# ARQUITECTURAS DE INTEGRACIÓN DEL PROCESO DE DESCUBRIMIENTO DE CONOCIMIENTO CON SISTEMAS DE GESTIÓN DE BASES DE DATOS: UN ESTADO DEL ARTE

Ricardo Timarán Pereira*

**RESUMEN**

Las investigaciones en Descubrimiento de Conocimiento en Bases de Datos (DCBD), se centraron inicialmente en definir modelos de descubrimiento de patrones y desarrollar algoritmos para éstos. Investigaciones posteriores se han focalizado en el problema de integrar DCBD con sistemas de bases de datos, produciendo como resultado el desarrollo de sistemas y herramientas de Descubrimiento de Conocimiento cuyas arquitecturas se pueden clasificar en tres categorías: débilmente, medianamente y fuertemente acopladas con un Sistema de Gestión de Bases de Datos (SGBD).

* Master of Science en Ingeniería - Universidad Politécnica de Donetsk (Ucrania).
Especialista en Multimedia Educativa Universidad Antonio Nariño
Candidato a Doctor en Ingeniería - Universidad del Valle.

Profesor Asistente del Departamento de Ingeniería de Sistemas - Universidad del Nariño

En este artículo se presenta una revisión del estado del arte de las arquitecturas de integración del proceso de Descubrimiento de Conocimiento con SGBD que forma parte de la propuesta de investigación doctoral denominada "Nuevas primitivas SQL para el Descubrimiento de Conocimiento en Arquitecturas Fuertemente Acopladas con un SGBD" que actualmente está desarrollando el autor de este artículo en el Doctorado en Ingeniería, área de énfasis Ciencias de la Computación de la Universidad del Valle.

Palabras claves: Descubrimiento de Conocimiento en Bases de Datos, Minería de Datos.

## ABSTRACT

Researches on Knowledge Discovery in Databases (KDD) was initially oriented toward the definition of new pattern discovery models and the development of the corresponding algorithms. At present, research has focused on issues related to integrating KDD with database systems, to generate systems and tools for KDD whose architectures can be classified in one of three categories: loosely coupled, middly coupled and tightly coupled with a Database Management System (DBMS).

In this paper a review of the state of the art on architectures for the process of integrating Knowledge Discovery with a DBMS is presented. It is part of the proposal doctoral research "New primitives SQL for Knowledge Discovery on tightly coupled architectures with a DBMS" that at the moment it is developed by the author of this paper in the program of Ph.D. in Engineering in Computer Science area of emphasis of University of Valley.

Keywords:Knowledge Discovery in Databases, Data Mining.

## 1. INTRODUCCIÓN

El explosivo crecimiento en los volúmenes de los datos y las bases de datos que superan los métodos tradicionales de análisis basados en hojas de cálculo y consultas ad-hoc [11], ha generado una urgente necesidad de contar con nuevas técnicas y herramientas que puedan, inteligente y automáticamente, transformar los datos en información útil: en conocimiento [8]. Estas técnicas y herramientas son el objeto del emergente campo de *Descubrimiento de Conocimiento en Bases de Datos* (DCBD).

DCBD es el proceso no trivial de identificación de patrones válidos, novedosos, potencialmente útiles y fundamentalmente entendibles al usuario a partir de los datos [12]. El proceso es interactivo e iterativo e involucra numerosos pasos (Selección, Preprocesamiento/ Limpieza de Datos, Transformación/Reducción, Minería de Datos, Interpretación / Evaluación) con la intervención del usuario en la toma de muchas decisiones. La principal etapa de este proceso es la de Minería de Datos en donde se lleva a cabo el descubrimiento de patrones [13].

Una herramienta para el descubrimiento de conocimiento en bases de datos debe integrar una variedad de componentes (técnicas de minería de datos, consultas, métodos de visualización, interfaces, etc.), que juntos puedan eficientemente identificar y extraer patrones interesantes y útiles de los datos almacenados en las bases de datos.

Muchos investigadores [7], [35], [3], [16], [21], [25] han reconocido la necesidad de integrar los sistemas de descubrimiento de conocimiento y bases de datos, haciendo de ésta una área activa de investigación. Los enfoques de integración de DCBD y SGBD reportados en la literatura se pueden ubicar en uno de tres tipos de arquitectura: sistemas débilmente acoplados, medianamente acoplados y fuertemente acoplados.

En este artículo se presenta una revisión del estado del arte de las arquitecturas de integración del proceso de Descubrimiento de Conocimiento con SGBD. Está organizado en secciones. En la sección 2 se analiza la arquitectura de descubrimiento de conocimiento débilmente acoplada con un SGBD. En la sección 3 se especifican los sistemas DCBD medianamente acoplados. En la sección 4 se presenta la integración fuerte entre un sistema de descubrimiento de conocimiento y un SGBD y finalmente en la sección 5 se presentan algunas conclusiones.

## 2. ARQUITECTURA DCBD DÉBILMENTE ACOPLADA

Una arquitectura es débilmente acoplada cuando los algoritmos de Minería de Datos y demás componentes se encuentran en una capa externa al SGBD, por fuera del núcleo y su integración con éste se hace a partir de una interfaz cuya función, en la mayoría de los casos se limita a los comandos "leer de " y "escribir en" [21].

En una arquitectura débilmente acoplada, los procesos de minería de datos corren en un espacio de direccionamiento diferente al del SGBD [35], [36]. Mientras el SGBD provee el almacenamiento persistente, la mayor parte del procesamiento de datos se realiza en herramientas y aplicaciones por fuera del motor del SGBD (ver figura 2.1).

Esta arquitectura está presente en la mayoría de las herramientas de DCBD[35]. Algunas como Alice [23], C5.0_RuleQuest [32], Qyield [30], CoverStory [25] ofrecen soporte únicamente en la etapa de minería de datos y requieren un pre y un pos procesamiento de los datos. Hay una gran cantidad de este tipo de herramientas [24], especialmente para clasificación apoyadas en árboles de decisión, redes neuronales y aprendizaje basado en ejemplos. El usuario de este tipo de herramientas puede integrarlas a otros módulos como parte de una aplicación completa [29].

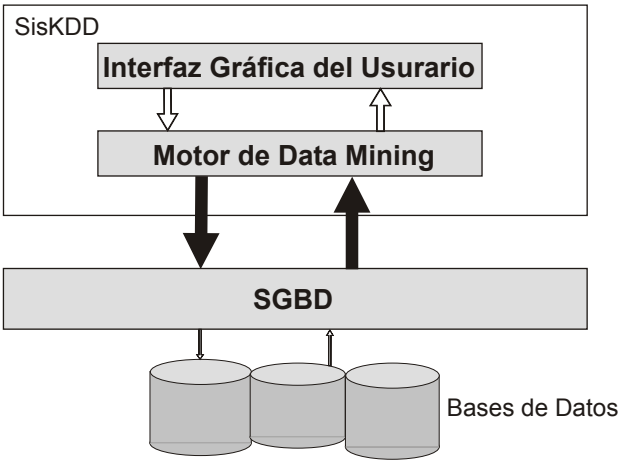


**Figura 2.1. Arquitectura DCBD débilmente acoplada**

Otros ofrecen soporte en más de una etapa del proceso de DCBD y una variedad de tareas de descubrimiento, típicamente, combinando clasificación, visualización, consulta y clustering, entre otras [29]. En este grupo están Clementine [33], DBMiner [15], [16], [18], DBLearn [19], Data Mine [21], IMACS [6], Intelligent Miner [20], Quest [3] entre otras. Una evaluación de un gran número de herramientas de este tipo se puede encontrar en [14].

La implementación de este tipo de arquitectura se hace a través de SQL embebido en el lenguaje anfitrión del motor de minería de datos [4], [5], [35], [36]. Los datos residen en el SGBD y son leídos registro por registro a través de ODBC, JDBC o de una interfaz de cursores SQL. La ventaja de esta arquitectura es su portabilidad. Sus principales desventajas son la escalabilidad y el rendimiento. El problema de escalabilidad consiste en que las herramientas y aplicaciones bajo este tipo de arquitectura, cargan todo el conjunto de datos en memoria, lo que las limita para el manejo de grandes cantidades de datos. El bajo rendimiento se debe a que los registros son copiados uno por uno del espacio de direccionamiento de la base de datos al espacio de direccionamiento de la aplicación de minería

de datos [4] [5] y estas operaciones de entrada/salida, cuando se manejan grandes volúmenes de datos, son bastante costosas, a pesar de la optimización de lectura por bloques presente en muchos SGBD (Oracle, DB2, Informix, etc.) donde un bloque de tuplas puede ser leído al tiempo.

Estos sistemas, a pesar de que son fáciles de integrar a cualquier SGBD, son ineficientes en términos computacionales. El problema radica en que los costosos algoritmos de descubrimiento de conocimiento se implementan por fuera del núcleo del SGBD, impidiendo cualquier aplicación de las técnicas de optimización de consultas por parte del optimizador del SGBD.

## 3. ARQUITECTURA DCBD MEDIANAMENTE ACOPLADA

Una arquitectura es medianamente acoplada cuando ciertas tareas y algoritmos de descubrimiento de patrones se encuentran formando parte del SGBD mediante procedimientos almacenados o funciones definidas por el usuario.

En una arquitectura medianamente acoplada las principales tareas de descubrimiento de conocimiento se descomponen en subtareas, algunas de las cuales se mueven al SGBD como consultas SQL en procedimientos almacenados o funciones definidas por el usuario (ver figura 3.1). La principal ventaja de esta arquitectura es que tiene en cuenta las capacidades de escalabilidad, administración y manipulación de datos del SGBD. Su mayor reto es obtener un buen rendimiento ya que el optimizador de consultas del SGBD no cuenta con nuevas estrategias de búsqueda y transformación que le permitan generar eficientes planes de ejecución, para las consultas que involucren descubrimiento de conocimiento. Este problema se puede solucionar, en algún grado, implementando algoritmos ingeniosos, pero que incrementan la complejidad y el costo de desarrollo de estos sistemas [10].

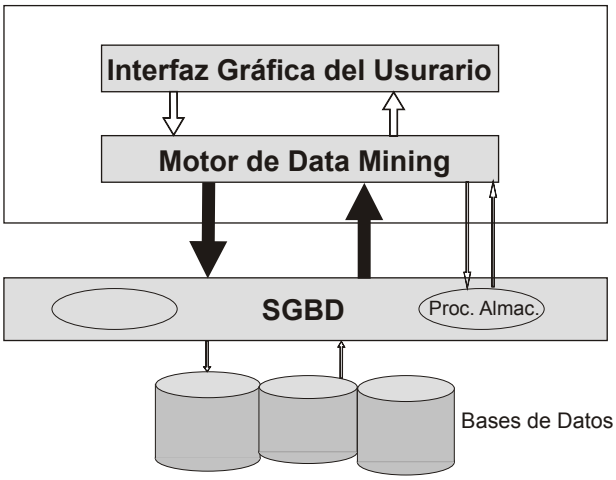


**Figura 3.1. Arquitectura DCBD medianamente acoplada**

En la arquitectura medianamente acoplada a través de procedimientos almacenados [35], [36], [10], los algoritmos de descubrimiento son encapsulados como procedimientos almacenados que corren en el mismo espacio de direccionamiento del SGBD. Un procedimiento almacenado es un conjunto nombrado de instrucciones y lógica de procedimientos de SQL compilado, verificado y almacenado en la base de datos. El motor de minería de datos invoca al procedimiento almacenado y le transmite los parámetros requeridos para hacer una tarea. Un solo mensaje desencadena la ejecución de un conjunto de instrucciones SQL almacenadas. La ventaja de este método es que brinda mejor desempeño, ya que ejecutan todas las conexiones, aplicaciones y la base de datos en el mismo espacio de direccionamiento, además, un procedimiento almacenado recibe igual trato que cualquier otro objeto de la base de datos y es registrado en su catálogo.

En la arquitectura medianamente acoplada por medio de funciones definidas por el usuario (FDUs) [35], [36], [5], [10], las funciones son definidas e implementadas en un lenguaje de programación de propósito general. El ejecutable de una FDU se almacena en el SGBD y

éste puede acceder e invocar la función en el momento que ésta sea referenciada en una instrucción SQL. Los algoritmos de Minería de Datos se expresan como una colección de funciones definidas por el usuario. La mayoría del procesamiento se realiza en la FDU y el SGBD se usa principalmente para proveer tuplas hacia las FDUs. Las FDUs, al igual que los procedimientos almacenados, corren en el mismo espacio de direccionamiento que el SGBD. La principal ventaja de las FDUs sobre los procedimientos almacenados es su tiempo de ejecución ya que pasar tuplas a un procedimiento almacenado es más lento que hacia una FDU, debido a que en un procedimiento almacenado se deben ejecutar una serie de instrucciones SQL y en una FDU no. El resto del procesamiento es el mismo. La principal desventaja es el costo de desarrollo ya que los algoritmos de minería de datos tienen que ser escritos como FDUs.

Un análisis de este tipo de arquitectura se reporta en [35]. En este trabajo, el algoritmo Apriori [1], [2] se implementa de varias maneras en SQL, e.g. usando únicamente SQL-92 y usando SQL con extensiones objeto-relacional (OR) tales como BLOBS(*binary large objects*) y funciones definidas por el usuario. De acuerdo con los autores, las pruebas de rendimiento mostraron que usando únicamente SQL-92 el algoritmo Apriori fue demasiado lento. El rendimiento mejoró a niveles aceptables usando características OR, pero con una implementación demasiado compleja y a un alto costo de desarrollo.

## 4. ARQUITECTURA DCBD FUERTEMENTE ACOPLADA

Una arquitectura es fuertemente acoplada cuando la totalidad de las tareas y algoritmos de descubrimiento de patrones forman parte del SGBD como una operación primitiva, dotándolo de las capacidades de descubrimiento de conocimiento y posibilitándolo para desarrollar aplicaciones de este tipo (ver figura 4.1).

En una arquitectura fuertemente acoplada, un algoritmo completo de descubrimiento de conocimiento se integra al motor de un SGBD como una primitiva. Debido a que todos los algoritmos son ejecutados conjuntamente con los datos en el SGBD, la ventaja potencial de este enfoque es que resuelve los problemas de escalabilidad y rendimiento de las otras arquitecturas. Tal vez la mayor limitación de este método es que los desarrolladores de aplicaciones y herramientas DCBD no están dispuestos a poner algoritmos completos en los sistemas de bases de datos por razones de competencia. Por otro lado, muchos algoritmos cambian frecuentemente con los resultados de las nuevas investigaciones, lo que dificulta su oportuna actualización [10].

Hay propuestas de investigación que discuten la manera cómo tales sistemas pueden ser implementados y las extensiones del lenguaje SQL necesarias para soportar estas operaciones de descubrimiento de patrones: DMQL [17], M-SQL [22], MINE RULE [26], [27] y NonStop SQL/MX [9], [10]. Sin embargo, a la fecha no existe un consenso general sobre el conjunto de primitivas necesarias para soportar el descubrimiento de conocimiento en un SGBD. Este punto no ha sido lo suficientemente investigado aún. La búsqueda de tales primitivas continuará en el futuro cercano y será un área activa de investigación [9], [10], [21].

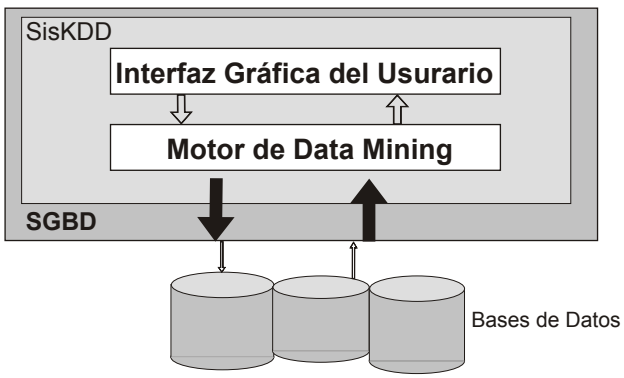


**Figura 4.1. Arquitectura DCBD fuertemente acoplada**

Con relación a las propuestas de extensión del lenguaje SQL, Han et al. [17] proponen el DMQL (Data Mining Query Language), un lenguaje de consultas de minería de datos para bases de datos relacionales, el cual adopta una sintaxis *SQL-like* para facilitar el alto nivel de minería de datos y una natural integración con el lenguaje relacional SQL.

DMQL extiende el lenguaje SQL con una colección de operadores para generalización, reglas caracterizantes, reglas discriminantes, reglas de clasificación y reglas de asociación con varias clases de umbrales (*thresholds)*. DMQL puede servir como un lenguaje de desarrollo para implementaciones de sistemas de minería de datos.
Imielinski et al. [22], proponen el M-SQL, un lenguaje de consulta que extiende el SQL con un conjunto pequeño de primitivas para minería de datos en un operador especial unificado, el operador MINE. MINE genera y recupera todo un conjunto de reglas que cumplen con el soporte y la confianza establecidas por el usuario. M-SQL se puede incrustar en el lenguaje anfitrion C++ (de manera similar como SQL es embebido en C) para proveer APIs (Application Programming Interface) para aplicaciones de descubrimiento de conocimiento.

La principal ventaja de estas dos propuestas: DMQL y M-SQL, es que extienden el lenguaje SQL con nuevos operadores para poder expresar operaciones de Minería de Datos con una sintaxis SQL. La desventaja de estas propuestas es la arquitectura bajo la cual estos lenguajes fueron implementados.

El lenguaje DMQL está implementado en el sistema DBMiner [15], [16], [18], un sistema para Minería de Datos débilmente acoplado con SGBD relacional. La arquitectura de DBMiner consta de una interfaz gráfica de usuario(GUI), un motor de descubrimiento y un módulo de comunicación de datos. El motor de descubrimiento contiene módulos para el análisis de consultas, generalización y descubrimiento de patrones. El módulo de comunicación de datos permite el intercambio de datos entre el motor de descubrimiento y el servidor SQL [15].

De igual forma, M-SQL hace parte del prototipo Data Mine [22], un sistema para Minería de Datos que, de acuerdo a sus autores [22], es débilmente acoplado con un SGBD. La arquitectura de Data Mine consta de un motor de minería de datos, una interfaz gráfica de usuario y una interfaz de comunicación de datos. Data Mine puede leer datos a partir de archivos ASCII o desde una base de datos directamente. El SGBD usado actualmente es Sybase, pero el diseño asegura un acoplamiento débil entre la base de datos y el motor de minería de datos, lo que hace posible el uso de otros sistemas gestores de bases de datos. Data Mine provee además, una interfaz de programación de aplicaciones para *Minería de Datos*.

Meo et al. [26], proponen un modelo de operador unificado *SQL-like* (i.e. con una sintáxis parecida al *SQL*) para encontrar reglas de asociación en datos agrupados por diferentes atributos. El operador MINE RULE se diseña como una extensión del lenguaje SQL para obtener diferentes tipos de reglas de asociación: reglas de asociación simples, con condicionales, con clustering, con generalización, con jerarquías. El operador MINE RULE produce una nueva tabla donde cada tupla corresponde a una regla descubierta.

Los autores proponen una semántica formal para el operador MINE RULE. La semántica se describe por medio de una extensión del álgebra relacional con nuevos operadores: Group by ( G ), Unnest ( h ), Extended ( E ), Substitute ( å ), Rename ( r ), Powerset ( R ), que permiten transformar una relación con el fin de descubrir reglas de asociación [26] [27]. Además, proponen una arquitectura, que a juicio de los autores, está fuertemente acoplada con *SQL server*, para soportar el operador MINE RULE [28].

El sistema se divide en tres módulos

fundamentales: La *interfaz del usuario*; el *kernel de Minería de Datos*, que ejecuta la extracción de las reglas y el *SGBD*, que almacena los datos origen, las reglas que se obtienen y los resultados intermedios (ver figura 4.2).

El *kernel,* a su vez, está compuesto por el *translator*, que se encarga del análisis sintáctico de una orden MINE RULE y genera un conjunto de programas SQL para ser utilizados posteriormente; el *preprocesador*, que a partir de los datos iniciales, genera un conjunto de tablas codificadas (a cada item candidato a aparecer en las reglas se le asocia un identificador único), que son almacenadas nuevamente en el SGBD; el *core operator*, compuesto por un conjunto de algoritmos para reglas de asociación, produce, a partir de las tablas codificadas, preparadas por el preprocesador, un conjunto de reglas codificadas, i.e. reglas de asociación cuyos elementos están codificados, que son almacenadas en forma de tablas en el SGBD; el *postprocesador*, decodifica las reglas y las entrega en una forma entendible para el usuario.

Según los autores, la implementación del operador MINE RULE no puede ser completamente relacional (i.e. que todo se exprese con operadores SQL) aún en el contexto de una arquitectura fuertemente acoplada. La parte del núcleo de Minería de Datos es responsabilidad de un procedimiento no-SQL, llamado *core operator* el cual recibe los datos recuperados por el SQL server, extrae las reglas y las retorna en forma de una relación SQL3 hacia el SGBD.

Muchas características del operador MINE RULE (como la evaluación de subconsultas, agrupamiento, etc.) son mejor ejecutados en SQL, por fuera del *core operator* [28]. Por esta razón, en esta arquitectura existe una frontera entre los procesos relacionales y de Minería de Datos, dependiendo si los procesos los puede realizar el SQL server o el *core operator*.

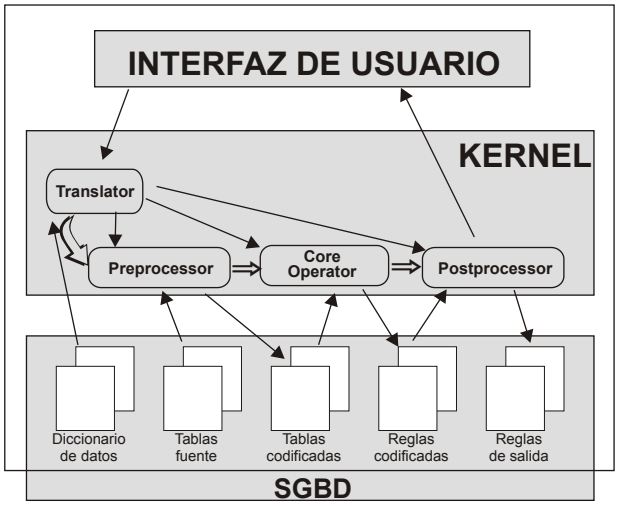


**Figura 4.2. Arquitectura de Minería de Datos que soporta al operador MINE RULE (28)**

Las ventajas de esta propuesta son: el poder expresivo del operador MINE RULE para formular cualquier tarea sobre reglas de asociación; la definición de una sintaxis del operador y la extensión del álgebra relacional para soportar formalmente al operador. La desventaja es la arquitectura en la cual se ha implementado el operador MINE RULE, que a pesar que los autores consideran fuertemente acoplada con un SGBD, no lo es en su totalidad, debido a que los algoritmos de Minería de Datos forman parte de un módulo independiente llamado *kernel* por encima del SGBD como lo muestra la figura 4.1 [28]. En este módulo se realiza todo el proceso de descubrimiento de reglas. El SGBD se utiliza únicamente para almacenar los datos iniciales, las reglas de salida, los resultados intermedios y para realizar ciertas tareas del operador MINE RULE (como evaluación de subconsultas, agrupamientos, etc.), que según los autores, es más eficiente hacerlo por fuera del *core operator*. Además, no se reporta nada acerca de la optimización de consultas realizadas con el operador MINE RULE ni tampoco sobre el rendimiento de esta arquitectura.

Clear et al. [9], [10] proponen e implementan un conjunto de primitivas para soportar el proceso de DCBD al interior del motor de NonStop

SQL/MX, un nuevo SGBD objeto-relacional, paralelo de la Division Tandem de Compaq. Estas primitivas tales como *transposition, vertical partitioning, round-robin, horizontal partitioning, sequence functions y sampling*, extienden el motor del SGBD y se expresan a través de una interfaz SQL, junto con otras características de alto rendimiento del motor SQL/MX, permiten realizar algunas tareas básicas de descubrimiento de conocimiento de manera eficiente y escalable.

La primitiva *transposition*, implementada como TRANSPOSE en la cláusula *select*, resume varias cláusulas *group by* en una, permitiendo obtener múltiples resultados, tales como contar la frecuencia por cada atributo y el cruce de tablas, en una sola búsqueda sobre una tabla [9].

*Vertical partitioning* permite particionar verticalmente una tabla por un atributo o grupo de atributos y almacenar el resultado en archivos separados con el fin de proveer un acceso rápido a únicamente los atributos que se necesitan. *Round-robin* permite expandir los datos equitativamente a través de múltiples particiones horizontales. *Horizontal partitioning* permite realizar particiones horizontales sin requerir un conocimiento detallado de los datos o de la definición de una llave de particionamiento. Según los autores, estas tres estrategias de particionamiento de tablas son muy útiles en la paralelización de una consulta.

SQL/MX soporta un número de funciones de secuencia (*sequence functions)* que se pueden usar, para especificar en forma concisa, consultas que involucren secuencias de datos en el tiempo.

La primitiva *sampling* permite generar rápidamente respuestas por muestreo. Ofrece tres tipos de muestreo: *randómico*, los *primeros n registros* y *periódico*. Por cada tipo, se puede especificar el tamaño del muestreo indicando el porcentaje de los datos para el muestreo o con un número determinado de registros.

Esta propuesta es sin duda la única que se puede considerar fuertemente acoplada al integrar al interior del motor NonStop SQL/MX un conjunto de primitivas que permite soportar, de manera eficiente y escalable, algunas tareas básicas de descubrimiento de conocimiento. El hecho de ser NonStop SQL/MX un motor paralelo hace que esta arquitectura sea muy específica, ya que otros motores no paralelos seguramente no pueden aprovechar estas ventajas.

Uno de los proyectos de investigación que Microsoft actualmente adelanta es el de Minería de Datos [31], cuyo objetivo es proveer a la industria del software de estándares para Minería de Datos. Entre las metas de este proyecto están: desarrollar métodos computacionales eficientes para la extracción de patrones, limpieza y reducción de datos en grandes bases de datos; construir un sistema escalable y eficiente que esté fuertemente integrado con un sistema de bases de datos relacional/OLAP, usando métodos de bases de datos, estadísticas, complejidad de algoritmos y optimización. Éste es un proyecto a largo plazo.

A corto plazo, el objetivo será automatizar el proceso de Minería de Datos sobre Bodegas de Datos e incluye entre otras áreas la integración de Minería de Datos con SGBD. En esta área, en colaboración con el Grupo de SQL Server, se han identificado las interfaces que permitan la integración de Minería de Datos con SQL Server. El resultado es la definición de la especificación OLE-DB para DM (OLE-DB para Minería de Datos), una extensión de OLE DB (un conjunto de interfaces para acceder a un diverso rango de tipos de datos, localizados en una variedad de medios de almacenamiento de datos) que introduce una interfaz común para Minería de Datos. OLE-DB para DM permitirá que los desarrolladores de bases de datos fácilmente accedan y exitosamente apliquen la tecnología de Minería de Datos en sus aplicaciones. El trabajo futuro será el de hacer posible la integración fuerte entre las consultas de las bases de datos y la Minería de Datos.

Dentro de las arquitecturas fuertemente acopladas, actualmente en la Universidad del Valle-Cali-Colombia, en el doctorado en ingeniería, área de énfasis Ciencias de la Computación se está desarrollando la propuesta de investigación doctoral denominada "Nuevas Primitivas SQL para el Descubrimiento de Conocimiento en Arquitecturas Fuertemente Acopladas con un SGBD" [37]. Esta propuesta se enmarca dentro del lenguaje de consultas de una arquitectura de descubrimiento de conocimiento fuertemente acoplada con un SGBD y tiene como propósito extender el motor relacional de POSTGRES [34] con nuevas primitivas de descubrimiento de conocimiento que permita soportar eficientemente el proceso DCBD.

La arquitectura propuesta ofrece enriquecer el lenguaje de consulta SQL, extendiéndolo con nuevos operadores de descubrimiento de conocimiento, con el fin de permitir la extracción de patrones de datos bajo restricciones de confianza y soporte.

La eficiencia del sistema se garantiza mediante la extensión del álgebra relacional con nuevos operadores especializados que permitan expresar consultas DCBD, mediante el robustecimiento del optimizador de consultas con nuevas estrategias de búsqueda y transformación, basados en esa nueva álgebra extendida. El nuevo optimizador debe generar eficientes planes de ejecución para las nuevas consultas que involucran descubrimiento de conocimiento.

Por otra parte, el sistema será compatible con el modelo relacional al determinar un espacio uniforme de representación de los datos de la base de datos y el conocimiento generado lo que permitirá el desarrollo de aplicaciones.

## 5. CONCLUSIONES

Actualmente debido a que la cantidad de información almacenada en las bases de datos existentes ha crecido y sigue creciendo de manera considerable, se ha hecho necesario desarrollar métodos y herramientas eficientes para extraer conocimiento de esas bases de datos. Sin embargo, la mayor parte de las herramientas DCBD propuestas se han desarrollado como una capa externa a los SGBD, mediante un acoplamiento débil, lo que conlleva a que sean ineficientes en términos computacionales. Más aún, no existen actualmente un SGBD relacional que permitan descubrir este conocimiento eficientemente bajo una arquitectura DCBD fuertemente acoplada.

Por otra parte, existen propuestas de investigación para extender el lenguaje SQL con nuevos operadores que permitan el descubrimiento de conocimiento en SGBD, pero sus implementaciones se han realizado en su mayor parte en arquitecturas débilmente acopladas. Además, no existe actualmente un consenso general sobre el conjunto de primitivas necesarias para soportar el descubrimiento de conocimiento en un SGBD.

La propuesta de Clear et al. [9], [10] es sin duda la única que se puede considerar fuertemente acoplada, pero el hecho de ser NonStop SQL/MX un motor paralelo, hace que esta arquitectura sea muy específica, ya que otros motores no paralelos seguramente no pueden aprovechar estas ventajas.

La integración de los Sistemas de Descubrimiento de Conocimiento y Bases de Datos, especialmente en acoplamientos fuertes, es una área activa de investigación que continuará en el futuro cercano.

## REFERENCIAS BIBLIOGRÁFICAS

1.Agrawal R., Srikant R., Fast Algorithms for Mining Association Rules, VLDB Conference, Santiago, Chile, 1994.

2.Agrawal R., Mannila H., Srikant R., Toivonen H., Verkamo A.I., Fast Discovery of Association

Rules, in Advances in Knowledge Discovery and Data Mining, AAAI Press / The MIT Press, 1996.

3.Agrawal R., Mehta M., Shafer J., Srikant R., Arning A., Bollinger T., The Quest Data Mining System, 2° Conference KDD y Data Mining, Portland, Oregon, 1996.

4.Agrawal R., Shim K., Developing Tightly-Coupled Data Mining Applications on a Relational Database System, The Second International Conference on Knowledge Discovery and Data Mining, Portland, Oregon, 1996.

5.Agrawal R., Shim K., Developing Tightly-Coupled Applications on IBM DB2/CS Relational Database System: Methodology and Experience, Research Report, 1996.

6.Brachman R., Anand T., The Process of Knowledge Discovery in Databases: A First Sketch, Workshop on Knowledge Discovery in Databases, 1994.

7.Chaudhuri S., Data Mining and Database Systems: Where is the Intersection?, Bulletin of the Technical Committee on Data Engineering, Vol. 21 No. 1, Marzo, 1998.

8.[ChHY96] Chen M., Han J., Yu P., Data Mining: An Overview from Database Perspective, IEEE Transactions on Knowledge and Data Engineering, 1996.

9.Clear, J., Dunn, D., Harvey, B., Heytens, M., Lohman, P., Mehta, A., Melton, M., Rohrberg, L., Savasere, A., Wehrmeister, R., Xu, M., NonStop SQL/MX Primitives for Knowledge Discovery, KDD-99, San Diego, USA, 1999.

10.Clear, J., Dunn, D., Harvey, B., Heytens, M., Lohman, P., Mehta, A., Melton, M., Rohrberg, L., Savasere, A., Wehrmeister, R., Xu, M., Large Scale Knowledge Discovery Using NonStop SQL/MX, Technical Report, Compaq Computer Corporation, Tandem Division, 1999.

11.Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., From Data Mining to Knowledge Discovery: An Overview, in Advances in Knowledge Discovery and Data Mining, AAAI Pres/ The MIT Press, 1996.

12.Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., The KDD Process for Extracting Useful Knowledge from Volumes of Data, Comunications of the ACM, Vol 39, No. 11, November, 1996.

13.Fayyad U., Mining Databases: Towards Algorithms for Knowledge Discovery, Bulletin of the IEEE Computer Society, Vol.21, No. 1, March, 1998.

14.Goebel, M., Gruenwald, L., A Survey of Data Mining and Knowledge Discovery Software Tools, SIGKDD Explorations, Vol. 1, Issue 1, June, 1999.

15.Han J., Fu Y., Wang W., Chiang J., Zaiane O., Koperski K., DBMiner: Interactive Mining of Multiple-Level Knowledge in Relational Databases, ACM SIGMOD, Montreal, Canada, 1996.

16.Han J., Fu Y., Wang W., Chiang J., Koperski K., Li D., Lu Y., Rajan A., Stefanovic N., Xia B., Zaiane O., DBMiner: A System for Mining Knowledge in Large Relational Databases, The second International Conference on Knowledge Discovery & Data Mining, Portland, Oregon, 1996.

17.Han J., Fu Y., Wang W., Koperski K., Zaiane O., DMQL: A Data Mining Query Language for Relational Databases, SIGMOD 96 Workshop, On research issues on Data Mining and Knowledge Discovery DMKD 96, Montreal, Canada, 1996.

18.Han J., Chiang J., Chee S., Chen J., Chen q., Cheng S., Gong W., Kamber M., Koperski K., Liu G., Lu Y., Stefanovic N., Winstone L., Xia B., Zaiane O., Zhang S., Zhu H., DBMiner: A System for Data Mining in Relational Databases and Data Warehouses, CASCON: Meeting of Minds, Toronto, Canada, 1997.

19.Han J., Fu Y., Tang S., Advances of the DBLearn System for Knowledge Discovery in Large Databases, Int'l Joint Conference on Artificial Intelligence IJCAI, Montreal, Canada, 1995.

20.IBM Corporation, Intelligent Miner, http://www-4.ibm.com/software/data/iminer, 2001.

21.Imielnski T., Mannila, H., A Database Perspective on Knowledge Discovery, Communications of the ACM, Vol 39, No. 11, November, 1996.

22.Imielinski T., Virmani A., Abdulghani A., Data Mine: Application Programming Interface and Query Language for database Mining, 2° Conference KDD y Data Mining, Portland, Oregon, 1996.

23.Isoft S.A. Alice, http://www.alice-soft.com/html/prod_alice.htm, 2001.

24.Kdnuggets, http://www.kdnuggets.com/software, 2001.

25.Matheus C., Chang P., Piatetsky-Shapiro G., Systems for Knowledge Discovery in Databases, IEEE Transactions on Knowledge and Data Engineering, Vol 5, No 6, 1993.

26.Meo R., Psaila G., Ceri S., A New SQL-like Operator for Mining Association Rules, VLDB Conference, Bombay, India, 1996.

27.Meo R., Psaila G., Ceri S., An Extension to SQL for Mining Association Rules, Data Mining and Knowledge Discovery, Kluwer Academic Publishers, Vol 2, pp .195-224, Boston, 1998.

28.Meo R., Psaila G., Ceri S., A Tightly-Coupled Architecture for Data Mining, 14th. International Conference on Data Engineering ICDE98, 1998.

29.Piatetsky-Shapiro G., Brachman R., Khabaza T., An Overview of Issues in Developing Industrial Data Mining and Knoledge Discovery Applications, 2° Conference KDD y Data Mining, Portland, Oregon, 1996.

30.Quadrillion Corp., Q-Yield, http://www.quadrillion.com/qyield.shtm, 2001.

31.Research Microsoft, http://research.microsoft.com/dmx/datamining, 2001.

32.RuleQuest Research Inc., C5.0, http://www.rulequest.com, 2001.

33.SPSS, Clementine, http://www.spss.com/clementine, 2001.

34.Stonebraker, M., Rowe, L., A., The Design of POSTGRES, Proceedings of the ACM-SIGMOD Conference, Washington D.C., 1986.

35.Sarawagi, S., Thomas S., Agrawal R., Integrating Association Rule Mining with Relational Database Systems: Alternatives and Implications, ACM SIGMOD, 1998.

36.Sarawagi, S., Thomas S., Agrawal R., Integrating Association Rule Mining with Relational Database Systems: Alternatives and Implications, Data Mining and Knowledge Discovery, Kluwer Academic Publishers, Vol 4, 2000.

37.Timarán, R., Nuevas primitivas SQL para el Descubrimiento de Conocimiento en Arquitecturas Fuertemente Acopladas con un SGBD, propuesta investigación doctoral, Doctorado en Ingeniería, Universidad del Valle, Cali, Colombia, 2001, ritimar@borabora.univalle.edu.co.