UNA METODOLOGÍA ÁGIL PARA LA OBTENCIÓN DE REPORTES EN ESTUDIOS CLÍNICOS

AN AGILE METHODOLOGY FOR OBTAINING CLINICAL STUDIES REPORTS

Autor:

Ing. Luis Enrique Ramírez Noy[I]

[I)] Profesor Instructor. Asesor técnico docente. Universidad de las Ciencias Informáticas. Carretera a San Antonio de los Baños km 2 1⁄2, reparto Lourdes, Boyeros, Ciudad de la

Habana, Tel: 837-2521. <noy@uci.cu>

RESUMEN:

Los numerosos estudios de personas con discapacidad llevados a cabo en el marco del ALBA han demostrado la necesidad de conformar un procedimiento o recurso que permita la obtención rápida de reportes y consultas. Esto ocurre porque no es posible prever, desde la etapa de diseño del sistema de recogida de datos, toda la información que será requerida o utilizada en función social. Los reportes, así como el propio sistema, también varían de un estudio a otro al desarrollarse en diferentes países con diferentes grados de profundidad. Se propone entonces la sistematización de una metodología que fue utilizada con éxito durante la Misión Dr. José Gregorio Hernández en la República Bolivariana de Venezuela. Esta alternativa permite el desarrollo rápido de consultas y reportes en bases de datos típicas para estos estudios. Tiene la ventaja adicional del bajo acoplamiento con el sistema de recogida de datos, lo que posibilita su extensión o modificación sin riesgo de alterar el flujo de trabajo en dicho sistema.

PALABRAS CLAVE:

Reportes dinámicos, consultas sql, estudios clínicos, desnormalización de bases de datos, vistas, estudio de personas con discapacidad.

ABSTRACT:

The numerous studies about people with disabilities conducted in the framework of ALBA, have demonstrated the need to obtain a method or resource that allows rapid retrieval of reports and queries. This occurs because it is not possible to anticipate, from the design stage of the data collection system, all the information that will be required or used. These reports, and the system itself, also vary from one study to another since they are carried out with different depth in different countries. Then, here is the proposal of systematization for the methodology that was used successfully on the Mission Dr. José Gregorio Hernández in Bolivarian Republic of Venezuela. This approach allows the fast development of queries and reports typical of such studies databases. It has de additional advantage of very low coupling with the data collect system which makes it expandable and modifiable without risk to alter normal work flow.

KEY WORDS:

Dynamical reports, sql queries, clinical studies, database denormalization, database views, disability people studies.

1. INTRODUCCIÓN

Durante los últimos diez años, las ciencias médicas de nuestro país han acumulado una enorme experiencia en el estudio psicosocial, pedagógico y clínico-genético de personas con discapacidad. Una parte de los excelentes resultados obtenidos en estos se debe a la gestión informática, que ha debido garantizar la precisión y rapidez de los resultados junto a la seguridad y fidelidad de los datos obtenidos en el terreno.

Este tipo de estudio se logró con éxito en varios países: Cuba, Venezuela, Bolivia, Ecuador, Nicaragua, San Vicente y las Granadinas [1]. La diversidad de entornos y la rapidez con que se necesitaron los resultados, requerían una adaptación en tiempo record de las aplicaciones informáticas. Un componente sensible en estas lo constituye el sistema de reportes, que posibilitó la estadística de la población con discapacidad, facilitó el diagnóstico integral, la ejecución de ayuda material a casos críticos y la mejora de las condiciones de vida de muchas personas. La obtención y clasificación de toda esta información mediante una aplicación informática diseñada para cada país contó con la anuencia de las entidades gubernamentales responsables [2] [3] [4].

La experiencia demostró que no es posible diseñar e implementar una aplicación de este tipo sin que se requiera incluir, luego del despliegue, funcionalidades adicionales [5]. Estas inclusiones deben hacerse sin afectar el flujo de trabajo en la recogida de datos del estudio y de manera transparente al sistema. Por este motivo se hizo amplio uso de aplicaciones web, basadas en PHP y MySQL. En todos los casos, se partió de un producto desarrollado en la Universidad de Ciencias Informáticas [6] que debió ajustarse y perfeccionarse durante su utilización en el estudio [7].

En este trabajo se presenta una metodología para el desarrollo de reportes en este tipo de sistemas.

2. EXPERIENCIA INFORMÁTICA

En el desarrollo del estudio psicosocial, pedagógico y clínico-genético de personas con discapacidad en la República Bolivariana de Venezuela, se contemplaron varios reportes, referentes todos a los resultados científicos del estudio [6]. Sin embargo, una vez comenzado este, se hizo evidente la necesidad de extraer otros tipos de información que no se habían identificado en el levantamiento de

requisitos para el diseño del sistema.

Esto sucedió porque aunque el estudio se concibió con un eminente carácter científico, el Gobierno Bolivariano y las entidades involucradas pretendían dar respuesta a las principales necesidades de las personas con discapacidad y sus familias sobre la marcha. Se pueden citar algunos ejemplos: alimentación a niños y ancianos, prótesis a personas de determinada edad y/o sexo, reconstrucción o acondicionamiento de viviendas, reubicación o matriculación de infantes en edad escolar, etc. El tipo de información solicitado resultó ser diferente incluso entre diferentes regiones del propio país.

Fueron necesarias, por tanto, nuevas consultas a la base de datos así como la modificación de algunos reportes originales para ajustarlos a las características del estudio en Venezuela.

Por lo general, el personal técnico que gestionaba el sistema no era parte del equipo de desarrollo; en el mejor de los casos, había tenido escasa participación. Dada la necesidad y la rapidez con que debían estar a punto los cambios, se hizo imprescindible encontrar un mecanismo o procedimiento mediante el cual se pudieran obtener resultados de manera confiable, rápida e independiente.

En un principio se escribieron consultas SQL *ad-hoc*, esto es, para cada tipo de información o reporte solicitado, se estableció el siguiente procedimiento:

1. Identificar las variables solicitadas.
2. Identificar las tablas de la base de datos relacionadas con las variables.
3. Escribir la consulta, típicamente como referencias cruzadas.
4. Verificar la consistencia con las tablas principales.

Este algoritmo tenía varios inconvenientes. En primer lugar se necesitaba de algunas pruebas de verificación para garantizar la fiabilidad de los resultados. Dichas pruebas podían implicar nuevas consultas a la base de datos o a especialistas, según la naturaleza de la información. En segundo lugar, había una tendencia a realizar asociaciones innecesarias, que en ocasiones causaron congestión en el sistema, afectando temporalmente el proceso de recogida de datos. Todo ello, sin contar que era necesario un estudio detallado de la estructura de una base de datos con más de 180 tablas.

Para mitigar estos inconvenientes se diseñó una manera estándar para realizar consultas a la base de datos, basada en una característica importante del estudio:

sin importar la cantidad de datos que se recogen, la cantidad de tablas en las que resultan estos ni las relaciones entre estas, la base de datos podía interpretarse como registros lineales de personas. De hecho, el trabajo en el terreno para el estudio consiste en la recolección de tantos datos sobre cada persona con discapacidad, como se indica en los instrumentos de investigación [7].

Obviamente, el proceso de normalización durante el diseño de la aplicación ocultó esta característica. Sin embargo, toda vez que la información estaba debidamente almacenada y relacionada no era necesario conservar estas restricciones para manejar los reportes o consultas.

2.1. El procedimiento práctico

Esencialmente, la solución se basó en manipular un modelo desnormalizado de la base de datos. Haciendo uso de vistas se realizó una abstracción de la estructura de la misma y luego las consultas se realizaron sobre estas vistas. Se utilizaron las vistas porque permiten la adquisición del estado de la base de datos de manera inmediata en cada ejecución, pero la principal ventaja es que resultan en objetos que no afectan el flujo de trabajo del sistema de recogida de datos. Además, las vistas contribuyen con la seguridad y fiabilidad de la información almacenada en la base de datos, ya que estas no pueden alterarla.

A partir del análisis de los instrumentos de investigación utilizados, se determinó que estaban divididos en secciones. La mayoría de dichas secciones consistía en formularios para opciones múltiples o únicas; y muy pocas requerían información textual. Se conformaron por tanto un conjunto de vistas estándar en la base de datos, denominadas vistas *gossip*. El objetivo de las mismas, como su nombre indica, es identificar de manera inmediata para cada persona, las opciones marcadas en el instrumento y la información textual.

Para las opciones múltiples y únicas se crearon en la vista tantas columnas como posibilidades de marcado y se llenó la coincidencia con un número 1 (uno) entero. La información textual simplemente se ubicó en la columna correspondiente al dato y en cualquier otro caso, el valor en la fila se dejó *null*. Esto no supone ningún problema desde el punto de vista de la consistencia, puesto que la vista ofrece información de solo lectura.

De esta manera se optimizaron las consultas para cuenta de elementos al lograr con una suma el mismo efecto. A la vez, si en una consulta se pretendía inspeccionar las variables de múltiple o única selección, bastaba establecer la

comparación con el entero 1(uno) en cualquier caso.

Aunque en teoría podría hacerse una sola vista para toda la base de datos, las relaciones mucho-mucho dificultan este proceder. De todos modos, disponer de unas pocas vistas relacionadas por la misma llave primaria todavía es mucho más simple que lidiar con un grupo grande de tablas normalizadas. También es importante destacar como particularidad de este estudio, que fue necesario dejar al operador la realización de consultas cruzadas con otras tablas (INNER JOIN) para especificar la ubicación geográfica en las consultas a la base de datos.

Los nombres de las columnas en las vistas se escribieron para que resultaran lo más similar posible a la nomenclatura de los instrumentos, de modo que al utilizar el autocompletamiento de código en la escritura de las consultas, la propia vista ayudó a la identificación de campos.

Así, se obtuvieron cinco vistas *gossip* que permitían examinar la información de cualquier persona que ya estuviese en la base de datos y facilitaban la escritura de consultas al tener nomenclatura descriptiva. En lo adelante no se necesitó estudiar la estructura relacional de la base de datos, puesto que la misma quedó simplificada a estas cinco vistas.

2.1.1 Los nomencladores

Los nomencladores son un parámetro vital para cualquier reporte. Las vistas *gossip* no disponen de estos directamente, ya que cada opción posible resulta en el valor numérico de la unidad; pero tampoco los necesitan.

Los nomencladores resultan implícitos en los nombres de las columnas en las vistas. Cualquier información relacionada con los nomencladores, requiere por tanto la manipulación de los nombres de las columnas. La operación más simple consiste en nombrar los campos SELECT con un alias idéntico al nombre del campo en la vista.

Las solicitudes de información típicamente consisten en listados de personas con determinadas necesidades o características, y es válido que en el mismo resulten celdas vacías y celdas marcadas. De todos modos, también se utilizó una alternativa para evitar listados con demasiadas columnas que hacían engorrosa su lectura o análisis: esta fue la función *group_concat* de MySQL.

Por medio de esta función es posible ejecutar una subconsulta a un grupo específico de columnas, pero resultando en una sola cadena de texto. De modo

que en la propia consulta se hizo posible agrupar opciones por categoría. El uso de *group_concat* resultó particularmente útil para las secciones con selección múltiple.

3. PROPUESTA

Esta experiencia práctica puede generalizarse y enriquecerse. No son pocas las circunstancias en las que deben realizarse consultas elaboradas manualmente, a una base de datos de un sistema informático en operación. Este tipo de consultas por lo general se hacen con urgencia y la información resultante debe ser fidedigna.

La metodología propuesta se ejecuta en tres etapas:

Etapa de desnormalización. Se intenta modelar la base de datos como una sola tabla *gossip*, cuyas columnas serán las columnas de todas las tablas reales; con la particularidad que los nomencladores se utilizan directamente como columnas independientes (Figura 1). Debido a las relaciones mucho-mucho, es más cómodo y útil manejar varias vistas *gossip*, siempre intentando que resulten relacionadas por la misma llave primaria.

```
CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER VIEW `vintelectualcg` AS
  select
    `trm`.`Folio` AS `Folio`,
    `tpersona`.`Id_Entidad_Federal` AS `Id_Entidad_Federal`,
    `tpersona`.`Id_Municipio` AS `Id_Municipio`,
    `tpersona`.`Id_Parroquia` AS `Id_Parroquia`,
    if((`tpersona`.`Edad` between 0 and 4),1,0) AS `0-4`,
    if((`tpersona`.`Edad` between 5 and 14),1,0) AS `5-14`,
    if((`tpersona`.`Edad` between 15 and 19),1,0) AS `15-19`,
    if((`tpersona`.`Edad` between 20 and 29),1,0) AS `20-29`,
    if((`tpersona`.`Edad` between 30 and 39),1,0) AS `30-39`,
    if((`tpersona`.`Edad` between 40 and 59),1,0) AS `40-59`,
    if((`tpersona`.`Edad` between 60 and 200),1,0) AS `60ymas`,
    if((`tpersona`.`Antecedentes_Consanguin` = 1),1,0) AS `Consanguineo`,
    if((`tdatos_clinico`.`Clasif_instrumento` = 1),1,0) AS `PrenatalGenI`,
    if((`tdatos_clinico`.`Clasif_instrumento` = 2),1,0) AS `PrenatalInespI`,
    if((`tdatos_clinico`.`Clasif_instrumento` = 3),1,0) AS `PrenatalAmbI`,
    if((`tdatos_clinico`.`Clasif_instrumento` = 4),1,0) AS `PerinatalI`,
    if((`tdatos_clinico`.`Clasif_instrumento` = 5),1,0) AS `PostnatalI`,
    if((`tdatos_clinico`.`Clasif_instrumento` = 6),1,0) AS `PsicosisI`,
    if((`tdatos_clinico`.`Clasif_instrumento` = 7),1,0) AS `InclasifI`,
    if(((`tdatos_clinico`.`Clasif_result_analisis` = 1) or ((`tdatos_clinico`.`Clasif_genetista` = 1
    if(((`tdatos_clinico`.`Clasif_result_analisis` = 2) or ((`tdatos_clinico`.`Clasif_genetista` = 2
```

Figura 1. Fragmento de la definición de una de las vistas *gossip* en el estudio realizado. Esta vista incluyó el contenido de cuatro tablas con información clínica genética de personas con discapacidad intelectual.

Los campos de las vistas se conformarán en base a tres clasificaciones de formularios:

1. *Selección múltiple y selección única*. Se representan con tantas columnas como nomencladores tenga el formulario. Cada indicación en la tabla real se marca con 1(uno) entero en la vista.

2. *Campos textuales.* Se representan de manera idéntica en las vistas.

Esta etapa es la más trabajosa, puesto que requiere un análisis completo de la base de datos y las relaciones entre sus tablas. Este análisis solo se hace una vez y por tanto sería adecuado que los propios desarrolladores del sistema sean los que proporcionen las vistas de esta etapa.

En cualquier caso, los nombres de las columnas para las vistas deben ser lo más descriptivos posible, para facilitar la ejecución de la etapa siguiente.

Etapa operativa. Como el nombre indica, esta es la etapa de trabajo. Simplemente se hace uso de la abstracción que ofrecen las vistas *gossip*. Sin embargo, es importante tomar en consideración algunos aspectos:

1. Es muy recomendable utilizar alguna herramienta con autocompletamiento de código para la elaboración de las consultas. De esta manera se aprovechan los nombres descriptivos de las columnas en las vistas *gossip*.

2. El conteo de elementos se lleva a cabo mediante operaciones de suma sobre las columnas de interés, como se ilustra en la Figura 2.

```
CREATE OR REPLACE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER VIEW `FM_por_edad` AS
  select
    `tparroquia`.`Parroquia` AS `Parroquia`,
    sum(`vdiscapacidadps`.`0-4`) AS `e0a4`,
    sum(`vdiscapacidadps`.`5-14`) AS `e5a14`,
    sum(`vdiscapacidadps`.`15-19`) AS `e15a19`,
    sum(`vdiscapacidadps`.`20-29`) AS `e20a29`,
    sum(`vdiscapacidadps`.`30-39`) AS `e30a39`,
    sum(`vdiscapacidadps`.`40-59`) AS `e40a59`,
    sum(`vdiscapacidadps`.`60ymas`) AS `e60omas`
  from
    (`vdiscapacidadps` join `tparroquia` on(((`tparroquia`.`Id_Parroquia` = `vdiscapacidadps`.`Id_Parroquia
  where
    ((`vdiscapacidadps`.`Id_Municipio` = ".$Municipio.") and (`vdiscapacidadps`.`Id_Entidad_Federal` = ".$E
  group by
    `tparroquia`.`Id_Parroquia`
  order by
    `tparroquia`.`Id_Parroquia`;
```

Figura 2. Definición de una consulta para la obtención de totales. En este caso para obtener la cantidad de personas con discapacidad físico motora según rangos de edad y ubicación geográfica.

3. La discriminación (IF o WHERE) de todos los elementos en formularios de opciones múltiples y únicas se hace siempre mediante la comparación del parámetro de interés con el valor 1(uno) entero (Figura 3).

4. Para obtener listados compactos y legibles, se puede hacer uso de la función *group_concat* de MySQL o alguna equivalencia si se usa otro gestor [8]. En este caso siempre deberá realizarse una subconsulta sobre el grupo de columnas de la vista implicados. No se necesitan relaciones con tablas de nomencladores.

```
SELECT
  CONCAT(tpersona.Nombre_Persona," ",tpersona.Apellidos_Persona) AS Nombre,
  tparroquia.Parroquia,
  tpersona.Edad,
  tpersona.Lugar_Hab AS Dirección,
  GROUP_CONCAT(ttelefono.Numero) AS Teléfono,
  if(vdiscapacidadps.FM = 1,"FM","") AS FM,
  if(vdiscapacidadps.Visual = 1,"Visual","") AS Visual,
  if(vdiscapacidadps.Auditivo = 1,"Auditivo","") AS Auditivo,
  if(vdiscapacidadps.Mental = 1,"Mental","") AS Mental,
  if(vdiscapacidadps.Visceral = 1,"Visceral","") AS Visceral,
  if(vdiscapacidadps.Multiple = 1,"Múltiple","") AS Multiple,
  if(vintelectualps.Ligero = 1,"Ligero","") AS Ligero,
  if(vintelectualps.Moderado = 1,"Moderado","") AS Moderado,
  if(vintelectualps.Severo = 1,"Severo","") AS Severo,
  if(vintelectualps.Profundo = 1,"Profundo","") AS Profundo,
  if(vintelectualps.Adefinir = 1,"A definir","") AS Adefinir
FROM
  tpersona
  INNER JOIN tmunicipio ON (tpersona.Id_Municipio = tmunicipio.Id_Municipio)
  AND (tpersona.Id_Entidad_Federal = tmunicipio.Id_Entidad_Federal)
  INNER JOIN tparroquia ON (tpersona.Id_Entidad_Federal = tparroquia.Id_Entidad_Federal)
  AND (tpersona.Id_Municipio = tparroquia.Id_Municipio)
  AND (tpersona.Id_Parroquia = tparroquia.Id_Parroquia)
  LEFT JOIN ttelefono ON (ttelefono.Folio = tpersona.Folio)
  LEFT JOIN vdiscapacidadps ON (vdiscapacidadps.Folio = tpersona.Folio)
  LEFT JOIN vintelectualps ON (vintelectualps.Folio = tpersona.Folio)
WHERE
  tpersona.Id_Municipio = 1
GROUP BY
  tpersona.Folio
```

Figura 3. Definición de una consulta sobre vistas predefinidas. En este caso para obtener el listado de todas las personas y sus discapacidades.

Etapa de entrega. La entrega de la información pudiera parecer de escasa relevancia. Sin embargo, herramientas ofimáticas para hojas de cálculo como Microsoft Excel y OpenOffice Calc pueden suplir adecuadamente las necesidades de formatos de los clientes e incluso completar deficiencias en la elaboración de consultas. Y ya que los formatos de estas herramientas son los requeridos típicamente por quien solicita la información, no está nada mal explotar sus

posibilidades.

Las operaciones más importantes son:

1. *Transposición*. La opción de pegado especial tiene esta funcionalidad. Posibilita transponer las filas y columnas de la tabla obtenida en la consulta en el momento de copiarla en una hoja de cálculo.

2. *Concatenación*. Por cuestiones de tiempo o habilidad del operador, podría ser trabajoso utilizar la función *group_concat*. El uso de la función *concatenar* de las hojas de cálculo, puede suplir convenientemente esta dificultad con mínimo esfuerzo.

3. *Reemplazo*. Si no se ha elaborado adecuadamente la consulta, pueden resultar valores poco descriptivos o propensos a malas interpretaciones en el listado final. Por ejemplo sería adecuado reemplazar todos los números uno que ofrece la vista por **X** o la palabra **Sí**. Otro tipo de reemplazo útil es el que debe aplicarse para eliminar comas repetidas, ya que las operaciones de concatenación pueden resultar con comas consecutivas.

3.1. Repositorio de consultas

Luego de realizada cada consulta, es importante guardar su código adecuadamente indexado. De modo que sea fácil su posterior ubicación y reutilización.

Un elemento importante en el repositorio de consultas es la parametrización. Las condiciones WHERE de una consulta variarán inevitablemente con el tiempo, por lo que es recomendable tomar en cuenta esta posibilidad en el código de la misma. Claro que en este punto la experiencia práctica tiene la última palabra.

En cualquier caso, el objetivo del repositorio de consultas es agilizar la obtención de los reportes.

4. ALGUNAS CONSIDERACIONES ADICIONALES

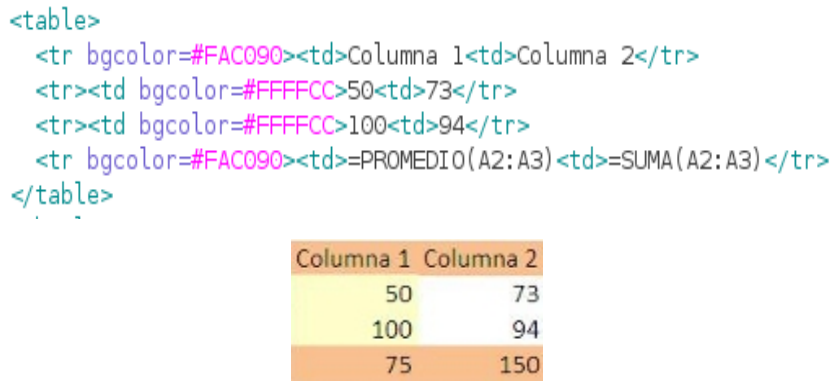
Las vistas *gossip* también pueden tener un papel importante en el monitoreo de la consistencia de la información en las tablas de la base de datos. Ya que es mucho más fácil acceder a la información de las tablas, pueden detectarse mediante

consultas simples:

1. Entradas nulas.
2. Entradas de selección única con más de una entrada.
3. Desemparejamiento de totales.

Las vistas *gossip* pueden ser la base de otras vistas de control que pueden indicar inmediatamente cualquiera de estos problemas para facilitar su corrección e incluso detectar errores en el propio sistema de recogida de datos.

Existe otra posibilidad importante, propiciada por las vistas *gossip*. Dado el bajo acoplamiento, nada impide el desarrollo de una aplicación independiente para la obtención de reportes. Esta podría hacer uso del repositorio de consultas parametrizadas para obtener reportes directamente desde un navegador web. Acá también se puede explotar una característica importante de las herramientas ofimáticas: son capaces de interpretar código HTML como hojas de cálculo, incluso hasta el manejo de fórmulas. La Figura 4 muestra cómo puede procederse para obtener reportes sin disponer de módulos de exportación a formatos *.xls* o *.ods* y sin necesidad de acceder al código de la aplicación de recogida de datos ni afectar su funcionamiento.



```
<table>
  <tr bgcolor=#FAC090><td>Columna 1<td>Columna 2</tr>
  <tr><td bgcolor=#FFFFCC>50<td>73</tr>
  <tr><td bgcolor=#FFFFCC>100<td>94</tr>
  <tr bgcolor=#FAC090><td>=PROMEDIO(A2:A3)<td>=SUMA(A2:A3)</tr>
</table>
```

Columna 1	Columna 2
50	73
100	94
75	150

Figura 4. Una hoja de cálculo interpreta el código html (arriba) como la tabla que representa incluyendo fórmulas (abajo).

Esta característica fue aprovechada para la obtención de los informes parciales y finales a nivel estatal y nacional, durante la Misión Dr. José Gregorio Hernández en Venezuela. Para ello se desarrolló una aplicación web con php, especializada en reportes y basada en la metodología descrita.

5. CONCLUSIONES

La operación de un sistema de recogida de datos siempre requerirá consultas adicionales a las diseñadas. Estas consultas por lo general se deben resolver con rapidez y exactitud. Pero el personal técnico que maneja el sistema casi nunca ha sido parte del equipo de desarrollo y ello supone una dificultad.

La aplicación de esta metodología supone varias ventajas:

1. Menor tiempo de obtención de estas consultas adicionales.
2. No se requiere un conocimiento detallado de la base de datos.
3. No se interfiere con el funcionamiento del sistema de recogida de datos.
4. Se asegura la fidelidad de los datos obtenidos.

Adicionalmente, se propone la integración o el uso de herramientas ofimáticas que dan el acabado final a la información solicitada.

La efectividad de estos procedimientos se validó en el desarrollo de la Misión Dr. José Gregorio Hernández en la República Bolivariana de Venezuela.

6. REFERENCIAS BIBLIOGRÁFICAS

1. Vicepresidencia de la República de Ecuador. Declaración de Quito. [En línea] 2010. Disponible en: http://www.vicepresidencia.gov.co/Noticias/2010/Paginas/101210c.aspx.

2. Misión Dr. José Gregorio Hernández. [En línea] 2008. [Citado el: 21 de 3 de 2011.] Disponible en: http://misionjgh.blogspot.com/.

3. Ministerio de Relaciones Exteriores de Cuba. Colaboración en la Misión Moto Méndez. [En línea] 2010. [Citado el: 20 de 3 de 2011.] Disponible en: http://www.cubaminrex.cu/Cooperacion/2010/mision4.html.

4. Viceprecidencia de la República de Ecuador. Misión Manuela Espejo. [En línea] 2010. [Citado el: 20 de 3 de 2011.] Disponible en: http://www.vicepresidencia.gob.ec/programas/manuelaespejo/mision?start=1.

5. Gobierno Bolivariano de Venezuela. Programa de la Jornada Científica de la Misión Dr. José Gregorio Hernández (23 de octubre de 2008). [En línea]. [Citado

el: 27 de 3 de 2011.] Disponible en: http://www.vive.gob.ve/eventoDespleg.php?id_e=174&mostrar=todos&mes=&ano=&tipo=&pag=.

6. Rigó Portillo D. Sistema Automatizado de Registro de Retraso Mental Cubano. Ciudad de la Habana: s.n., 2007.

7. Hernández Marrero FM. Desarrollo de la aplicación informática del Estudio Integral de las personas con discapacidad en Venezuela, misión Dr. José Gregorio Hernández. Ciudad de la Habana: s.n., 2008.

8. Postgres. PostgreSQL SQL tricks. [En línea] [Citado el: 29 de 3 de 2011.] Disponible en: http://postgres.cz/index.php/PostgreSQL_SQL_Tricks#MySQL_function_group_concat_in_PostgreSQL.

Recibido: 7 noviembre 2011.
Aprobado: 8 mayo 2012.