Herramienta informática para la identificación y reparación de inconsistencias de estructura y datos entre bases de datos PostgreSQL.

Ing. Eddy Figueredo Aguilar.
Ing. Odisleysi Martinez Furones.

Universidad de las Ciencias Informáticas. Facultad Regional Granma. Ave Camilo Cienfuegos,Granma, Cuba.
Autor para la correspondencia: efigueredo@grm.uci.cu .

Resumen

El manejo de los datos mundialmente se ha hecho eficiente con el surgimiento de las bases de datos (BD), dando paso a la necesidad de crear una forma para garantizar la seguridad de estos datos, de ahí surge el respaldo a la información almacenada. La redundancia de la información almacenada en las BD, es generalmente la principal causa de la ocurrencia de inconsistencias, incluyendo además los procesos de actualización o cualquier cambio que se haga en el estado de las mismas que no sea llevada a su respaldo. Mediante el presente trabajo se realiza un análisis del proceso de actualización en las BD y entre BD y su respaldo, con el objetivo de entender las causas de las apariciones de las inconsistencias y modelar una herramienta informática capaz de tratarlas, para luego implementarla. En la primera parte del mismo se realiza un estudio del estado del arte seleccionándose la estrategia para llevar a cabo la tarea. Se establece además una metodología para guiar el proceso de desarrollo del software, que incluye el desarrollo de los artefactos necesarios para implementar y probar la herramienta informática deseada.

Palabras claves:
Herramienta informática, inconsistencias, proceso de actualización, redundancia, respaldo.

Summary

The data management world has been efficient with the emergence of the databases (DB), giving way to the need to create a way to ensure the safety of these data, which raises backup for stored information. The redundancy of the information stored on the DB, is usually the main cause of the occurrence of inconsistencies, further comprising updating processes or any change made in the same state is not brought to a backup. Through this work, an analysis of the upgrade process in DB and between DB and backup, in order to understand the causes of the appearances of the inconsistencies and modeling tool capable of treating them, and then implement it. In the first part of it is a study of the state of the art selected strategy to perform the task. It also provides a methodology to guide the software development process, including the development of the artifacts necessary to implement and test the software tool desired.

Keywords:
Software tool, inconsistencies, updating process, redundancy, backup.

Introducción.

En la actualidad el mundo se maneja a través de la información, la cual está creciendo considerablemente en las instituciones y empresas. Un estudio de la Corporación Egan Marino C. (EMC²) -corporación, fabricante mundial en soluciones de infraestructura de la información-, asegura que en el 2010 la cantidad de información generada mundialmente estuvo cerca de 1,2 zettabytes. Un zettabyte equivale a 1 trillón gigabytes. En 2009, y en medio de la gran recesión económica, la cantidad de información digital creció en un 62 %, en comparación con el 2008, donde se lograron 800 billones gigabytes o 0,8 zettabytes, con una tasa de crecimiento anual del 60 %, el universo digital está creciendo rápidamente y se estima que la información digital alcanzará los 1.8 zettabytes para el año 2011 (ANON. 2010 a)(EMC Corporation 2008). Para contribuir a la organización y manipulación eficiente de la información se crearon los sistemas gestores de bases de datos (en lo adelante SGBD), los cuales influyen en el rendimiento, en el buen uso de las BD, en su correcto funcionamiento, y en su seguridad, ya que implican un mayor control sobre las mismas (Sumathi, Esakkirajan 2007). Sin embargo algunos de los inconvenientes o desventajas de los SGBD están dados por su tamaño, pues los SGBD son programas que requieren una gran cantidad de espacio en disco y de memoria para trabajar de forma eficiente. Otro de los inconvenientes es el costo económico que tienen, ya que varían en dependencia de la funcionalidad que ofrezca, del entorno y el mantenimiento que suele ser un porcentaje del precio del SGBD anualmente. Las prestaciones también constituyen otra de las desventajas debido a que los SGBD están desarrollados para ser útiles en muchas aplicaciones. Esto trae consigo que algunas de ellas no sean tan rápidas como en los sistemas de archivos. Los SGBD también son vulnerables a los fallos, como todos los datos están centralizados el sistema se hace más vulnerable ante los fallos que se produzcan. Otra de las afectaciones a las BD puede ocurrir debido a que los archivos que mantienen almacenada la información son creados por diferentes tipos de programas de aplicación y existe la posibilidad de que si no se controla detalladamente el almacenamiento, se pueda originar un duplicado de información. Esto aumenta los costos de almacenamiento y acceso a los datos, además de que origina la inconsistencia de los datos, la cual ocurre cuando existe información contradictoria o incongruente en la BD, es decir diversas copias de un mismo dato no concuerdan entre sí, por ejemplo: se actualiza la dirección de un cliente en un archivo y que en otros archivos permanezca la anterior (Martínez 2002). Se pueden identificar distintos tipos de inconsistencias, dentro de las que se encuentra: de datos, estructurales, permisos, funciones y disparadores. Encontrar estas diferencias es muy engorroso cuando se trata de una gran cantidad de información por lo que sería necesaria una forma de facilitar la búsqueda de las mismas. Las BD también pueden presentar afectaciones referentes a la pérdida de la información. Según la Máquina Internacional de Negocio (IBM) el 17,5 % de la pérdida de la información se debe a

sabotajes o hechos terroristas, otro 17,5 a accidentes, un 14,0 a eventos atmosféricos, el 7,0 a inundaciones, los terremotos representan un 10,5 % mientras que se reporta un 8,8 por errores de software, 9,5 por caídas eléctricas o fallos eléctricos, otros factores definidos fueron por fallos de red y roturas de conducción con 3,5 % respectivamente, en tanto, por errores de hardware se identificó un 5,3 % y un 2,8 para otras eventualidades no definidas dentro de las mencionadas (Espinosa 2009). Como se ha podido observar, las afectaciones que presenta una BD son bastante diversas, y en su conjunto hacen que aumente la probabilidad de que la misma presente fallos en algún momento. Debido a esto las empresas u organismos emplean generalmente las copias de respaldo para evitar la pérdida total o parcial de la información. Generalmente el contenido a salvaguardar se define de acuerdo con su importancia. Es importante tener en cuenta que estas copias de respaldo deben tener la garantía de poder recuperarlas una vez que ocurra un fenómeno determinado. Dentro de los mecanismos o tipos de copias en función de la cantidad de información con la que se trabaje se encuentran: la copia de seguridad total o íntegra, la copia de seguridad incremental, y la copia de seguridad diferencial (ANON. 2010 b). Sin embargo, estos mecanismos, no garantizan la seguridad y disponibilidad total de la información pues puede darse el caso de existir diferencias en la copia de respaldo. Para identificar estas diferencias se han creado herramientas entre las que se destacan EMS DB Comparer for PostgreSQL (EMS Software Development 2011) y PostgreSQL Data Sync (SQL Maestro Group 2012 b). Estas herramientas están hechas para Windows, comparan y sincronizan el contenido de la BD y no resuelven todos los tipos de diferencias. PostgreSQL es uno de los SGBD más conocidos y está basado en POSTGRES: sucesor del IGRES SGBD desarrollado durante el período comprendido entre los años 1975 a 1977 (Stonebraker, Rowe 1986), Versión 4.2, desarrollada en la Universidad de California Berkeley en el Departamento de Informática. POSTGRES abrió camino a muchos conceptos que solo se pusieron disponibles tiempo después en algunos sistemas de BD comerciales. PostgreSQL es un descendiente del código-abierto del código original de Berkeley. Apoya en gran parte al estándar SQL y ofrece muchos rasgos modernos. Además, el usuario puede crear nuevos tipos de datos, funciones, así como operadores, entre otros. Debido a la licencia libre, PostgreSQL puede usarse, puede modificarse, y puede distribuirse por todos de manera gratis para cualquier propósito, sea privado, comercial, o académico (The PostgreSQL Global Development Group 1996 a). Debido a sus prestaciones, a sus características y a su licencia, hoy en día es uno de los SGBD más utilizados mundialmente. En la actualidad la versión más utilizada en la facultad 3 de la Universidad de las Ciencias Informáticas (UCI) es la 8.4 debido a que fue liberada en el 2009 con más de 250 mejoras respecto a sus anteriores versiones, dentro de las que se encuentra: la restauración de BD en procesos paralelos acelerando la recuperación de un respaldo hasta 8 veces, los privilegios por columna, la configuración de ordenamiento configurable por BD, las nuevas herramientas de monitoreo de consultas que le otorgan a los

administradores mayor información sobre la actividad del sistema (León, Rodríguez, Veitia 2011). Por lo antes expuesto se enfrenta la siguiente situación problemática: ha aumentado el volumen de la información almacenada, así como la complejidad de los mecanismos utilizados para su almacenamiento, los SGBD y las herramientas de administración existentes no garantizan la integridad de la información, todos los SGBD están propensos a la ocurrencia de fallas que provoquen la pérdida total o parcial de la información y no existe una herramienta informática libre que sea capaz de identificar y reparar las inconsistencias que se presentan entre BD PostgreSQL.

Materiales y Métodos.

Con el objetivo de elaborar un marco teórico para lograr la investigación se emplearon los siguientes métodos:

Método teórico análisis histórico-lógico: Para realizar la recopilación de la información sobre las BD PostgreSQL, su trayectoria y comportamiento con respecto al fenómeno de la inconsistencia de datos, además de la investigación de las leyes generales del funcionamiento y desarrollo de este fenómeno. Método teórico modelación: Para realizar el diseño de una propuesta de solución a la problemática planteada.

Método teórico Análisis-Sintético: Para analizar las teorías, documentos, estudiar la bibliografía referente al tema, etc.

Método empírico entrevista: Para identificar la Problemática y el Problema a resolver se hizo necesario utilizar este método haciéndoles encuestas a algunos arquitectos de BD de distintos proyectos de la UCI, dentro de los que se encuentra el Ing. Juniel Tamayo Hernández, arquitecto de BD del Proyecto División de Antecedentes Penales de la República Bolivariana de Venezuela (DAP) y el Ing. Vlamir Rodríguez Fernández administrador de BD del proyecto Fiscalía fase 1.

Bases conceptuales

Definir los conceptos principales ayuda al entendimiento y fundamenta las bases para la realización de la indagación, a continuación se desglosan los conceptos más importantes para la investigación:

SGBD

Los SGBD consisten en la recopilación de datos interrelacionados y un conjunto de programas para acceder a los datos. Son software que ayudan a mantener y utilizar una BD (Sumathi, Esakkirajan 2007).

PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional (SGBDOR), distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el SGBD de código abierto más potente del mercado y en sus últimas versiones no tiene nada que anhelar a otras BD comerciales. (The PostgreSQL Global Development Group 1996 b)

Inconsistencias de BD PostgreSQL

Las BD tienen como principal objetivo almacenar la información. Teniendo en cuenta el incremento diario de esta, es necesario lograr la creación de BD con un diseño óptimo para evitar los problemas que trae consigo la redundancia de la información, así es como se eliminan los datos repetidos o los problemas al realizar cualquier transformación en la misma. Generalmente cualquier anomalía con la BD surge a partir de una operación realizada. Las causas de la ocurrencia de la inconsistencia de datos en un sistema pueden ser accidentales o intencionales (con fines indebidos). Dentro de las causas intencionales, se tiene el robo de la información o la lectura sin autorización y la modificación o la destrucción no autorizada. En las causas accidentales están: las fallas de conexión durante el procesamiento de transacciones, fallas del hardware debido al medioambiente o catástrofes, anomalías por acceso concurrente, anomalías que resultan de la distribución de los datos entre varios servidores y un error lógico que viole la suposición de que las transacciones respetan las restrictivas de consistencia de los datos o un error cometido por algún usuario al introducir datos (Rose 1993).

Las anomalías en BD son ciertos problemas que aparecen con frecuencia en el manejo de las mismas cuando el diseño no ha sido realizado de forma normalizada, o sea, no se han diseñado teniendo en cuenta un conjunto de reglas para evitar problemas de lógica, estas reglas son conocidas como formas normales. Las anomalías básicas que surgen a partir de transformaciones que se hagan en las BD se definen como:

- **Anomalía de inserción**: Imposibilidad de dar de alta una tupla por no disponer del valor de un atributo principal.

- **Anomalía de borrado**: Pérdida de información por dar de baja una tupla.

- **Anomalía de modificación**: Tiene que ver con la redundancia. En general, la normalización reduce la redundancia, pero no la elimina por completo.

La inconsistencia ocurre cuando existe información contradictoria o incongruente en la BD, es decir diversas copias de un mismo dato no concuerdan entre sí. Esto ocurre debido a la existencia de redundancia en la BD, si por algún motivo la actualización de alguno de los datos redundantes fallara se estaría en presencia de inconsistencia. Los tipos de inconsistencias que se identificaron en las BD son los siguientes:

Inconsistencia de datos:

Los SGBD garantizan la integridad de la información, teniendo en cuenta la no existencia de redundancia, ya que esta significa en muchos casos la posibilidad de inconsistencia en la información, pues la inconsistencia de datos se identifica a partir de transformaciones que se hagan en las BD, surgiendo a partir de las anomalías de actualización, es decir, cuando un dato redundante es actualizado en una parte, y no coincide en otra.

Inconsistencia de estructura:

Las inconsistencias de estructura, como su nombre lo indica, tienen mucho que ver con la estructura de la BD y por lo tanto se definen en el diseño de la misma. Las inconsistencias de estructura son las diferencias estructurales que se dan en el diseño o las relaciones entre objetos de las BD, entre una BD origen y su respaldo, es decir, debido a cualquier problema o anomalía de actualización al cambiarse el nombre de una tabla o la relación entre ellas en una BD origen, no se hace en su respaldo quedando esta última inconsistente con respecto a su origen.

Inconsistencias de programación:

Se definen como las inconsistencias que surgen al realizar cambios en las consultas para el trabajo con las BD, al no realizarse los cambios en la copia de respaldo de la misma. Es decir, las diferencias estructurales surgidas en las consultas que afectan la respuesta o el funcionamiento de estas.

Resultados y Discusión.

Para identificar y corregir las inconsistencias de bases datos mundialmente existen herramientas, algunas de las más relevantes se comparan a continuación y se tomó como criterio de revisión sus tipos de licencias, los SGBD, sistema Operativo (SO), e inconsistencias que trabajan.

Herramientas	*Licencia*	*SGBD*	*SO*	*Inconsistencia que trabaja*
EMS DB Comparer for PostgreSQL 3.3	propietario	PostgreSQL	Windows	contenido, estructura
dbForge Schema Compare for SQL Server 1.10	propietario	Oracle, MySQL	Windows	contenido, estructura
D-Softs Database Comparer 2.0	propietario	MSSQL	Windows	esquemas, contenido
PostgreSQL Data Sync	propietario	PostgreSQL	Windows	contenido
DreamCoder for PostgreSQL	libre	PostgreSQL	Windows	contenido, estructura
Nueva herramienta	libre	PostgreSQL	Windows, Linux	contenido, estructura, programación

Tabla 1. Comparación entre las herramientas para la detección y corrección de inconsistencias de BD

El estudio realizado arrojó la existencia de varias herramientas que realizan la comparación y sincronización de BD, destacando solo las mencionadas, para el SGBD PostgreSQL. Mostró

además que existen herramientas para trabajar con el gestor que incluyen entre sus funcionalidades la sincronización, tal es el caso de DreamCoder for PostgreSQL. Además, se estudiaron otras herramientas que no trabajan con PostgreSQL con el objetivo de analizar las funcionalidades necesarias y sacar definiciones sobre las inconsistencias a tener en cuenta. Se concluye entonces que la vía factible para resolver el problema es desarrollar una nueva herramienta, ya que ninguna de las herramientas expuestas resuelve todas las inconsistencias antes analizadas, además de que la mayoría son propietarias y corren sobre el SO Windows. La herramienta que se desarrollará es para la versión 8.4 de PostgreSQL aunque se pretende probar y extender a PostgreSQL 9.x.

Funciones de similitud sobre cadenas de texto.

Como bien se ha tratado las BD revolucionaron el mundo moderno, ya que almacenan grandes volúmenes de información, sin embargo, estas pueden presentar información duplicada que en ocasiones llevan a una decisión equivocada. El proceso que detecta este conflicto se conoce en el ámbito de las BD como: merge-purge, data deduplication o instance identification, así como detección de duplicados.

En 1946 el primero en plantear el proceso de detección de duplicados fue Halbert L. Dunn, al transcurrir de los años se fue perfeccionando, en el año 2000, Winkler propuso mejoras al modelo. En su forma más simple, dicho proceso es como sigue:

Dado un conjunto R de registros.

1) Se define un umbral real ø ∈ [0,1].

2) Se compara cada registro de R con el resto.

3) Si la similitud entre una pareja de registros es mayor o igual que ø, se asumen duplicados; es decir, se consideran representaciones de una misma entidad real.

Persiguiendo que la función tenga efectividad es necesaria alguna función de similitud que, dados dos registros con la misma estructura, devuelva un número real en el intervalo [0,1] igual a uno si ambos registros son idénticos y menor cuanto más diferentes sean. Tal valor depende de la similitud entre cada pareja de atributos respectivos. Luego, es necesaria otra función de similitud al nivel de atributo (no de registro), siendo frecuente en este contexto tratar los atributos, sin importar su tipo de datos, como cadenas de texto (strings). Estas funciones se clasifican en dos categorías: basadas en caracteres y basadas en palabras completas o tokens. Se empleó una variante de las funciones de similitud basadas en palabras completas o tokens, en forma de algoritmo. Las funciones de similitud basadas en tokens consideran cada cadena como un conjunto de subcadenas separadas por caracteres especiales, como por ejemplo espacios en

blanco, puntos y comas, esto es, como un conjunto de tokens, y calculan la similitud entre cada pareja de tokens mediante alguna función de similitud basada en caracteres.

Para darle solución al problema tratado se emplea un híbrido de lo estudiado, teniendo en cuenta que se buscan inconsistencias, o sea, elementos que deberían ser los mismos y no lo son. Por tanto, se trabaja con elementos para la búsqueda de duplicados pero se analizan los resultados de forma inversa. Para encontrar una inconsistencia y mostrar la distancia entre cada token debe estar en su umbral mayor.

Con el objetivo de lograr el desarrollo de la herramienta se analizaron puntos para la implementación en los cuales se incluye la metodología seleccionada, la arquitectura, el lenguaje de programación, el ambiente de desarrollo, así como las pruebas a realizar a la misma para verificar el cumplimiento de las funciones deseadas. Las metodologías tradicionales son usadas por grupos grandes de trabajo, su uso generalmente implica un plan de trabajo más amplio. Teniendo en cuenta que se trata de un equipo de trabajo pequeño y que además se cuenta con poco tiempo para la implementación, luego del análisis realizado, se decidió utilizar una metodología ágil. A partir del estudio realizado sobre las metodologías ágiles se decidió utilizar AUP pues esta cuenta con las características necesarias para guiar el desarrollo de la herramienta propiciando el cumplimiento de su objetivo, posee características que tributan a las necesidades del desarrollo del software, es decir entregables suficientes para hacer entendible el sistema para posteriores implementaciones sobre el mismo. La arquitectura seleccionada fue la arquitectura en capas pues dentro de su funcionamiento contempla la reutilización de componentes, siendo este concepto utilizado en la implementación. También permite el mantenimiento y las mejoras a la solución, debido al bajo acoplamiento entre las capas, así como la alta cohesión de las mismas y la habilidad de cambiar su implementación sin cambiar las interfaces. Tiene en cuenta otras soluciones, ya que se reutilizan las funcionalidades expuestas por las diferentes capas, especialmente si las capas de las interfaces son diseñadas con la reutilización en mente. Además, la capacidad de realizar pruebas se beneficia de tener interfaces bien definidas para cada capa, así como de la habilidad para cambiar a diferentes implementaciones de las interfaces (Trowbridge, Mancini, Quick, Hohpe, Newkirk, Lavigne 2003). Para el desarrollo de la aplicación se decide utilizar JDK 1.7, teniendo en cuenta que la herramienta a desarrollar será libre, para su posterior uso se utilizara en las máquinas cliente OpenJDK 7 la versión libre del kit de desarrollo Java. Como IDE para la programación se seleccionó NetBeans 7.1 pues cuenta con bibliotecas gráficas muy potentes para el trabajo con interfaces gráficas de usuario, posee además muy buenos mecanismos de auto completamiento y corrección que facilita la implementación permitiendo que la misma se haga de forma rápida y eficiente. Es además una herramienta libre y sin restricciones de uso.

Propuesta de Solución.

La arquitectura está definida en 2 capas, estas son:

Presentación:

La capa de presentación, es la encargada de interactuar con el usuario de la aplicación mediante una interfaz de usuario. La interfaz de usuario estará contenida por varias vistas, por las cuales el usuario va a ir realizando las operaciones mediante pasos ordenados para arribar a la reparación de la base de datos que contenga inconsistencias.

Lógica de negocio:

La capa lógica de negocio es la responsable de realizar las tareas para las cuales se diseña el sistema. También es la encargada de gestionar el almacenamiento de los datos, mediante el SGBD PostgreSQL.

Teniendo en cuenta el análisis de los algoritmos expuestos, se estableció una guía para la implementación que establece los pasos necesarios para realizar la comparación deseada en la implementación. Se conformó el código a partir de la serie de pasos descritos a continuación:

1. Tratar los objetos seleccionados de las bases de datos como conjuntos independientes.

2. Definir la existencia de un conjunto menor o la igualdad de los mismos.

3. Comparar cada conjunto buscando la ocurrencia del menor en el mayor, definiendo igualdades y diferencias.

Es importante aclarar que este procedimiento se realiza por cada elemento una vez comprobado que son iguales en cuanto al nombre y el tipo, estableciendo una forma de comparación recursiva hacia abajo teniendo en cuenta que cada elemento puede contener un subconjunto.

Aporte y Novedad.

Luego del estudio realizado de las características del proceso de resolución de inconsistencias de estructura y de datos, del objeto de investigación y de las tendencias actuales de desarrollo de software, díganse metodologías de desarrollo, lenguajes de modelado, lenguajes de programación, es importante destacar que existen múltiples herramientas para la detección y corrección de las inconsistencias entre BD. Estas herramientas son en su mayoría propietarias y además no solucionan todas las inconsistencias surgidas, al no contemplar algunas funcionalidades, como la comparación de datos. Teniendo en cuenta estos aspectos, y la variedad de inconsistencias que ocurren entre BD PostgreSQL, se decide implementar una

herramienta que agrupe las funcionalidades necesarias para resolver las inconsistencias analizadas. La herramienta a desarrollar "Herramienta Informática para la Solución de Inconsistencias" (HIPSI), tiene como principal objetivo solucionar las inconsistencias analizadas (de datos y de estructura) que surgen a partir de las anomalías y su desarrollo será bajo licencia libre. Teniendo como novedad la forma de tratar las inconsistencia, al analizar las funciones de similitud entre cadenas de texto.

Conclusiones y Recomendaciones.

En el presente trabajo se propuso una herramienta para la identificación y reparación de inconsistencias de datos y de estructuras entre BD PostgreSQL, enfocada en el proceso de actualización de las BD con el objetivo de contribuir a su mantenimiento y potenciando la integridad de sus datos. Se cumplieron los objetivos propuestos y se arribó a las siguientes conclusiones:

- Se realizó un estudio del estado del arte del proceso de actualización de BD incluidos los algoritmos empleados para ello y las herramientas informáticas desarrolladas para este fin, logrando identificar las principales tendencias y concluyendo que no existe ninguna herramienta que solucione el problema identificado en esta investigación.

- Se aplicaron la metodología y arquitectura definidas generando los entregables previstos para cada fase del proceso de desarrollo, lo que permitió modelar la solución.

- Se utilizaron estándares de programación y patrones de diseño logrando la implementación de una herramienta informática para solucionar inconsistencias en BD PostgreSQL.

- La solución informática desarrollada fue sometida a un proceso de validación a través de la aplicación de métricas al diseño así como de pruebas de caja blanca y caja negra al código y a las interfaces del sistema respectivamente, finalmente, se realizaron pruebas de integridad a las BD comparadas para validar la ausencia de inconsistencias. En todos los casos, los resultados finales fueron satisfactorios.

Para posteriores versiones de la herramienta es necesario se tomen en cuenta, en función de un mejor desempeño del personal en las próximas versiones del proyecto, las siguientes recomendaciones:

- Extender la herramienta en versiones posteriores para que trabaje con PostgreSQL en su versión 9 en adelante, ya que actualmente la herramienta trabaja con la versión 8.4.

- Ampliar las comparaciones de la herramienta a encontrar inconsistencias dentro de una misma Base Datos.

- Agregar al sistema una funcionalidad que permita la persistencia de los cambios que se hagan en las BD de manera que luego puedan ser consultados.

Referencias.

- ANON., 2010a. Estudio de EMC proyecta gran crecimiento de la información en 10 años. In: [online]. 27 April 2010. [Accessed 22 February 2012]. Available from: http://www.colombianproductions.com/mymit/joom1515/index.php?option=com_content&view=article&id 53: estudio-de-emc-proyecta-gran-crecimiento-de-la-informacion-en-10 anos&catid=38: noticias&Itemid=55.

- ANON., 2010b. Tipos de copia de seguridad. In: [online]. 15 2010. Available from: http://www.ite.educacion.es/formacion/materiales/85/cd/REDES_LINUX/backup/Tipos_de_copia_de_seguridad.html.

- EMC CORPORATION, 2008. El nuevo fenómeno mundial: la "Sombra Digital."In: [online]. 2008. [Accessed 12 June 2012]. Available from: http://argentina.emc.com/about/news/press/2008/20080311- 01.htm.

- EMS SOFTWARE DEVELOPMENT, 2011. Página del autor - EMS Software Development – Softpedia en español. In: [online]. 2011. [Accessed 11 June 2012]. Available from: http://www.softpedia.es/autor-EMS-Software-Development-19199.html.

- ESPINOSA, Jose Edgar Juarez, 2009. Respaldo de Información. In: [online]. 30 August 2009. [Accessed 24 February 2012].Available from: http://sistemasmultiempresa.com/RespaldodeInformacion.aspx.

- LEÓN, A. R. Sotolongo, RODRÍGUEZ, M. Gutierrez and VEITIA, B. Piñeiro, 2011. CRUD-PG. In: Revista Cubana de Ciencias Informáticas. 2011. Vol. 5, no. 1.

- MARTÍNEZ, M.C. Beatriz Beltrán, 2002. 2002. S.l.: s.n. Benemérita Universidad Autónoma de Puebla.

- ROSE, César E., 1993. Archivos, Organización y procedimientos. 1993. S.l.: s.n.

- SQL MAESTRO GROUP, 2012b. Página del autor - SQL Maestro Group - Softpedia en español. In: [online]. 9 April 2012. [Accessed 11 June 2012]. Available from: http://www.softpedia.es/autor-SQL-Maestro-Group-6790.html.

- STONEBRAKER, M. and ROWE, L. A., 1986. The design of POSTGRES. S.l.: s.n. ISBN 0897911911. ACM.

- SUMATHI, S. and ESAKKIRAJAN, S., 2007. Fundamentals of Relational Database Management Systems. Poland: Polish Academy of Sciences. ISBN 3-540-48397-7. Springer Berlin Heidelberg New York.

- SUMATHI, S. and ESAKKIRAJAN, S., 2007. Fundamentals of Relational Database Management Systems. Poland: Polish Academy of Sciences. ISBN 3-540-48397-7. Springer Berlin Heidelberg New York.

- THE POSTGRESQL GLOBAL DEVELOPMENT GROUP, 1996a. PostgreSQL 8.1.0 Documentation. 1996. S.l.: s.n.

- THE POSTGRESQL GLOBAL DEVELOPMENT GROUP, 1996b. PostgreSQL 8.1.0 Documentation.1996. S.l.: s.n.

- TROWBRIDGE, David, MANCINI, Dave, QUICK, Dave, HOHPE, Gregor, NEWKIRK, James and LAVIGNE, David, 2003. Enterprise Solution Patterns using Microsoft .NET. S.l.: Microsoft Corporation.