas protagonistas, especialmente la reunión diaria del equipo de desarrollo para coordinación e integración. (14)
- ✓ **Crystal Methodologies:** Se trata de un conjunto de metodologías para el desarrollo de software caracterizadas por estar centradas en las personas que componen el equipo y la reducción al máximo del número de artefactos producidos. El equipo de desarrollo es un factor clave, por lo que se deben invertir esfuerzos en mejorar sus habilidades y destrezas, así como tener políticas de trabajo en equipo definidas. (15)
- ✓ **Dynamic Systems Development Method (DSDM):** Define el marco para desarrollar un proceso de producción de software. Sus principales características son: es un proceso iterativo e incremental y el equipo de desarrollo y el usuario trabajan juntos. Propone cinco fases: estudio viabilidad, estudio del negocio, modelado funcional, diseño y construcción, y finalmente implementación. Además existe realimentación en todas las fases. (16)

- ✓ **Adaptive Software Development (ASD):** Sus principales características son: iterativo, orientado a los componentes software más que a las tareas y tolerante a los cambios. El ciclo de vida que propone tiene tres fases esenciales: especulación, colaboración y aprendizaje. En la primera de ellas se inicia el proyecto y se planifican las características del software; en la segunda desarrollan las características y finalmente en la tercera se revisa su calidad, y se entrega al cliente. La revisión de los componentes sirve para aprender de los errores y volver a iniciar el ciclo de desarrollo. (17)
- ✓ **Feature-Driven Development (FDD):** Define un proceso iterativo que consta de 5 pasos. Las iteraciones son cortas (hasta 2 semanas). Se centra en las fases de diseño e implementación del sistema partiendo de una lista de características que debe reunir el software. (18)
- ✓ **Lean Development (LD):** En esta metodología los cambios se consideran riesgos, pero si se manejan adecuadamente se pueden convertir en oportunidades que mejoren la productividad del cliente. Su principal característica es introducir un mecanismo para implementar dichos cambios. (19)
- ✓ **Extreme Programming (XP):** En esta metodología los principios y prácticas son de sentido común pero llevadas al extremo, de ahí proviene su nombre. Es una metodología centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. Se basa en la realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. (20)

En la siguiente tabla se hace una comparación entre las metodologías en base a tres parámetros: vista del sistema como algo cambiante, tener en cuenta la colaboración entre los miembros del equipo y características más específicas de la propia metodología como son simplicidad, excelencia técnica, resultados y adaptabilidad (20):

	CMM	ASD	Crystal	DSDM	FDD	LD	Scrum	XP
Sistema como algo cambiante	1	5	4	3	3	4	5	5
Colaboración	2	5	5	4	4	4	5	5
Características Metodología (CM)								
-Resultados	2	5	5	4	4	4	5	5
-Simplicidad	1	4	4	3	5	3	5	5
-Adaptabilidad	2	5	5	3	3	4	4	3
-Excelencia técnica	4	3	3	4	4	4	3	4
-Prácticas de colaboración	2	5	5	4	3	3	4	5

Figura 3: Ranking de "agilidad" (Los valores más altos representan una mayor agilidad)

Como se muestra en la Figura No. 3, la metodología XP tiene los valores más altos, por lo cual el uso de esta metodología proporciona muchas ventajas en lo que se refiere a los parámetros evaluados. Para la integración de las reglas de asociación al SGBD PostgreSQL se determina utilizar la metodología XP pues sus características se ajustan a las necesidades del presente trabajo. Con el uso de esta metodología se consiguen productos fáciles de usar y con mayor rapidez, más fiables y robustos contra los fallos, gracias al diseño de las pruebas de forma previa a la codificación.

1.6 HERRAMIENTAS Y LENGUAJE DE PROGRAMACIÓN

1.6.1 PostgreSQL 9.1

PostgreSQL es el líder en sistemas de Bases de Datos de código abierto, sus características técnicas la hacen una de las bases de datos más potentes y robustas del mercado. El conjunto de características maduras que tiene PostgreSQL no sólo rivaliza con sistemas de bases de datos propietarias, sino que los supera en características avanzadas, extensibilidad, seguridad y estabilidad. PostgreSQL funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema.

La versión 9.1 de PostgreSQL ofrece muchas características que los usuarios han estado solicitando por años, retirando obstáculos para el despliegue de aplicaciones nuevas o migradas en PostgreSQL. Estas incluyen:

- ✓ **Replicación Sincrónica**: permitiendo alta disponibilidad con consistencia sobre múltiples servidores.
- ✓ **Regionalización por columna**: soportando correctamente el ordenamiento por lenguaje en las bases de datos, tablas o columnas.
- ✓ **Tablas unlogged**: importante incremento del rendimiento para datos efímeros.

PostgreSQL presenta varias ventajas, entre las cuales se destacan las siguientes:

- ✓ Ideal para tecnologías Web.
- ✓ Fácil de Administrar.
- ✓ Su sintaxis SQL es estándar y fácil de aprender.
- ✓ Footprint bajo de memoria, bastante poderoso con una configuración adecuada.
- ✓ Multiplataforma.
- ✓ Capacidades de replicación de datos.
- ✓ Soporte empresarial disponible. (21)

1.6.2 PgAdmin III

PgAdmin III es una aplicación gráfica para gestionar el gestor de Bases de Datos PostgreSQL y sus derivados (EnterpriseDB Postgres Plus Advanced Server y Greenplum Database), siendo esta una herramienta completa y popular con licencia Open Source. Está escrita en C++ usando la librería gráfica multiplataforma wxWidgets, lo que permite que se pueda usar en Linux, FreeBSD, Solaris, Mac OS X y Windows. Es capaz de gestionar versiones a partir de PostgreSQL 7.3 ejecutándose en cualquier plataforma, así como versiones comerciales de PostgreSQL como Pervasive Postgres, EnterpriseDB, Mammoth Replicator y SRA PowerGres.

PgAdmin III está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. El interfaz gráfico soporta todas las

características de PostgreSQL y facilita enormemente la administración. La aplicación también incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor, un agente para lanzar scripts programados, soporte para el motor de replicación Slony-I y mucho más. La conexión al servidor puede hacerse mediante conexión TCP/IP o Unix Domain Sockets (en plataformas *nix), y puede encriptarse mediante SSL para mayor seguridad. (22)

1.6.3 PL/pgSQL

PL/pgSQL (Procedural Language/PostgreSQL Structured Query Language) es un lenguaje imperativo provisto por el gestor de Base de Datos PostgreSQL. Permite ejecutar comandos SQL mediante un lenguaje de sentencias imperativas y uso de funciones, dando mucho más control automático que las sentencias SQL básicas.

Desde PL/pgSQL se pueden realizar cálculos complejos y crear nuevos tipos de datos de usuario. Como un verdadero lenguaje de programación, dispone de estructuras de control repetitivas y condicionales, además de la posibilidad de creación de funciones que pueden ser llamadas en sentencias SQL normales o ejecutadas en eventos de tipo disparador.

Una de las principales ventajas de ejecutar programación en el servidor de Base de Datos es que las consultas y el resultado no tienen que ser transportadas entre el cliente y el servidor, ya que los datos residen en el propio servidor. Además, el gestor de Base de Datos puede planificar optimizaciones en la ejecución de la búsqueda y actualización de datos.

Las funciones escritas en PL/pgSQL aceptan argumentos y pueden devolver valores de tipo básico o de tipo complejo (por ejemplo, registros, vectores, conjuntos o incluso tablas), permitiéndose tipificación polimórfica para funciones abstractas o genéricas (referencia a variables de tipo objeto). (23)

1.7 CONCLUSIONES DEL CAPÍTULO

En este capítulo se abarcó de manera detallada todo lo referente al proceso de Minería de Datos y se plantearon definiciones fundamentales para lograr una mejor comprensión de la misma. Se realizó una explicación sobre las herramientas para trabajar con Minería de Datos más usadas a nivel mundial y se

decidió que es necesario implementar los algoritmos de reglas de asociación para integrarlos al Gestor de Base de Datos PostgreSQL 9.1. Se hizo una comparación entre las metodologías del proceso de Minería de datosD hasta seleccionar la metodología CRISP-DM, la cual es la más óptima para dar solución al problema planteado. También se seleccionaron los algoritmos a implementar, los cuales son A-Priori y Partition, que pertenecen a las reglas de asociación de la Minería de Datos. Para concluir el capítulo se abordaron los temas referentes a la herramienta y el lenguaje de programación del Gestor de Base de Datos PostgreSQL 9.1 donde se decidió utilizar PL/pgSQL.

CAPÍTULO 2: DESCRIPCIÓN DE LA SOLUCIÓN

Introducción

En el presente capítulo se describen las principales características que tendrá la extensión. La misma cuenta con el modelo de dominio, también se definen las historias de usuarios, en la cuales se planifican las iteraciones y se realiza una estimación de la realización de cada tarea. Además se incluye la lista de reserva del producto, donde se muestran los requisitos tanto los funcionales como los no funcionales. También se describirán brevemente los algoritmos implementados y se mostrarán imágenes de la implementación. Para finalizar el capítulo se explicará la integración de los algoritmos al SGBD PostgreSQL 9.1.

2.1 Identificación del problema

El proyecto de PostgreSQL es uno de los proyectos de la Universidad de las Ciencias Informáticas que se enfoca en el trabajo con las Bases de Datos. Este proyecto tiene como misión potenciar las funciones del Gestor de Base de Datos PostgreSQL. Por lo cual es necesario agregarle algunos algoritmos de reglas de asociación de la Minería de Datos, pues a la hora de realizar análisis con grandes volúmenes de datos se vuelve engorroso si no se cuenta con las técnicas necesarias para facilitar más el trabajo. Integrarle estos algoritmos al SGDB hace que el sistema sea más potente a la hora realizar cualquier análisis de Minería de Datos.

2.2 Modelo de dominio

El modelo de dominio es un artefacto de la disciplina de análisis, construido con las reglas de UML durante la fase de concepción que captura los tipos más importantes de objetos en el contexto del sistema. Puede utilizarse para capturar y expresar el entendimiento ganado mediante el análisis como paso previo al diseño de un sistema. Este es utilizado por el analista como un medio para comprender de una manera mucho más fácil el sistema que se va a realizar. También puede ser tomado como el punto de partida para el diseño del sistema.

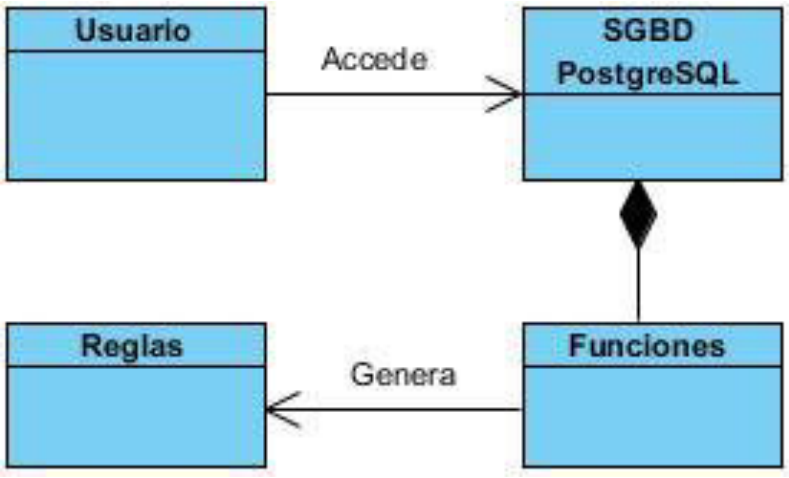


Figura 4: Modelo de dominio del sistema

Usuario: Persona que interactúa con la herramienta.
SGBD PostgreSQL: Herramienta de administración de Base de Datos.
Funciones: Conjunto de funciones que se van a integrar al SGBD PostgreSQL.
Reglas: Conjuntos de reglas que genera el SGDB PostgreSQL.

2.3 Propuesta del componente a desarrollar

Lo que se propone realizar con el siguiente trabajo diploma es integrar los algoritmos de las reglas de asociación al SGBD PostgreSQL, lo cual es de gran importancia pues una vez realizado el objetivo se podrá ganar tiempo a la hora del análisis de los datos, pues no es necesario convertir los datos ni conectarse a otra herramienta. Además incorporando las reglas de asociación se permite aprovechar las potencialidades del SGBD PostgreSQL para realizar el proceso del análisis.

2.4 Historias de usuarios

Una historia de usuario es una representación de un requisito funcional escrito utilizando el lenguaje común de los usuarios. Las mismas son utilizadas en las metodologías de desarrollo ágiles para la especificación de requisito. Las historias de usuario son una forma rápida de administrar los requisitos de los usuarios sin tener que elaborar gran cantidad de documentos, estas permiten responder rápidamente a los requisitos cambiantes.

Se definieron para el desarrollo de los algoritmos de las reglas de asociación 2 historias de usuarios, las cuales se muestran en las siguientes tablas:

Tabla 2: Historia de usuario: Generar reglas de asociación a través del algoritmo A-Priori

Historia de Usuario	
Número: 1	**Nombre de la Historia de Usuario:** Generar reglas de asociación a través del algoritmo A-Priori
Cantidad de modificaciones a la Historia de Usuario: 0	
Usuario: Audrey Cordero Sánchez	**Iteración asignada:** 1
Prioridad en negocio: Muy Alta	**Puntos estimados:** 3 Semana
Riesgo en desarrollo: Alto	**Puntos reales: (**[Tiempo real dedicado a la realización de la HU en semanas.]**)**
Descripción: Permite al usuario generar a partir de los datos contenidos en una base de datos reglas de asociación basado en el conocimiento de los conjuntos frecuentes para reducir los espacios de búsqueda y aumentar la eficiencia.	
Observaciones: N/A	
Prototipo de interfaces N/A	

Tabla 3: Historia de usuario: Generar reglas de asociación a través del algoritmo Partition

Historia de Usuario	
Número: 2	**Nombre de la Historia de Usuario:** Generar reglas de asociación a través del algoritmo Partition

Cantidad de modificaciones a la Historia de Usuario: 0	
Usuario: Audrey Cordero Sánchez	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 3 Semana
Riesgo en desarrollo: Alto	**Puntos reales: (**[Tiempo real dedicado a la realización de la HU en semanas.]**)**
Descripción: Permite al usuario generar a partir de los datos contenidos en una Base de Datos reglas de asociación dividiendo las transacciones de la Base de Datos en partes, no necesariamente del mismo tamaño, para que estas transacciones en cada partición puedan almacenarse en la memoria operativa.	
Observaciones: N/A	
Prototipo de interfaces: N/A	

2.5 Lista de reserva del producto

La lista de reserva del producto representa todos los requerimientos, tanto funcionales como no funcionales ordenados todos según la prioridad con la estimación de cada uno de los requerimientos para su implementación por semanas y el rol que realizó la estimación del requerimiento.

Tabla 4: Lista de reserva del producto

Item *	Descripción	Estimación	Estimado por
Prioridad: Muy Alta			
1	Generar reglas de asociación a través del algoritmo A-Priori	3 Semanas	Analista
Prioridad: Alta			
2	Generar reglas de asociación a través del algoritmo Partition	3 Semanas	Analista

Prioridad: Media			
Prioridad: Baja			
Requisitos No Funcionales			
1	Facilidad de uso: Para utilizar la extensión es necesario poseer conocimientos elementales del Gestor PostgreSQL.		
2	Software: Para utilizar la extensión debe estar instalado, el gestor PostgreSQL a partir de la versión 9.1.		
3	Hardware: El ordenador donde se utilice la extensión debe contar con 256 Mb de memoria RAM como mínimo, un microprocesador de 300MHz de frecuencia.		

2.6 Tareas de la ingeniería

Una vez descritas las historias de usuarios, se procede a describir cada una de las tareas que se van a elaborar dentro de cada una de ellas. Las tareas de la ingeniería se consideran como las entradas de trabajo para el equipo de programadores. Es la ficha que contiene el número identificador de la tarea, el identificador de la historia de usuario con la que está relacionada, el nombre de la tarea, la fecha de inicio, la fecha de fin, el equipo responsable y la descripción. A continuación se muestran tres ejemplos de las tareas de ingenierías asociadas a la historias de usuarios: Generar reglas de asociación a través del algoritmo A-Priori.

Tabla 5: Tarea de la ingeniería de la historia de usuario 1

Tarea de Ingeniería

Número Tarea: 1	**Número Historia de Usuario:** 1
Nombre Tarea: Estudio de los pasos del algoritmo A-Priori	
Tipo de Tarea : Otra (Estudio)	**Puntos Estimados:** 1 semana
Fecha Inicio: 24/02/2013	**Fecha Fin:** 02/03/2013
Programador Responsable: Audrey Cordero Sánchez	
Descripción: Realizar un estudio de la documentación del algoritmo A-Priori de las reglas de asociación de la Minería de Datos.	

Tabla 6: Tarea de la ingeniería de la historia de usuario 1

	Tarea de Ingeniería
Número Tarea: 2	**Número Historia de Usuario:** 1
Nombre Tarea: Calcular el soporte y la confianza de los ítem-set	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1 semana
Fecha Inicio: 03/03/2013	**Fecha Fin:** 09/03/2013
Programador Responsable: Audrey Cordero Sánchez	
Descripción: Se cargan los datos de la tabla sobre la cual se va a trabajar, se introducen los valores del mínimo de soporte y del mínimo de confianza. Además se calculan los valores de soporte, confianza de cada ítem y se compara que el resultado del soporte obtenido sea mayor que el mínimo soporte.	

Tabla 7: Tarea de la ingeniería de la historia de usuario 1

	Tarea de Ingeniería
Número Tarea: 3	**Número Historia de Usuario:** 1

Nombre Tarea: Mostrar las reglas de asociación	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1 semana
Fecha Inicio: 10/03/2013	**Fecha Fin:** 16/03/2013
Programador Responsable: Audrey Cordero Sánchez	
Descripción: Se unen los items que tengan el soporte mayor que el mínimo soporte y se conforman los ítem-set. Después se generan las reglas que cumplan con la condición de que su confianza sea mayor que el mínimo de confianza	

2.7 Plan de iteraciones

El plan de iteraciones es una planificación donde los desarrolladores y clientes establecen los tiempos de implementación ideales de las historias de usuario, la prioridad con la que serán implementadas y las historias que serán implementadas en cada versión del programa. Después de un plan de iteraciones tienen que estar claros estos cuatro factores: los objetivos que se deben cumplir (que son principalmente las historias que se deben desarrollar en cada versión), el tiempo que tardarán en desarrollarse y publicarse las versiones del programa, el número de personas que trabajarán en el desarrollo y cómo se evaluará la calidad del trabajo realizado.

Tabla 8: Plan de Iteraciones

Release	Descripción de la iteración	Orden de la HU a implementar	Duración total

1	En esta iteración se implementarán las Historias de Usuario que tengan la prioridad en el negocio MUY ALTA.	1,2	6 semanas
2	En esta iteración se implementarán las Historias de Usuario que tengan la prioridad en el negocio ALTA.	3	3 semanas

2.8 Estándares de codificación

Los estándares de codificación son pautas de programación que no están enfocadas a la lógica del programa, sino a su estructura y apariencia física para facilitar la lectura, comprensión y mantenimiento del código.

Identación:

La unidad de identado es de 2 espacios. El uso de la tabulación debe ser evitado porque (tal como se escribía en el siglo pasado) no existe un estándar que determine con precisión el ancho que va a producir la tabulación.

```
FOR var1 IN SELECT c, a FROM tablatemp1
LOOP
  FOR var2 IN SELECT c, a FROM tablatemp1
    LOOP
    IF (var1.columna <> var2.columna  AND var1.valor <> var2.valor) THEN
      EXECUTE ' SELECT count(*) FROM ' ||$3|| ' WHERE ' ||var1.columna|| ' = ' || '''' ||var1.valor|| '''' || ' AND '
      ||var2.columna|| ' = ' || '''' ||var2.valor|| '''' INTO cantidad_segundo;
      IF (cantidad_segundo / cantidad_tuplas)  > min_soporte THEN
        EXECUTE 'SELECT count(*) FROM tablatemp2 WHERE  d='|| '''' ||var2.columna|| ''''|| 'AND e ='|| '''' ||var2.valor||
        ''''|| 'AND f ='|| '''' ||var1.columna|| ''''|| 'AND g ='|| '''' ||var1.valor||''''  INTO var_tab_2;
          IF var_tab_2 = 0 THEN
            INSERT INTO tablatemp2 VALUES (var1.columna, var1.valor, var2.columna, var2.valor, cantidad_segundo / cantidad_tuplas);
          END IF;
      END IF;
    END IF;
  END LOOP;
END LOOP;
```

Figura 5: Ejemplo de identación en la implementación del algoritmo A-Priori

Comentarios:

Es conveniente dejar información que pueda ser leída tiempo después por personas (posiblemente usted mismo) que necesitan entender que fue lo que se hizo en el fragmento de código. Los comentarios deben ser escritos correctamente y claros. Generalmente deben usarse comentarios de una sola línea. Reserve los comentarios de bloques para la documentación formal o para comentar porciones de código.

```
-- Tablas temporales para debolver los conjuntos de items
CREATE TEMP TABLE tablatempl (c VARCHAR, a VARCHAR, b NUMERIC);  -- Columnal y Valorl, Soporte
CREATE TEMP TABLE tablatemp2 (d VARCHAR, e VARCHAR, f VARCHAR, g VARCHAR, h NUMERIC);  -- Columnal y Valor2, Columnal y Valor2,
CREATE TEMP TABLE tablatemp3 (i VARCHAR, j VARCHAR, k VARCHAR, l VARCHAR, m VARCHAR, n VARCHAR, soporte NUMERIC);  -- Columnal
CREATE TEMP TABLE tablatemp4 (r VARCHAR, q NUMERIC, p NUMERIC ); -- RESULTADO FINAL

-- Cantidad de columnas existentes en la tabla
SELECT count(*) INTO cantidad_column FROM pg_attribute atributo, pg_class
clase, pg_tables tablas WHERE atributo.attrelid = clase.relfilenode AND atributo.attstattarget = '-1' AND
clase.relname = tablas.tablename AND tablas.tablename LIKE $3;

-- Cantidad de Tuplas de la tabla
EXECUTE 'SELECT COUNT(*) FROM '||$3 INTO cantidad_tuplas;

-- Devuelve el nombre de las columnas existentes en la tabla
OPEN mi_cursor FOR SELECT atributo.attname FROM pg_attribute atributo, pg_class
clase, pg_tables tablas WHERE atributo.attrelid = clase.relfilenode AND atributo.attstattarget = '-1' AND
clase.relname = tablas.tablename AND tablas.tablename LIKE $3;

-- Ciclo para obtener 3 columna(Nom_Colum - Nom_Var - Soporte)
WHILE (cantidad_column <> 0)
LOOP
  FETCH mi_cursor INTO nombre_column;
  FOR variable IN EXECUTE ' SELECT ' ||nombre_column||  ', count(*) FROM ' ||$3||' GROUP BY '||nombre_column
    LOOP
      IF (variable.soporte / cantidad_tuplas)  > min_soporte THEN
        INSERT INTO tablatempl VALUES (nombre_column, variable.nombre, (variable.soporte / cantidad_tuplas) );
      END IF;
    END LOOP;
cantidad_column := cantidad_column-1;
END LOOP;

-- Devuelve la primera iteracion paso(la primera tabla temporal con un solo item)
-- RETURN query SELECT * FROM tablatempl;
```

Figura 6: Ejemplo de comentarios en la implementación del algoritmo A-Priori

Declaración de variables:

Cada variable debe de ser declarada en una línea y comentada. El nombre de las variables debe de comenzar con letras minúsculas y cada palabra relevante por la que esté compuesta debe ser con letra minúscula y separada por un guión bajo. Cada variable que sea declarada estará comentada para lograr un mejor entendimiento.

```
variable nombreresult;  -- Tipo de dato de la lra tabla()
nombre_column VARCHAR;  -- Nombre de las columnas existentes en la tabla
nombre_tabla VARCHAR;  -- Nombre de la trabla q se va a trabajar (La pasan por parametro)
cantidad_tuplas NUMERIC DEFAULT 0;  -- Total de tuplas de la tabla
cantidad_column NUMERIC DEFAULT 0;  -- Cantidad de columnas de la tabla
cantidad_segundo INTEGER DEFAULT 0;  -- Cantidad utilizada en la 2da tabla
cantidad_tercero INTEGER DEFAULT 0;  ---- Cantidad utilizada en la 3ra tabla
var_tab_2 INTEGER DEFAULT 0;  -- Varible de la tabla temporal 2
var_tab_3 INTEGER DEFAULT 0;  -- Varible de la tabla temporal 3
var_tab_4 INTEGER DEFAULT 0;  -- Varible de la tabla temporal 4
var_tab_5 INTEGER DEFAULT 0;  -- Varible de la tabla temporal 5
var_tab_6 INTEGER DEFAULT 0;  -- Varible de la tabla temporal 6
var_tab_7 INTEGER DEFAULT 0;  -- Varible de la tabla temporal 7
var_tab_8 INTEGER DEFAULT 0;  -- Varible de la tabla temporal 8

conf1 NUMERIC DEFAULT 0;  -- Resultado del soporte para CALCULAR la confianza: lra -> 2da y 3ra (Tabla 3)
conf2 NUMERIC DEFAULT 0;  -- Resultado del soporte para CALCULAR la confianza: 2da -> lra y 3ra (Tabla 3)
conf3 NUMERIC DEFAULT 0;  -- Resultado del soporte para CALCULAR la confianza: 3ra -> lra y 2da (Tabla 3)
conf4 NUMERIC DEFAULT 0;  -- Resultado del soporte para CALCULAR la confianza: lra y 2da -> 3ra (Tabla 3)
conf5 NUMERIC DEFAULT 0;  -- Resultado del soporte para CALCULAR la confianza: lra y 3ra -> 2da (Tabla 3)
conf6 NUMERIC DEFAULT 0;  -- Resultado del soporte para CALCULAR la confianza: 2da y 3ra -> lra (Tabla 3)
conf7 NUMERIC DEFAULT 0;  -- Resultado del soporte para CALCULAR la confianza: lra (Tabla 2)
conf8 NUMERIC DEFAULT 0;  -- Resultado del soporte para CALCULAR la confianza: 2da (Tabla 2)
-------------
r_conf1 NUMERIC DEFAULT 0;  -- Resultado del cálculo de la confianza para la regla: lra -> 2da y 3ra (Tabla 3)
r_conf2 NUMERIC DEFAULT 0;  -- Resultado del cálculo de la confianza para la regla: 2da -> lra y 3ra (Tabla 3)
r_conf3 NUMERIC DEFAULT 0;  -- Resultado del cálculo de la confianza para la regla: 3ra -> lra y 2da (Tabla 3)
r_conf4 NUMERIC DEFAULT 0;  -- Resultado del cálculo de la confianza para la regla: lra y 2da -> 3ra (Tabla 3)
r_conf5 NUMERIC DEFAULT 0;  -- Resultado del cálculo de la confianza para la regla: lra y 3ra -> 2da (Tabla 3)
r_conf6 NUMERIC DEFAULT 0;  -- Resultado del cálculo de la confianza para la regla: 2da y 3ra -> lra (Tabla 3)
r_conf7 NUMERIC DEFAULT 0;  -- Resultado del cálculo de la confianza para la regla: lra -> 2da (Tabla 2)
r_conf8 NUMERIC DEFAULT 0;  -- Resultado del cálculo de la confianza para la regla: 2da -> lra (Tabla 2)
-------------

regla VARCHAR DEFAULT ''; -- donde guardo las reglas q genero para devolver
```

Figura 7: Ejemplo de declaraciones de variables en la implementación del algoritmo A-Priori

Identificadores:

Los identificadores pueden estar formados por cualesquiera de las 26 letras minúsculas o mayúsculas (A... Z, a... z), los 10 dígitos (0... 9) y el carácter subrayado "_". Debe evitarse el uso de caracteres internacionales (ej: ñ, ü) porque no siempre pueden ser leídos o entendidos correctamente en todos los lugares. No se debe usar el símbolo dólar "$" o la barra invertida "\" en los identificadores.

Sentencias Simples:

Cada línea debe contener como máximo una sentencia. Debe poner un punto y coma ";" al final de cada sentencia simple. Tenga en cuenta que una sentencia de asignación puede resultar en la asignación de una función o de un objeto como literal y en todos los casos como sentencia de asignación debe estar finalizada con un punto y coma.

2.9 Implementación de los algoritmos

2.9.1 Algoritmo A-Priori

La implementación del algoritmo A-Priori se realizó utilizando el pseudocódigo mostrado en la Figura No. 8. La función algoritmo_apriori sólo permite trabajar con tablas que tengan atributos nominales y en la misma no debe haber atributos con valores desconocidos para obtener el resultado deseado.

La función toma como entrada los parámetros de mínimo de soporte, mínimo de confianza, el nombre de la tabla sobre la cual se va a trabajar y la cantidad de reglas que se desea generar. Se devuelve como resultados un conjunto de reglas para los conjuntos de items que cuentan con la condición de ser mayor a los parámetros de mínimo de soporte y mínimo de confianza que son introducidos por el usuario.

A continuación se muestra un fragmento de la función implementada:

```
FOR var3 IN SELECT d,e,f,g FROM tablatemp2
LOOP
  FOR var4 IN SELECT d,e,f,g FROM tablatemp2
    LOOP
      IF (var3.columnal <> var4.columnal and var3.columna2 <> var4.columnal) THEN
        EXECUTE ' SELECT count(*) FROM ' ||$3|| ' WHERE ' ||var3.columnal|| ' = ' || '''' ||var3.valor1|| '''' || ' AND ' ||var3.co

        EXECUTE 'SELECT count(*) FROM tablatemp3 WHERE  i='|| '''' ||var3.columna2|| ''''|| ' AND j ='|| '''' ||var3.valor2||''''||
        EXECUTE 'SELECT count(*) FROM tablatemp3 WHERE  i='|| '''' ||var4.columnal|| ''''|| ' AND j ='|| '''' ||var4.valor1||''''||
        EXECUTE 'SELECT count(*) FROM tablatemp3 WHERE  i='|| '''' ||var3.columnal|| ''''|| ' AND j ='|| '''' ||var3.valor1||''''||

        IF ((cantidad_tercero / cantidad_tuplas)  > min_soporte and (var_tab_3=0 and var_tab_4=0 and var_tab_5 =0)) THEN
          INSERT INTO tablatemp3 VALUES (var3.columnal, var3.valor1, var3.columna2, var3.valor2, var4.columnal, var4.valor1, cantid
        END IF;
      END IF;

      IF (var3.columnal <> var4.columna2 and var3.columna2 <> var4.columna2) THEN
        EXECUTE ' SELECT count(*) FROM ' ||$3|| ' WHERE ' ||var3.columnal|| ' = ' || '''' ||var3.valor1|| '''' || ' AND ' ||var3.co

        EXECUTE 'SELECT count(*) FROM tablatemp3 WHERE  i='|| '''' ||var3.columna2|| ''''|| ' AND j ='|| '''' ||var3.valor2||''''||
        EXECUTE 'SELECT count(*) FROM tablatemp3 WHERE  i='|| '''' ||var4.columna2|| ''''|| ' AND j ='|| '''' ||var4.valor2||''''||
        EXECUTE 'SELECT count(*) FROM tablatemp3 WHERE  i='|| '''' ||var3.columnal|| ''''|| ' AND j ='|| '''' ||var3.valor1||''''||

        IF ((cantidad_tercero / cantidad_tuplas)  > min_soporte and (var_tab_6=0 and var_tab_7=0 and var_tab_8=0)) THEN
          INSERT INTO tablatemp3 VALUES (var3.columnal, var3.valor1, var3.columna2, var3.valor2, var4.columna2, var4.valor2, cantid
        END IF;
      END IF;
    END LOOP;
END LOOP;

-- Ciclo para obtener la confianza de la tabla temporal 2
-- y add a la tabla temporal 4, q es la q se devuelve
FOR var4 IN SELECT d,e,f,g,h FROM tablatemp2
  LOOP
    /*1ra*/ EXECUTE ' SELECT count(*) FROM tablatemp2 WHERE d = '||'''' ||var4.columnal|| ''''|| ' AND e= ' || '''' ||var4.valor1||
    /*2da*/ EXECUTE ' SELECT count(*) FROM tablatemp2 WHERE f = '||'''' ||var4.columna2|| ''''|| ' AND g= ' || '''' ||var4.valor2||

    /*1ra -> 2da*/ r_conf7:= (conf8 / cantidad_tuplas) / conf7;
    regla:= var4.columnal||'='||var4.valor1||'-->'||var4.columna2||'='||var4.valor2;
    IF r_conf7 > min_confianza THEN
```

Figura 8: Código de la implementación del algoritmo A-Priori

2.9.2 Algoritmo Partition

La implementación del algoritmo Partition se realizó utilizando el pseudocódigo mostrado en la Figura No. 9. La función algoritmo_partition sólo permite trabajar con tablas que tengan atributos nominales y en la misma no debe haber atributos con valores desconocidos para obtener el resultado deseado.

La función toma como entrada los parámetros de mínimo de soporte, el nombre de la tabla sobre la cual se va a trabajar y la cantidad de particiones que se desea generar. Se devuelve como resultados un conjunto de reglas para los conjuntos de items que cuentan con la condición de ser mayor al parámetros de mínimo de soporte es introducido por el usuario.

A continuación se muestra un fragmento de la función implementada:

```
WHILE (cantidad_part <> 0)
LOOP
  EXECUTE 'CREATE TEMP TABLE ' ||$2||dif_tab|| ' AS SELECT * FROM ' ||$2|| ' OFFSET '  || selec_tabla  || ' LIMIT '||result_tot
  EXECUTE 'SELECT COUNT(*) FROM '||$2||dif_tab INTO cantidad_tuplastemp; --Cuenta la cant de filas de la tabla(Divisiones) temp
   dif_tab := dif_tab + 1;  --Incrementar la variable dif_tab en 1;
  cantidad_part :=  cantidad_part - 1;  --disminuir la variable i en 1;
  selec_tabla := selec_tabla + result_total_filas;  --Sumarle a la variable b la variable a;
  WHILE (cantidad_column <> 0)  -- Busco en cada una de las tablas tsi el soporte de cada atributo es mayor
  LOOP
    FETCH mi_cursor INTO nombre_column;
    FOR variable IN EXECUTE ' SELECT ' ||nombre_column||  ', COUNT(*) FROM ' ||$2||dif_tab|| ' GROUP BY '||nombre_column
    LOOP
      IF (variable.soporte / cantidad_tuplastemp)  > min_soporte THEN
        INSERT INTO tablatempl VALUES (nombre_column, variable.nombre, (variable.soporte / cantidad_tuplastemp) );
      END IF;
    END LOOP;
    cantidad_column := cantidad_column-1;
  END LOOP;
END LOOP;  --RETURN query SELECT * FROM tablatempl;

cantidad_part:=$3;
dif_tab := 0; -- Diferenciador de las tablas
WHILE (cantidad_part <> 0)
LOOP
  cantidad_part :=  cantidad_part - 1;
  FOR var1 IN SELECT c, a FROM tablatempl -- Ciclo para obtener 5 columna(2_Nom_Colum - 2_Nom_Var - Soporte)
  LOOP
    FOR var2 IN SELECT c, a FROM tablatempl
      LOOP
      IF (var1.columna <> var2.columna  AND var1.valor <> var2.valor) THEN
        EXECUTE ' SELECT COUNT(*) FROM ' ||$2||dif_tab||  ' WHERE '  ||var1.columna|| ' = ' || '''' ||var1.valor|| '''' || ' AN
        EXECUTE 'SELECT COUNT(*) FROM '||$2||dif_tab INTO cantidad_tuplastemp2; --Cuenta la cant de filas de la tabla temp para
        IF (cantidad_segundo / cantidad_tuplastemp2)  > min_soporte THEN
          EXECUTE 'SELECT count(*) FROM tablatempfinal WHERE  d='|| '''' ||var2.columna|| ''''|| 'AND e ='|| '''' ||var2.valor|
            IF var_tab_temp_fin = 0 THEN
              INSERT INTO tablatempfinal VALUES (var1.columna, var1.valor, var2.columna, var2.valor, cantidad_segundo / cantida
            END IF;
        END IF;
```

Figura 9: Código de la implementación del algoritmo Partition

2.10 Integración de los algoritmos al SGBD PostgreSQL

La versión 9.1 de PostgreSQL permite añadir nuevas funcionalidades en el SGBD PostgreSQL, ahora los usuarios pueden crear, cargar, actualizar y administrar fácilmente las docenas de extensiones disponibles utilizando el objeto de base de datos EXTENSION. (21)

Una extensión es un complemento que sirve para la integración de aplicaciones obteniéndose una nueva función, esto le permite a los desarrolladores interactuar con la aplicación y así aumentar la cantidad de funcionalidades que se puedan realizar.

La integración de los algoritmos implementados con el SGBD se va a realizar mediante la creación de una extensión por las ventajas que PostgreSQL brinda para su creación. Entre estas ventajas se encuentra,

que en lugar de ejecutar un script SQL para cargar objetos que estén “separados” en su base de datos, se tendrá la extensión como un paquete que contendrá todos los objetos definidos en ella. Esto trae como beneficio que al actualizar o eliminar la extensión se pueden eliminar todos los objetos utilizando el comando DROP EXTENSION sin necesidad de especificar cada uno de los mismos definidos dentro de ella. Además se cuenta con un repositorio para obtener extensiones y contribuir con éstas.

2.10.1 Creación de la extensión reglas de asociación

Para la creación de la extensión se crean dos archivos, en el primero se definen las características de la extensión y en el segundo los objetos SQL que se desean agregar. Los mismos deben ser ubicados dentro del directorio de la instalación “C:\Archivos de programa\PostgreSQL\9.1\share\extension”.

En el archivo “reglas_asociacion.CONTROL” creado para agregar la extensión donde se cargarán las funciones de los algoritmos implementados se definieron los siguientes parámetros:

- ✓ **Comment:** una breve descripción sobre el contenido de la extensión creada.
- ✓ **Encoding:** El tipo de codificación utilizado.
- ✓ **Default_version:** La versión de la extensión.
- ✓ **Schema:** El esquema donde se almacenarán los objetos creados por la extensión.

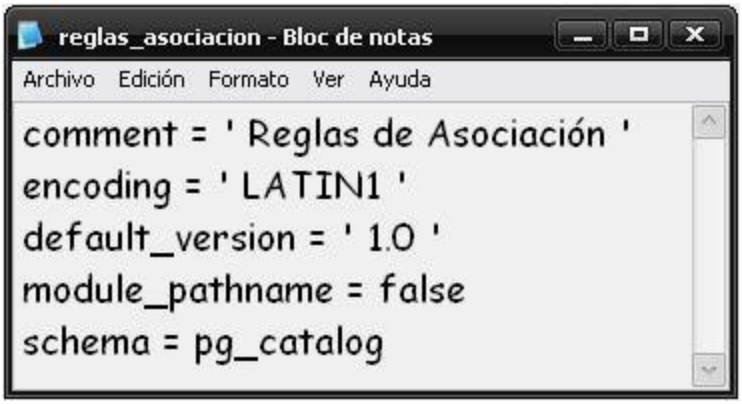


Figura 10: Archivo que contiene las características de la extensión

Una vez definido en el archivo “reglas_asociacion.control” se especifica el archivo que contendrá el código de las funciones desarrolladas “reglas_asociacion--1.0.sql”.

reglas_asociacion--1.0.sql - Bloc de notas

Archivo Edición Formato Ver Ayuda

```
-- Function: algoritmo_apriori(numeric, numeric, character varying, numeric)
-- DROP FUNCTION algoritmo_apriori(numeric, numeric, character varying, numeric);

CREATE OR REPLACE FUNCTION algoritmo_apriori(min_soporte numeric, min_confianza numeric,
nombtabla character varying, cantidad_reglas numeric)
  RETURNS SETOF record AS
$BODY$
DECLARE

mi_cursor refcursor;
var1 tipo_variable;  -- variables del FOR d la tabla temporal 1 para obtener 4 columas
var2 tipo_variable;  -- variables del FOR d la tabla temporal 1 para obtener 4 columas
var3 tipo_var_cuatro;  -- variables del FOR d la tabla temporal 2 para obtener 6 columas
var4 tipo_var_cinco;  -- variables del FOR d la tabla temporal 2 para obtener 6 columas
var5 tipo_var_siete;  -- variables de lal tabla temporal 3
variable nombreresult;  -- Tipo de dato de la 1ra tabla()
nombre_column VARCHAR;  -- Nombre de las columnas existentes en la tabla
nombre_tabla VARCHAR;  -- Nombre de la trabla q se va a trabajar (La pasan por parametro)
cantidad_tuplas NUMERIC DEFAULT 0;  -- Total de tuplas de la tabla
cantidad_column NUMERIC DEFAULT 0;  -- Cantidad de columnas de la tabla
cantidad_segundo INTEGER DEFAULT 0;  -- Cantidad utilizada en la 2da tabla
cantidad_tercero INTEGER DEFAULT 0;  ---- Cantidad utilizada en la 3ra tabla
var_tab_2 INTEGER DEFAULT 0;  -- Varible de la tabla temporal 2
var_tab_3 INTEGER DEFAULT 0;  -- Varible de la tabla temporal 3
var_tab_4 INTEGER DEFAULT 0;  -- Varible de la tabla temporal 4
var_tab_5 INTEGER DEFAULT 0;  -- Varible de la tabla temporal 5
```

Figura 11: Archivo que contiene el código de la extensión

Para que los usuarios puedan utilizar la extensión de reglas de Minería de Datos simplemente deben ejecutar el comando “CREATE EXTENSION reglas_asociacion” que cargará la extensión como se muestra en la siguiente figura:

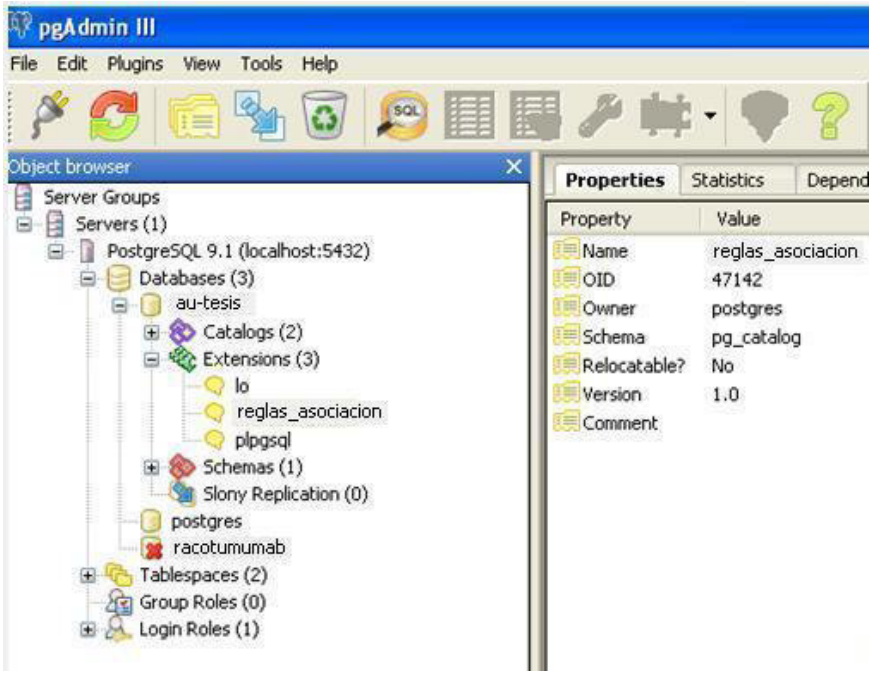


Figura 12: Extensión "reglas_asociacion" creada

2.11 Conclusiones del capítulo

En este capítulo han sido analizados todos los elementos que describen las características y diseño de la extensión. Se elaboró una propuesta con las perspectivas de solucionar el problema identificado, para eso fueron desarrolladas las fases de exploración y diseño propuestas por la metodología utilizada en las que se definieron un total de 9 tareas de la ingeniería agrupadas en 3 historias de usuarios. Se generó además el plan de iteraciones para de esta forma especificar en la iteración que se desarrollará cada historia de usuario y también se realizaron los estándares de codificación. Para finalizar el capítulo se describió cómo a través de la creación de una extensión se integraron los algoritmos al SGDB PostgreSQL 9.1.

CAPÍTULO 3: APLICACIÓN Y VALIDACIÓN DE LA SOLUCIÓN PROPUESTA.

Introducción

En la metodología XP se definen un grupo de normas para la validación de los productos, logrando que los mismos puedan ser probados para la verificación de su correcto funcionamiento. Se analizan en este capítulo las técnicas que definen la metodología XP para diseñar los casos de pruebas que guiarán la validación de la aplicación. Además se realiza un experimento para comparar los resultados de los algoritmos implementados con otra herramienta. Para ello se toman dos juegos de datos diferentes, por un lado se usa los datos del weather.dat y por otro el almacén de ensayos clínicos provenientes del Centro de Inmunología Molecular.

3.1 Métodos de Prueba

Existen disímiles estrategias de pruebas, XP divide las pruebas del sistema en dos grupos: pruebas unitarias, encargadas de verificar el código y diseñada por los programadores, y pruebas de aceptación o pruebas funcionales destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida diseñadas por el cliente final. El objetivo principal de las pruebas es la de evaluar la calidad del producto que se está desarrollando, mediante la aplicación de pruebas concretas para validar que las suposiciones hechas durante las diferentes fases se están cumpliendo satisfactoriamente, esto quiere decir que se verifica que el producto funcione como se diseñó.

3.1.1 Pruebas estructurales o Caja Blanca

Las pruebas de caja blanca realizan un seguimiento del código fuente según se van ejecutando los casos de prueba, de manera que se determinan de forma concreta las instrucciones y bloques en los que existen errores. Conociendo el funcionamiento del producto se pueden desarrollar pruebas de caja blanca, para asegurar que todas las piezas del programa concuerdan, es decir que la operación interna se ajusta a las especificaciones y que todos los componentes internos se han comprobado de forma adecuada, es decir analiza los caminos de ejecución. Mediante los casos de prueba de caja blanca se pueden derivar casos de prueba que garanticen que:

- ✓ Se ejecutan por lo menos una vez todos los caminos independientes de cada módulo.
- ✓ Que se ejercitan todas las decisiones lógicas en sus vértices verdaderos y falsos.

- ✓ Que se ejecutan todos los bucles en sus límites y con sus límites operacionales.
- ✓ Que se ejecutan las estructuras de datos internas para asegurar su validez.

En ocasiones resultan necesarias realizar las pruebas de caja blanca para detectar errores que se producen ante las siguientes situaciones:

- ✓ Los errores lógicos y suposiciones, los cuales son proporcionales a la probabilidad de que se ejecute algún cambio en el programa.
- ✓ Cuando pensamos que un camino lógico tiene pocas posibilidades de ejecutarse y de hecho se ejecuta de forma normal y esto significa o conlleva a que las suposiciones sobre el flujo de control y datos conlleven a errores de diseño y se encuentran solamente cuando comienza la prueba de camino básico.
- ✓ También se pueden encontrar errores tipográficos mediante estas pruebas, lo cual es casi imposible de detectar mediante otro tipo de prueba. (24)

3.1.2 Pruebas de Funcionalidad o Caja Negra

Teniendo en cuenta la función específica para la que se diseñó el producto, se puede llevar a cabo pruebas que demuestran que cada una de las funciones es totalmente operativa y a la vez buscar errores en cada función. Este tipo de prueba se le llama prueba de caja negra, también conocidas como prueba de funcionalidad. Además permite obtener un conjunto de condiciones de entradas que permitan ejercitar totalmente todos los requisitos funcionales del programa. Estas pruebas se llevan a cabo sobre la interfaz del software, y es completamente indiferente el comportamiento interno y la estructura del programa. Mediante los casos de prueba de la caja negra se puede indicar que:

- ✓ Las funciones del software son operativas.
- ✓ La entrada se acepta de forma adecuada.
- ✓ Se produce una salida correcta.
- ✓ La integridad de la información externa se mantiene.

La prueba de la caja negra puede descubrir errores de funciones incorrectas o ausentes entre las que se encuentran:

- ✓ Errores de interfaz.
- ✓ Errores en estructuras de datos o en accesos a bases de datos externas.

- ✓ Errores de rendimiento.
- ✓ Errores de inicialización y terminación.

Las pruebas de caja negra se realizan previamente en el proceso de prueba, esta prueba ignora intencionadamente la estructura de control y centra su atención en el campo de información. Mediante las técnicas de caja negra se obtienen un conjunto de casos de prueba que satisfacen los siguientes criterios:

1. Casos de prueba que reducen, en un coeficiente que es mayor que uno, el número de casos de prueba adicionales que se deben diseñar para alcanzar una prueba razonable.
2. Casos de prueba que digan algo sobre la presencia o ausencia de clases de errores en lugar de errores asociados solamente con la prueba que está realizando.

Se decide aplicar la técnica de caja negra debido a que es más importante presentarle al cliente que las funcionalidades de la aplicación estén correctamente funcionando. A continuación en la Tabla No. 9 se muestra un ejemplo de un caso de prueba que fue utilizado en el algoritmo A-Priori para la realización de las prueba.

Tabla 9: Caso de Prueba: Algoritmo A-Priori

Escenarios	Descripción	Respuesta del Sistema	Flujo Central
EC 1.1 Insertar datos correctos.	En este escenario se crean las reglas de asociación introduciendo datos correctos.	Se muestran las reglas de asociación creadas.	1- El usuario selecciona la función algoritmo_apriori. 2- Introduce los datos requeridos para la función algoritmo_apriori. 3- Se generan las Reglas de asociación para la función algoritmo_apriori.
EC 1.2 Insertar datos incorrectos.	En este escenario se crean las reglas de asociación introduciendo datos incorrectos.	Se muestra un mensaje de error en la herramienta.	
EC 1.3 Insertar datos vacíos.	En este escenario se crean las reglas de asociación dejando datos a insertar vacíos.	Se muestra un mensaje de error en la herramienta.	

3.2 Presentación de los resultados de las pruebas funcionales

Las pruebas han sido aplicadas a las 2 historias de usuarios permitiendo detectar varios errores. Fueron encontradas en una primera iteración 3 no conformidades las cuales fueron resueltas en cuatro días. Para una segunda iteración se encontró 1 no conformidad, la cual fue solucionada en un período de un día. Finalmente para una tercera iteración no se encontraron no conformidades. A continuación se muestra una gráfica donde se recogen la cantidad de no conformidades encontradas y resueltas en cada iteración.

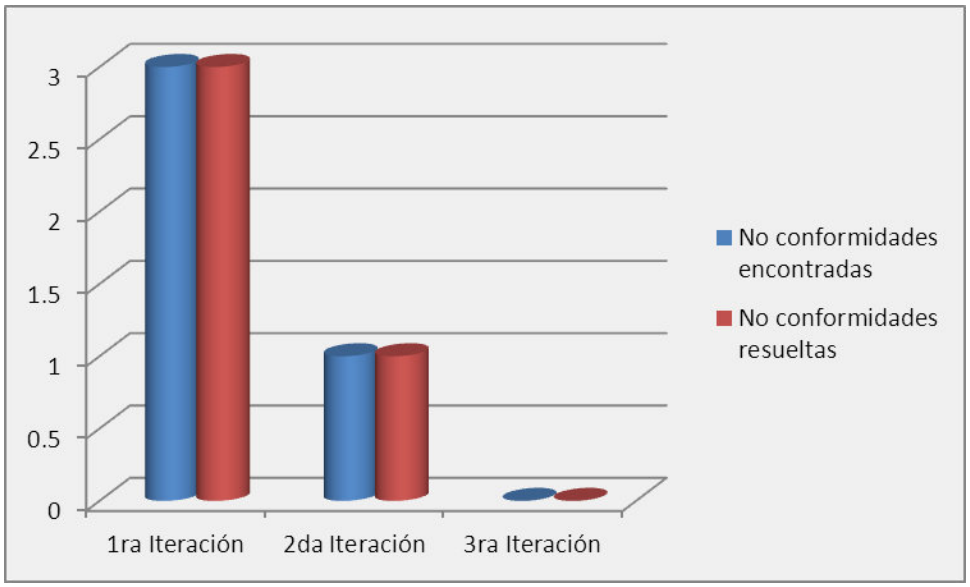


Figura 13: Resultado de las pruebas aplicadas

3.3 Comprensión del negocio

Los **objetivos del negocio** están enmarcados a conocer las relaciones que se establecen entre las características de los pacientes:

- ✓ Relación entre la etapa de la enfermedad e inclusión y evaluación final.
- ✓ Relación entre la evaluación del tratamiento y el tipo de evaluación.
- ✓ Relación entre la evaluación del tratamiento y el hábito toxico.
- ✓ Relación entre la respuesta del tratamiento y la evaluación del tratamiento.
- ✓ Relación entre los signos vitales y la evaluación del tratamiento.
- ✓ Relación entre el examen del laboratorio y la evaluación del tratamiento.
- ✓ Relación entre las evaluaciones clínicas y la técnica de imagenología y radiología.

Como **objetivo de Minería de Datos** se identificó:

Obtener reglas que permitan descubrir la influencia que tiene la evaluación del tratamiento con parámetros como el código la etapa de la enfermedad, el tipo de evaluación de la enfermedad, el hábito toxico del paciente, respuesta del tratamiento recibido, examen del laboratorio, los signos vitales del paciente, el examen físico, la técnica de imagenología y radiología y el tipo de evaluación final.

3.4 Comprensión de los datos

La comprensión de datos está relacionada con la recolección y descripción de la información inicial con la que se comienza el proceso de obtener conocimiento, una vez establecidos los objetivos a seguir. Además, se desarrollan actividades que permiten su exploración, a fin de identificar problemas con su calidad.

La información recopilada fue obtenida de una única fuente, que de manera centralizada es la empleada para recopilar los datos de los pacientes. Este Mercado de Datos cuenta con 35 tablas, de ellas 18 son hechos y 17 son dimensiones. A continuación se describen los atributos significativos para darle solución a los objetivos propuestos:

Tabla 10: Atributos Significativos del Mercado de Datos Racotumumab

Nombre del atributo	Tipo de dato	Descripción	Nombre de la tabla
etapa_enfermedad	character varying	Almacena el identificador de la etapa de la enfermedad de cada uno de los pacientes	racotumumab
evaluacion_enfermedad	character varying	Almacena el estado actual de la enfermedad de cada paciente	racotumumab
habito_toxico_paciente	character varying	Almacena el hábito toxico que tiene cada paciente	racotumumab

respuesta_tratamiento_recibido	character varying	Almacena la respuestas emitida por cada paciente según el tratamiento que recibe	racotumumab
examen_laboratorio	character varying	Almacena el resultado del examen realizado en el laboratorio a cada paciente	racotumumab
signos_vitales_paciente	character varying	Almacena el estado que presentan los signos vitales del paciente	racotumumab
examen_fisico	character varying	Almacena el resultado del examen físico realizado al paciente	racotumumab
imagenologia_radiologia	character varying	Almacena el resultado obtenido de cada paciente	racotumumab
tipo_evaluacion_final	character varying	Almacena el resultado de la evaluación final realizada a cada paciente	racotumumab

3.5 Implementación y evaluación

El objetivo del negocio corresponde al descubrimiento de patrones ocultos en los datos, esto permite clasificar la evaluación del tratamiento aplicada a los pacientes, basado en las relaciones que se establecen entre los atributos seleccionados. Por lo que pueden considerarse los modelos como aceptados, desde el punto de vista analítico, para apoyar la toma de decisiones administrativas del Centro de Inmunología Molecular.

Para la implementación del proyecto los directivos del Centro de Inmunología Molecular son los encargados de emprender acciones y determinar, si así lo estiman conveniente, una estrategia a seguir, basada en la información descubierta por los algoritmos. Además se generará un informe con todo el

proceso de minería, que servirá de apoyo o consulta para el proceso administrativo y la toma de decisiones.

La integración de la extensión de Minería de Datos al Sistema Gestor de Base de Datos le permite a los especialistas de genética médica llamar a las funciones de los algoritmos de reglas de asociación y ante cualquier actualización de los datos de la aplicación se puede volver a realizar un análisis de la información para obtener conocimiento desde la aplicación de genética médica.

A continuación se realiza un experimento al mercado de datos racotumumab para validar los resultados obtenidos en esta investigación.

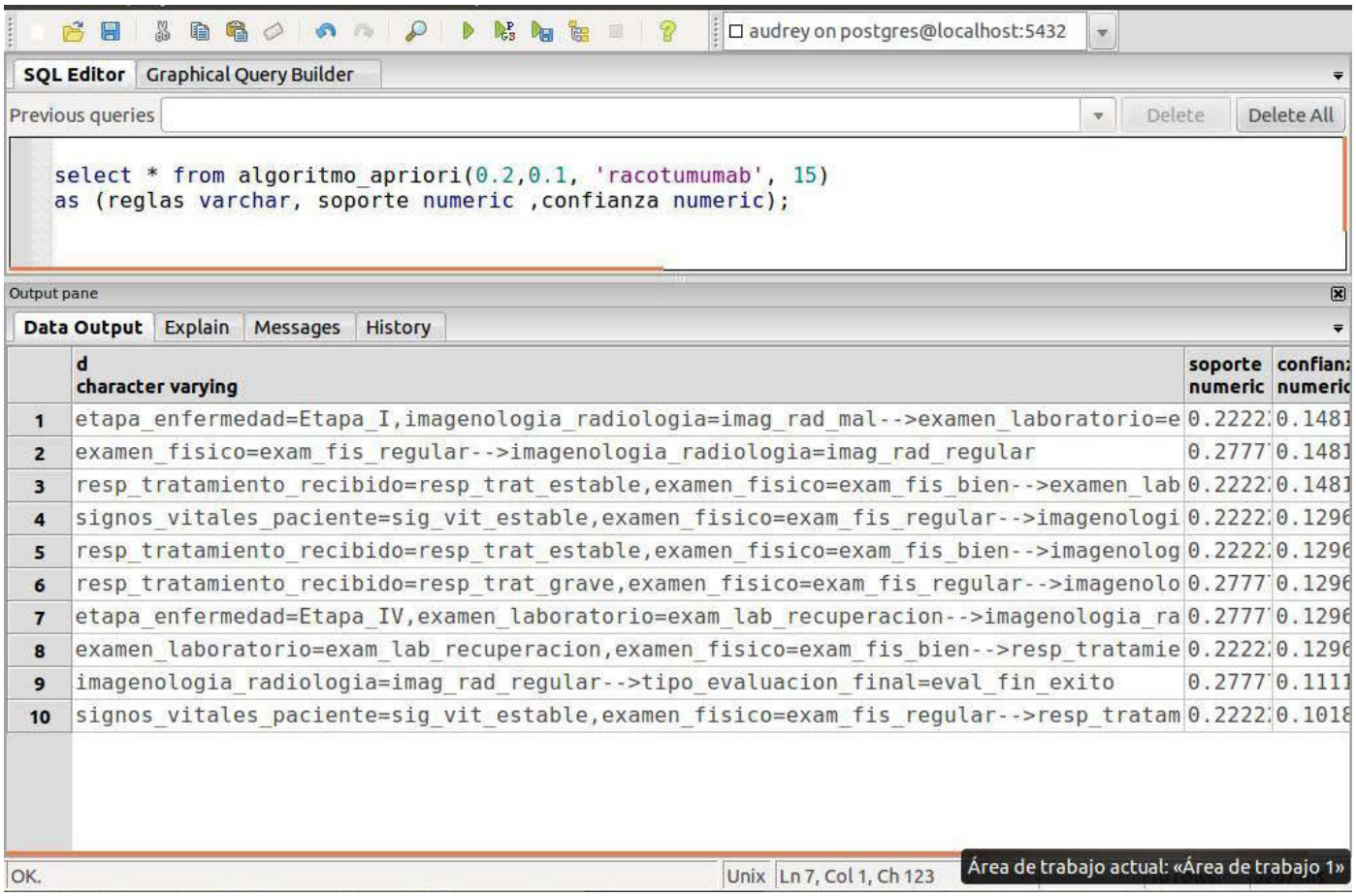


Figura 14: Resultado obtenido del algoritmo A-Priori en el SGBD PostgreSQL 9.1

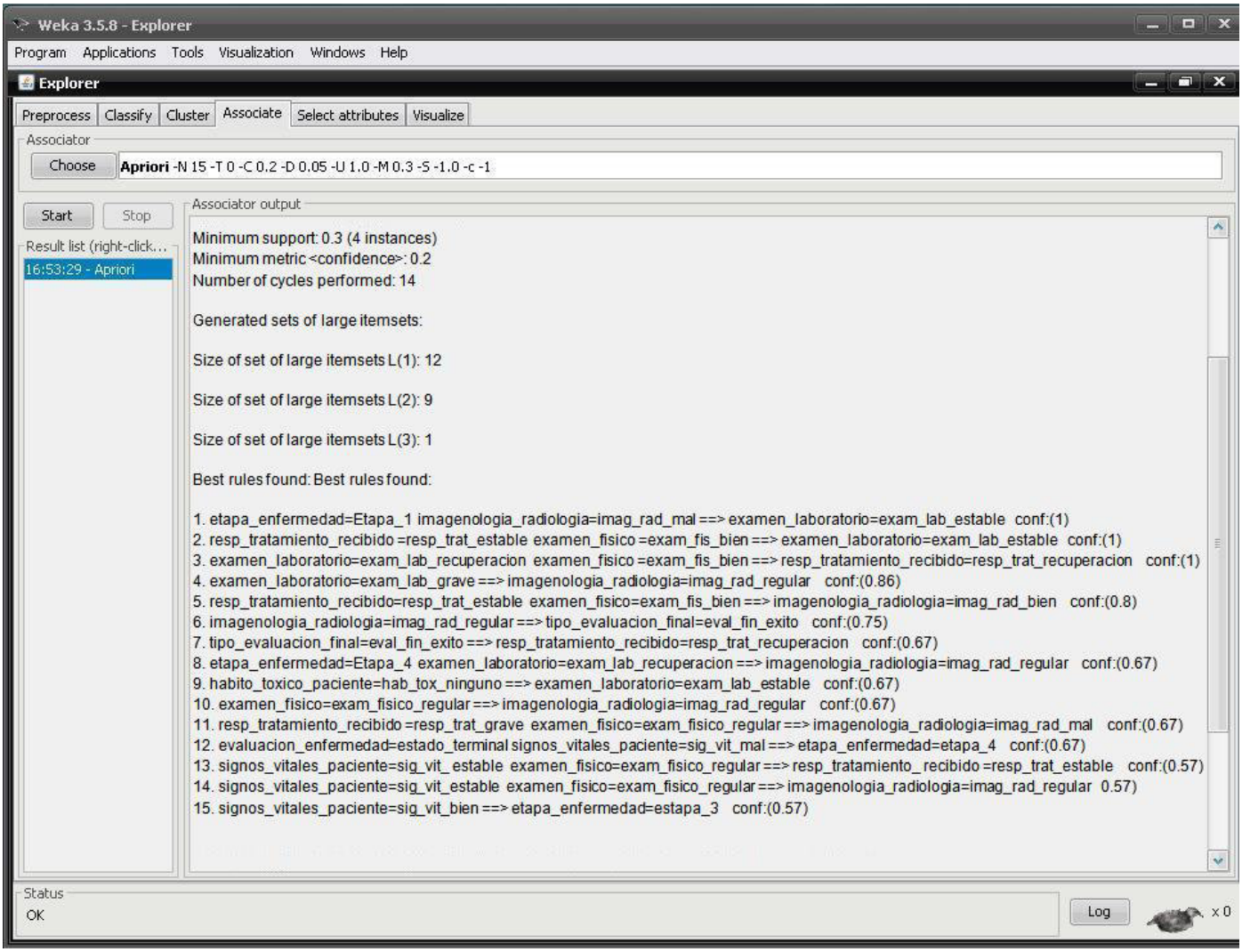


Figura 15: Resultado obtenido del algoritmo A-Priori en el Weka

Al realizar el análisis del mercado de datos racotumumab los resultados en el SGBD PostgreSQL 9.1 se obtuvieron en 19.205 segundos. Con la herramienta de Minería de Datos Weka los resultados se obtuvieron en 16.404 segundos. Con una diferencia de 5 reglas generadas, lo cual tiene un nivel de aceptación de un 67%.

3.6 Conclusiones del capítulo

En el presente capítulo se realizaron las pruebas de funcionalidad a la aplicación del proceso de Minería de Datos al Mercado de Datos Racotumumab del Centro de Inmunología Molecular, donde se analizaron los campos importantes para darle cumplimiento a los objetivos de negocio y Minería de Datos propuestos. Para la integración de la información se realizó una tabla como resultado de una vista a la cual se le aplicaron los algoritmos, obteniendo como resultado un conjunto de reglas que permiten clasificar la evaluación del tratamiento realizado. Además se realizó la validación de los algoritmos integrados al SGBD PostgreSQL 9.1, con el cual se demostró que la integración de los algoritmos al mismo permite aprovechar sus potencialidades.

CLONCLUSIONES GENERALES

De los resultados obtenidos en esta investigación se puede decir que la técnica de Minería de Datos reglas de asociación permite obtener como resultado final reglas que por sus características son unas de las formas de representar más divulgadas y unas de la más compresivas de entender por las personas.

Además se pudo evidenciar que las herramientas libres existentes de Minería de Datos tienen el inconveniente de ser independientes del SGBD por lo cual se implementaron dos de los algoritmos de las técnicas de reglas de asociación y se integraron al SGBD PostgreSQL 9.1 a través de la creación de una extensión, lo que contribuye a la soberanía tecnológica del país y a que el gestor sea más competitivo.

Se realizó la validación de los algoritmos implementados utilizando pruebas funcionales, y se aplicó la solución en el Mercado de Datos Racotumumab.

RECOMENDACIONES

Integrar el algoritmo Eclat de Minería de Datos al SGBD PostgreSQL, el cual pertenece a las Reglas de Asociación, pues este algoritmo es tan descriptivo como los utilizados en la investigación y está entre los más utilizados.

TRABAJOS CITADOS

1. **Aranda, Yadira Robles**. *Algoritmos de Minería de Datos: Árboles de decisión y reglas de Inducción Integrados a PostgreSQL*, 2012.

2. **José Manuel Molina López, Jesús García Herrero.** *Técnicas de análisis de datos: Aplicaciones prácticas utilizando Microsoft Excel y Weka*, 2011.

3. **Monteserin, Ariel.** *Agrupamiento de datos,* 2009.

4. **Ussama M, Fayyad.** *Advances in Knowledge Discovery and Data Mining,* 1996.

5. **Serventes, M.** *Algoritmos TDIDT aplicados a la Minería de Datos Inteligente.* Universidad de Buenos Aires: Facultad de Ingeniería, Laboratorio de Sistemas Inteligentes, 2008.

6. **Ansel Yoan Rodríguez González, José Francisco Martínez Trinidad, Jesús Ariel Carrasco Ochoa, José Ruiz Shulcloper.** *Minería de Reglas de Asociación sobre Datos Mezclados,* 2009.

7. *Reporte Técnico de Mineria de Datos: Serie Gris.* s.l. : http://wwwcenatav.co.cu/doc/RTecnicos/RT%20SerieGris_003web.pdf, 2010.

8. **C. Fernández, M. A. Vicente, A. Gil, L. M. Jiménez, R. Ñeco.** *Integración de Matlab y Weka para la docencia en asignaturas de aprendizaje automático (Machine Learning)*. España: Universidad Miguel Hernández de Elche, 2010.

9. **Lumpkin, George**. *Oracle Database para Data Warehousing e Inteligencia de Negocios.* s.l. Informe Ejecutivo de Oracle, 2010.

10. *Oracle Advanced Analytics.* oracle.com. [En línea] Oracle, 2012. http://www.oracle.com/es/products/database/options/advanced-analytics/index.html.

11. **P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, R. Wirth.** *CRISP-DM Step by step data mining guide*, 2009.

12. **Arencibia, José Alberto Gallardo.** *Metodología para el desarrollo de proyectos en Minería de Datos CRISP-DM,* 2009.

13. **Karina Collazo Oliva, Yordailis Morales Díaz.** *Propuesta para la utilización de la Minería de Datos en la recuperación de información en la sección Vigilancia Tecnológica de D´TIC.* s.l. : Repositorio Institucional, 2010.

14. **Schwaber K., Beedle M., Martin R.C.** *Agile Software Development with SCRUM*. Prentice Hall, 2001.

15. **Cockbun, A.** *Agile Software Development.* Addison-Wesley, 2011.

16. **Stapleton J.** Dsdm Dynamic Systems Development Method: The Method in Practice. Addison-Wesley, 1997.

17. **Highsmith J., Orr K**. Adaptive Software Development: A Collaborative Approach to Managing Complex Systems. Dorset House, 2008.

18. **Coad P., Lefebvre E., De Luca J.** Java Modeling In Color With UML: Enterprise Components and Process. Prentice Hall, 2009.

19. **Poppendieck M., Poppendieck T.** *Lean Software Development: An Agile Toolkit for Software Development Managers*. Addison Wesley, 2009.

20. **Patricio Letelier, Carmen Penadés.** *Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP)*. Universidad Politécnica de Valencia (UPV): Departamento de Sistemas Informáticos y Computación (DSIC), 2011.

21. PostgreSQL. PostgreSQL.org. [En línea] PostgreSQL 9.1 Press Kit, 12 de Septiembre de 2011. [Citado el: 7 de Enero de 2012.] http://www.postgresql.org/about/press/presskit91/es/.

22. *Guía documentada para Ubuntu: PgAdminIII.* Ubuntu, 2011.

23. **Domínguez Rodríguez, Karel Manuel y Téllez Sánchez, Lino.** *Sistema de apoyo a la toma de decisiones en el proceso de negociación comercial*. Holguín, Cuba: s.n, 2011.

24. **PRESSMAN, R. S**. *Ingeniería del Software .Un enfoque práctico.Quinta Edición,* 2008.

BIBLIOGRAFÍA

Aranda, Yadira Robles. *Algoritmos de Minería de Datos: Árboles de decisión y reglas de Inducción Integrados a PostgreSQL*, 2012.

José Manuel Molina López, Jesús García Herrero. *Técnicas de análisis de datos: Aplicaciones prácticas utilizando Microsoft Excel y Weka*, 2011.

Monteserin, Ariel. *Agrupamiento de datos,* 2009.

Ussama M, Fayyad. *Advances in Knowledge Discovery and Data Mining,* 1996.

Serventes, M. *Algoritmos TDIDT aplicados a la Minería de Datos Inteligente.* Universidad de Buenos Aires: Facultad de Ingeniería, Laboratorio de Sistemas Inteligentes, 2008.

Ansel Yoan Rodríguez González, José Francisco Martínez Trinidad, Jesús Ariel Carrasco Ochoa, José Ruiz Shulcloper. *Minería de Reglas de Asociación sobre Datos Mezclados,* 2009.

Reporte Técnico de Mineria de Datos: Serie Gris. s.l. : http://wwwcenatav.co.cu/doc/RTecnicos/RT%20SerieGris_003web.pdf, 2010.

C. Fernández, M. A. Vicente, A. Gil, L. M. Jiménez, R. Ñeco. *Integración de Matlab y Weka para la docencia en asignaturas de aprendizaje automático (Machine Learning)*. España: Universidad Miguel Hernández de Elche, 2010.

Lumpkin, George. *Oracle Database para Data Warehousing e Inteligencia de Negocios.* s.l. Informe Ejecutivo de Oracle, 2010.

Oracle Advanced Analytics. oracle.com. [En línea] Oracle, 2012. http://www.oracle.com/es/products/database/options/advanced-analytics/index.html.

P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, R. Wirth. *CRISP-DM Step by step data mining guide*, 2009.

Arencibia, José Alberto Gallardo. *Metodología para el desarrollo de proyectos en Minería de Datos CRISP-DM,* 2009.

Karina Collazo Oliva, Yordailis Morales Díaz. *Propuesta para la utilización de la Minería de Datos en la recuperación de información en la sección Vigilancia Tecnológica de D´TIC.* s.l. : Repositorio Institucional, 2010.

Schwaber K., Beedle M., Martin R.C. *Agile Software Development with SCRUM*. Prentice Hall, 2001.

Cockbun, A. *Agile Software Development.* Addison-Wesley, 2011.

Stapleton J. Dsdm Dynamic Systems Development Method: The Method in Practice. Addison-Wesley, 1997.

Highsmith J., Orr K. Adaptive Software Development: A Collaborative Approach to Managing Complex Systems. Dorset House, 2008.

Coad P., Lefebvre E., De Luca J. Java Modeling In Color With UML: Enterprise Components and Process. Prentice Hall, 2009.

Poppendieck M., Poppendieck T. *Lean Software Development: An Agile Toolkit for Software Development Managers*. Addison Wesley, 2009.

Patricio Letelier, Carmen Penadés. *Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP"*. Universidad Politécnica de Valencia (UPV): Departamento de Sistemas Informáticos y Computación (DSIC), 2011.

PostgreSQL. PostgreSQL.org. [En línea] PostgreSQL 9.1 Press Kit, 12 de Septiembre de 2011. [Citado el: 7 de Enero de 2012.] http://www.postgresql.org/about/press/presskit91/es/.

Guía documentada para Ubuntu: PgAdminIII. Ubuntu, 2011.

Domínguez Rodríguez, Karel Manuel y Téllez Sánchez, Lino. *Sistema de apoyo a la toma de decisiones en el proceso de negociación comercial*. Holguín, Cuba: s.n, 2011.

PRESSMAN, R. S. *Ingeniería del Software .Un enfoque práctico.Quinta Edición,* 2008.

S. Richard, J. L. Vargas*. Revista Iberoamericana de Inteligencia Artificial: Minería de Datos Conceptos y Tendencias.* 25. 925, 2010.

Varian, Hal. Data: data everywhere. *The Economist.* Special Reports. 2010, 22.

Institute, McKinsey Global. *Big Data: the next frontier for innovation, competition and productivity*. Special Reports, 2011, 05.

Pérez, Beatriz. *ProTest-Proceso de Testing Funcional,* 2009.

Falcon Rodriguez, Yolanda y Leyva Osorio,Reynaldo. *Mercado de Datos Estadístico de Inmigración y Extranjería para el Departamento de Turismo y Comercio de la Oficina Nacional de Estadísticas.* Habana: s.n, 2010

Soto Jaramillo, C. M. *Incorporación de Técnicas Multivariantes en un Sistema Gestor de Bases de Datos Incorporadas,* 2009.