Aplicación para la Encuesta de Opinión Estudiantil Utilizando Consultas Difusas.

Introduccion .

==========

La Coordinación encargada de la elaboración y aplicación de la encuesta de opinión estudiantil en la Universidad Simón Bolívar (EOE-USB) decidió digitalizar los resultados de la misma unos años atrás, generando una base de datos, con los resultados historicos de la misma.

Dicha Base de Datos, representa una fuente de información estadisticamente muy valiosa para entender la perspectiva del estudiante acerca de las catedras de la universidad, y de los profesores que la imparten.

El tipo de información alojada en esta base de datos, se presenta como ideal para ser procesada mediante consultas de logica difusa, es decir, bajo preguntas en lenguaje natural, que son extrapoladas a consultas sobre una base de datos relacional; bajo esta idea y aprovechando la existencia de herramientas ideadas para generar consultas en logica difusa; se implementó un sistema experimental, que permite demostrar las virtudes de las consultas con lógica difusa en una aplicación de utilidad real.

Implementacion:

---------- ----------

Para la implementación de este proyecto, se utilizaron 2 motores para la elaboración de consultas difusas; SQLFI, un conjunto de librarias en JAVA que establecen una capa intermedia entre la aplicación y la Bd y Postgresqlf, una modificación del DBMS PostgreSQL que permite la elaboración de consultas difusas de manera nativa.

Para la implementación del sistema que aprovechará estos motores, se utilizo PLAY Framework, una herramienta orientada al desarrollo agil de aplicaciones web sobre el modelo MVC basada en el lenguaje de programación JAVA, tal herramienta se eligió debido a la necesidad de utilizar JAVA para poder utilizar de la manera mas eficiente las librerias de SQLFI.

Los mayores retos para la implementación de este sistema, provinieron del hecho de que ambos motores para aplicaciones difusas se encuentran en una fase experimental, por lo que aun poseen varias limitaciones, defectos y una documentación escaza.

Afortunadamente dichos problemas pudieron ser sorteados en base a una bateria de pruebas que permitió hallar los puntos debiles de cada motor para plantear soluciones ajustadas a tales fallas; a continuación se presentan los problemas encontrados en cada manejador.

POSTGRESQLF:

- La Utilizacion de predicados difusos sobre consultas complejas generaba salidas correctas pero con grados de pertenencia al conjunto errados.

Dicho problema fue solucionado mediante la creación de vistas materializadas (tablas temporales) donde se alojaba la complejidad de las consultas, de manera tal que la nueva consulta era mucho más sencilla, bajo tal esquema las consultas arrojaban la salida esperada.

- No se pueden utilizar numeros reales en la especificacion de los predicados difusos

Esta limitación decanta en reducir la calidad de las consultas al reducir su precision, una posible solución a esto es utilizar un rango entero de mayor magnitud al deseado y hacer los mapeos necesarios.

- En caso de utilizar tipos indebidos dentro de la elaboracion de una consulta con terminos difusos, el servidor de bases de datos aborta de maneran inesperada sin dejar ninguna explicación de porque ocurrio tal evento.

Para este caso, se hicieron varias pruebas sobre el manejador, hasta poder deducir cual era el origen del problema, posteriormente se procedio a probar las mismas consultas que daban problemas con los tipos correctos.

SQLFI:

- La Versión que poseaimos del motor no era la correcta:

Despues de seguir al pie de la letra, las instrucciones proveidas en la documentacion de este motor, revisando las salidas de error, y el codigo fuente del mismo, pudimos notar que el sistema buscaba objetos no disponibles, por lo que asumimos no poseer la versión final; despues de ponernos en contacto con el desarrollador del sistema, pudimos obtener la versión correcta y utilizar la herramienta.

- Las consultas complejas genera llamadas a consultar la tabla 'user_tab_columns' que no existe en postgresql, sin embargo la salida de las consultas es correcta.

Despues de notar este error, se hicieron varias pruebas sobre el sistema que demostraron que la salida era correcta, al indagar sobre este bug con el desarrollador del motor, nos comento que esto está relacionado con que postgresql no sigue todos los estandares del lenguaje SQL, y dado a que el sistema estaba implementado siguiendo dichos estandares.a veces se daban ciertos problemas de este tipo que aun no habian sido corregidos.

-Algunas llamadas a funciones no son reconocidas por las librerias del motor

Este es el caso de las llamadas a funciones almacenadas en la base de datos, tal problema se solucionó utilizando vistas que hicieran las llamadas de manera interna en el manejador; otra alternativa es hacer conexiones paralelas al servidor de base de datos para ejecutar tales consultas.

DIFERENCIAS ENTRE LOS MANEJADORES:

A nivel de funcionalidades

Postgresqlf: Permite realizar las siguientes operaciones :

CREACION DE PREDICADOS DIFUSOS POR EXTENSION Y EXPRESION
CREACION DE MODIFICADORES DIFUSOS
CREACION DE CONECTORES DIFUSOS
CREACION DE CUANTIFICADORES DIFUSOS
SELECT, UPDATE, CHECK, VISTAS, EXCEPT, INTERSECT, UNION CON CONDICIONES DIFUSAS

SQLFI: Permite realizar las siguientes operaciones:

CREACION, CONSULTA Y ELIMINACION DE PREDICADOS DIFUSOS POR CONDICION, EXPRESION, EXTENSION
CREACION, CONSULTA Y ELIMINACION DE MODIFICADORES DIFUSOS
CREACION, CONSULTA Y ELIMINACION DE CONECTORES DIFUSOS
CREACION, CONSULTA Y ELIMINACION DE COMPARADORES DIFUSOS
CREACION, CONSULTA Y ELIMINACION DE CUANTIFICADORES DIFUSOS
SELECT, UPDATE, CHECK, ASERCIONES, TRIGGERS, VISTAS, VISTAS IDB, PROCEDURES, FUNCIONES, PAQUETES CON CONDICIONES DIFUSAS

A Nivel de versatilidad:

Postgresqlf: A pesar de tener menos funcionalidades, postgresql se presenta mas comodo para realizar pruebas y crear y configurar las bases de datos de maneras mas versatiles, pues se cuenta con todas las herramientas de postgres, por ejemplo su cliente de conexiones; adicionalmente el hecho de que la herramienta se encuentre en el manejador, permite el uso de cualquier lenguaje de programacion para el desarrollo del sistema, con lo que se le da mayor libertad al desarrollador para el uso de herramientas y metodologias; sin embargo la estructura del esquema de base de datos es bastante rigida y no permite hacer modificaciones sobre el mismo.

SQLFI: la ventaja de utilizar sqlfi reside en el hecho de poder utilizar cualquier manejador de bases de datos, y en que el codigo escrito para tal fin funcionara en cualquiera de ellos ya que las librerias se encargan de hacer los mapeos necesarios; adicionalmente el esquema de la base de datos de sqlfi se presenta mas completo y flexible, por lo que se puede manipular la herramienta para ampliar sus funcionalidades sin mayor problema.

Conclusiones:

Como resultado al proceso de implementacion de esta herramienta, podemos resaltar en primer lugar, las virtudes y potencialidades de aplicar la logica difusa a problemas del mundo real.

En segundo lugar, se pudo probar a profundidas, las capacidades y debidilidades de los dos motores proveidos para generar las consultas, como se pudo comprobrar, ambos, estan capacitados para ser implementados en aplicaciones del mundo real, pero todavia quedan varios aspectos por pulir antes de

que sean herramientas totalmente confiables.