MANUAL TÉCNICO

ETAB

Esta página se ha dejado vacía a propósito

Índice de contenidos

Esta página se ha dejado vacía a propósito

Introducción

El presente manual técnico describe cada uno de los componentes del sistema eTAB y los pasos necesarios para instalarlo.

El sistema de información eTAB es parte de la iniciativa Salud Mesoamérica 2015 (SM2015). Esta es una iniciativa cuyo fin es reducir las inequidades en salud que están afectando al 20 por ciento mas pobre de la población en Centro America y Mexico. Esta iniciativa también tiene como objetivo apoyar los esfuerzos de los gobiernos de la región para alcanzar los Objetivos del Milenio.

Salud Mesoamérica 2015, pone especial atención a la áreas de salud reproductiva, nutrición maternal y neonatal maternal inmunización, y la prevención y control del dengue y la malaria. Para este fin Salud Mesoamérica 2015trabajara en conjunto con los ministerios de salud de la región y el Sistema de Salud Publica Mesoamericano. Este proyecto es parte de la plataforma de integración regional conocido como Proyecto Mesoamerica.

Los resultados esperados de incluyen una reducción significante en las tasa de mortalidad infantil para niños de menos de cinco años. Esta iniciativa también esta busca reducir la malnutrición crónica en la niñez y las mujeres embarazadas. Estos cambios son críticos para mejorar las estadísticas sobre partos y para ofrecer mejores condiciones para el crecimiento del recién nacido. A su vez esta iniciativa busca tener un efecto directo en comunidades pobres sobre la cobertura y calidad de vacunas, control pre y post natal y acceso a planificación familiar entre otros servicios.

Salud Mesoamérica 2015 espera generar conocimiento de relevancia global sobre como aumentar asistencia en salud de bajo costo en comunidades pobres. Para este fin el sistema de información eTAB permite analizar y dar seguimiento a los indicadores en salud con los que trabaja este proyecto.

Esta página se ha dejado vacía a propósito

Capítulo 1

Tecnologías utilizadas

El Tablero eTAB es un servicio Web disponible para que dependencias del sistema de salud suban sus datos para poder analizarlos, generar gráficas y reportes.

La aplicación cuenta con un módulo para efectuar la extracción,transformación y carga de datos (ETL) desde diferentes fuentes. Estos datos son agregados y almacenados en una base de datos relacional (OLTP). Los datos están organizados por catálogos de referencia e Indicadores medibles. Los usuarios del sistema pueden administrar estos indicadores y catálogos y todos sus tributos usando el las herramientas que brinda el sistema. Para efectuar consultas en línea los datos son agregados dentro de tablas optimizadas para el análisis en linea (OLAP). Las tablas para análisis son actualizadas periódicamente usando procedimientos almacenados de PostgresSQL.
La gestión de consultas a las tablas de análisis OLAP se realiza por medio de un servidor dedicado. La interacción entre el servidor OLAP y el resto de la aplicación se realiza por medio de consultas AJAX. El resultado de las consultas al servidor OLAP, es porcesado usando JQuery y graficado usando la libreria de gráficos D3.

Todo el software utilizado para creación del SIIG/eTAB son paquetes de software libre. Estos incluyen:

- GitHub: Gestor de control de versiones de código fuente
- Apache: Servidor de paginas web
- PostgreSQL: Gestor de bases de Datos

- Symfony: Entorno de desarrollo para PHP
- PHP: Lenguaje de desarrollo de la Aplicación eTAB
- Java/OpenJDK: Lenguaje de desarrollo del servidor OLAP
- Pentaho: Servidor OLAP desarrollado en Java
- Saiku: Interfaz de analisys para consultas a servidor Pentaho
- D3.js: Librería para la generación de gráficos
- JQuery: Lenguaje para interfaces de usuario
- RabbitMQ: Servidor de Mensajería
- EasyBook: Generador de documentos en formato PDF
- Bootstrap: Framework para interfaces de usuario

1.0.1 Gestor de base de datos

[PostgreSQL] (http://www.postgresql.org/)

Versión 9.1 Actualmente el sistema únicamente puede utilizar PostgreSQL por la siguiente razón: La aplicación debe proveer la capacidad de analizar datos para cualquier indicador. Cada indicador esta construido con varios datos y relacionados por una formula almacenada en el sistema.
Es posible crear una tabla por cada grupo de datos, con la limitante de que es necesario conocer el dato antes de guardarlo, lo cual no es sostenible a futuro.

La base de datos necesita guardar datos sin conocer de antemano sus características.
Esto se logra usando un esquema de datos generico EAV (entidad-atributo-valor). El manejo de esquemas EAV es implementado de diferentes formas para diferentes gestores de bases de datos,el Tablero eTAB usa la implementación de Postgres la cual crea un tipo especial de dato llamado HSTORE.

1.0.2 Servidor Web

[Apache2] (http://www.apache.org)

Apache es un servidor Web de código abierto, se ha realizado sobre Apache versión 2.2

1.0.3 Framework de desarrollo/Servidor

[PHP] (http://www.php.net)

[Symfony] (http://symfony.com/)

[GitHub] (https://github.com/)

El lenguaje que se ha utilizado es PHP 5.3.18 dentro de una estructura de desarrollo MVC manejada Symfony versión 2.4 Cada mimbro del equipo de desarrollo usa un aplicativo diferente para escribir/modificar el código fuente. Los mas populares popular es Netbeans(version libre para PHP) y Nano. Para manejar cambios y mejoras al código fuente se uso Github. La totalidad del código fuente esta disponible en https://github.com/erodriguez-minsal/SIIG

1.0.4 Framework JavaScript

jQuery (http://jquery.com/) versión 1.8.3 junto a jQuery UI (http://jqueryui.com/)

1.0.5 Framework para interfaces de usuario

Bootstrap (http://twitter.github.com/bootstrap/)

Bootstrap es un framework que hace HTML, CSS y Javascript simple y flexible para componentes e interacciones de interfaz de usuarios populares.

1.0.6 Librería para gráficos

D3 (http://d3js.org/)

Antes conocida como Protovis, D3 es una biblioteca de JavaScript para manipular documentos basados en datos. D3 ayuda dar vida a los datos usando HTML, SVG y CSS. D3 enfatiza los estándares Web ofreciendo todas las capacidades de los navegadores modernos sin ligarse a una estructura propietaria. A diferencia de otras librerías, D3 no crea imágenes, sino que interactúa la pagina para crear los gráficos usando elementos de HTML5 como Canvas y SVG.

1.0.7 Mensajería

RabbitMQ (http://www.rabbitmq.com/)

La carga de datos se apoya de las librerías de este paquete para crear una ‘lista de espera’ para evitar que el servidor se sature al recibir demasiadas peticiones simultaneas.

1.0.8 Servidor de Cubos OLAP

Pentaho (http://community.pentaho.com/)

La version 'comunidad' del servidor de cubos OLAP Pentaho es un proyecto de código abierto desarrollado usando Java 6, se uso la ultima version disponible 4.8, el paquete incluye el servidor de aplicaciones Tomcat.

1.0.9 Interfaz de Analisis de cubos

Saiku (http://community.pentaho.com/)

Saiku es una aplicacion de JAVA que oferece una interfaz escrita en JQuery para analizar cubos OLAP. En este proyecto se uso la version 'plugin' para Pentaho, version 2.4.

1.0.10 Documentación

La mayoría de la documentación ha sido escrita en formato markdown y se ha utilizado easybook (http://easybook-project.org/) para la gen

Capítulo 2

Instalación del eTAB

2.1 Requerimientos

- Servidor Web
- Gestor de base de datos
- PHP 5.3.8+
- Java 6

2.2 Instalación de Symfony2

2.2.1 Instalación de los requerimientos desde un servidor Debian

Es muy importante poner atención al indicador "#" significa que el comando debe ser ejecutado como usuario **root** y "**$**" que debe ser ejecutado como un **usuario normal**, en ambos casos desde una **consola de comandos**.

Actualizamos la lista de paquetes del sistema operativo.

```
# apt-get update
```

Instalamos todas las librerías y aplicaciones que se utilizan en el sistema (PHP, PostgreSQL y Git).

```
# apt-get install php5 php5-pgsql php5-sqlite sqlite php5-xdebug php-apc
php5-cli php5-xsl php5-intl php5-mcrypt apache2 postgresql acl git-core
curl postgresql-contrib php5-ldap php5-mysql php5-sybase php5-json
```

2.2.2 Crear usuario y directorio de trabajo

El directorio y usuario a utilizar pueden variar de acuerdo a los que se deseen elegir en cada instalación. Como ejemplo se usará un usuario llamado **siig** y el directorio de instalación **/var/www/siig**

Creamos el usuario

adduser siig

Creamos el directorio

mkdir /var/www/siig

Asignamos como dueño del directorio al usuario que acabamos de crear, en nuestro caso es siig

chown siig:siig /var/www/siig

Nos cambiamos al usuario que es dueño de la directorio siig

su siig

Accedemos a la carpeta web del Apache

$ cd /var/www

2.2.3 Obtener el código fuente

Puedes descargarlo desde: https://github.com/erodriguez-minsal/SIIG/tarball/master o clonar el repositorio

$ git clone https://github.com/erodriguez-minsal/SIIG.git siig

Recuerda que actualmente estamos en el directorio */var/www* y el último parámetro del git clone es la carpeta en donde se descargará el código fuente del repositorio, en este nuestro caso es *siig*.

`NOTA: A partir de este punto todos los comandos se deben ejecutar dentro de la carpeta en que se ha descargado el código fuente.`

Si se desea tener las aportaciones del equipo SM2015 Chiapas, es necesario cambiar de rama el repositorio, esto se logra ejecutando la siguiente sentencia:

git checkout chiapas

2.2.4 Instalar composer

Composer (http://getcomposer.org) es una librería de PHP para el manejo de dependencias. Para instalarlo, dentro de la carpeta donde descargaste el código fuente se debe ejecutar:

$ curl -s https://getcomposer.org/installer | php

2.2.5 Instalar todas las librerías necesarias

$ php composer.phar install

Dado que Symfony2 es un proyecto Open Source, depende de librerías y paquetes de terceros, por lo que puede presentarse el caso que al ejecutar composer install se produzca un error de dependencias. Al ejecutar el composer install este lee el archivo composer.json donde se encuentran las dependencias del proyecto. Al terminar la instalación este crea el archivo composer.lock el cual contiene las especificación exacta de las versiones de los paquetes instalados, por lo que puede utilizar como alternativa el siguiente archivo composer.lock (https://github.com/erodriguez-minsal/SIIG/tree/chiapas/app/Resources/doc/composer.lock) e intentar nuevamente la instalación, este solo en caso de presentarse el problema de dependencias.

Durante la instalación se solicitarán los siguientes parámetros (Si desea conservar el valor por defecto para cada entrada es suficiente con presionar Enter para confirmar el valor):

database_driver: pdo_pgsql

Esta variable contiene el driver php que manejará la comunicación con la base de datos en la capa de acceso a datos, dado que la plataforma trabaja con PostgreSQL su valor por defecto será pdo_pgsql, se recomienda no modificar este valor.

database_host: localhost

Se refiere al host donde se encuentra alojado el servidor de base de datos, en nuestro caso es el mismo que el servidor web.

database_port: null

El puerto del manejador de base de datos, en nuestro caso es null ya que utiliza el puerto por defecto para el PostgreSQL (5432).

database_name: indicadores

Nombre de la base de datos, mas adelante se creará la base de datos con ayuda de Symfony.

database_user: admin

Nombre del usuario para la base de datos, este se creará mas adelante en la sección Configuración de PostgreSQL.

database_password: rodriguez

Contraseña del usuario para la base de datos.

mailer_transport: smtp

Protocolo para la transferencia de correo electrónico.

mailer_host: localhost

Servidor de correo electrónico.

mailer_user: null

Usuario para el servidor de correo electrónico.

mailer_password: null

Contraseña del usuario para el servidor de correo electrónico.

locale: es_SV

Lenguaje por defecto para la aplicación.

secret: 295125e6c66ab2a1038b62ad3c910733510

Esta es una cadena que debe ser única, se utiliza para la generación de las tokens CSRF, pero que podría ser utilizado en cualquier otro contexto en donde una cadena única es útil, como por ejemplo, la encriptación de las contraseñas de usuario.

archivo_vitacora: %kernel.logs_dir%/minsal.log

Archivo donde se guardará el registro de eventos de la aplicación. %kernel.logs_dir% es una variable de Symfony2 que hace referencia a la ruta relativa app/logs/ (con respecto a directorio de instalación).

carpeta_siig_mondrian: %kernel.root_dir%/mondrian/

Carpeta de esquemas generados por la aplicación para Pentaho, esta variable es utilizada por Saiku. %kernel.root_dir% es una variable de Symfony2 que hace referencia a la ruta relativa app/ (con respecto a directorio de instalación).

conexion_bd_pentaho: Minsal

Nombre de conexión a base de datos dentro de Pentaho.

listado_metadata: datasources.siig

Archivo que lista esquemas existentes y su conexión. Este archivo debe estar ligado al fichero ../pentaho-solutions/system/olap/datasources.xml

2.3 Configuración

2.3.1 Servidor web

Esto es para una instalación de prueba en una máquina local, la instalación real en un servidor el administrador de servicios deberá realizar esta configuración con los parámetros más adecuados: ip, dominio, configuración en el DNS, etc.

2.3.1.1 Configurar un VirtualHost

Creamos el archivo para la definición del VirtualHost

nano /etc/apache2/sites-available/siig.localhost

El contenido será similar a esto:

```
<VirtualHost 127.0.0.7>

    ServerName siig.localhost
    DocumentRoot /var/www/siig/web

    <Directory /var/www/siig/web >
         Options Indexes FollowSymLinks MultiViews
```

```
        AllowOverride All
        Order allow,deny
        allow from all
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/siig-error.localhost.log
    # Possible values include: debug, info, notice,
warn, error, crit,
    # alert, emerg.
    LogLevel warn
    CustomLog
${APACHE_LOG_DIR}/siig-access.localhost.log combined

</VirtualHost>
```

En el archivo /etc/hosts agregamos la línea

127.0.0.7 siig.localhost

Habilitamos el VirtualHost

a2ensite siig.localhost

También es recomendable activar el módulo mod_rewrite

a2enmod rewrite

Reiniciar apache

/etc/init.d/apache2 restart

2.3.2 Permisos sobre carpetas

Es necesario tener soporte para ACL (https://help.ubuntu.com/community/FilePermissionsACLs) en la partición en que está el proyecto y luego ejecutar

$ setfacl -R -m u:www-data:rwx -m u:`whoami`:rwx app/cache app/logs web/uploads

$ setfacl -dR -m u:www-data:rwx -m u:`whoami`:rwx app/cache app/logs web/uploads

2.3.3 Verificar la configuración

Entra a la siguiente dirección desde el navegador *http://siig.localhost/config.php*. Si aparece algún error debe ser corregido antes de continuar.

2.4 Configuración de PostgreSQL

2.4.1 Editar archivo de configuración

Como usuario root realizar:

1. Editar el archivo */etc/postgresql/9.1/main/pg_hba.conf*
2. Cambiar la siguiente línea, sustituir la última palabra por *md5*

    local all all md5

Reiniciar PostgreSQL

    # /etc/init.d/postgresql restart

2.4.2 Crear el usuario dueño de la base de datos

Se creará el usuario dueño de la base de datos, las opciones utilizadas dependerán de los criterios que se quieran seguir, se muestra un ejemplo, ejecutar *createuser --help* para la explicación de las opciones. El nombre utilizado y la clave debe corresponder con los parámetros especificados al ejecutar *php composer.phar install* en unas secciones anteriores

    # su postgres

    $ createuser -d -S -R -P admin;

Al finalizar presionar la combinación Ctrl+D 2 veces para regresar al usuario siig y continuar con la instalación.

2.4.3 Crear la base de datos

Symfony hace uso del bundle Doctrine (http://gitnacho.github.io/symfony-docs-es/book/doctrine.html) para el manejo de la capa de datos.

    $ app/console doctrine:database:create

Este comando creará la base de datos.

    $ app/console doctrine:schema:update --force

Este comando creará la estructura de las tablas del sistema.

2.4.4 Cargar datos iniciales

```
$ app/console doctrine:fixtures:load
```

Con este comando se insertan los datos iniciales del sistema.

2.4.5 Crear un usuario administrador del SIIG

Symfony hace uso del bundle FOSUser (https://github.com/FriendsOfSymfony/FOSUserBundle/blob/master/Resources/doc/index.md) para la administración de usuarios.

```
$ app/console fos:user:create --super-admin
```

Con este *usuario* se iniciará **sesión** en la aplicación web al terminar instalación.

2.4.6 Instalación de HStore

HStore (http://www.postgresql.org/docs/9.1/static/hstore.html) es un tipo especial de campo de PostgreSQL.

- Conectarse al servidor de base de datos con el usuario postgres, esto se logra ejecutando desde una ventana de comando las siguientes sentencias:

```
# psql -U postgres -d database_name
```

El parametro database_name es el que se estableció anteriormente al ejecutar composer install. Ahora procedemos a crear la extensión hstore:

```
create extension hstore;
```

Salir de la línea de comandos del PostgreSQL con \q.

- Crear la tabla especial que no se manejará con el ORM, hacerlo con el usuario dueño de la base de datos (no con el usuario postgres, a menos que este mismo sea el dueño de la base de datos).

```
# psql -U database_user -d database_name
```

El parametro database_user y database_name se establecieron anteriormente al ejecutar composer install.

```
CREATE TABLE fila_origen_dato(
    id_origen_dato integer,
    datos hstore,
    ultima_lectura timestamp,

    FOREIGN KEY (id_origen_dato) REFERENCES
origen_datos(id) on update CASCADE on delete CASCADE
);
```

Salir de la línea de comandos del PostgreSQL con \q.

Si se prefiere, hay una alternativa de interfaz gráfica para la administración de PostgreSQL, este es *pgAdmin*. Para instalar este administrador ejecutar la siguiente sentencia:

```
# aptitude install pgadmin3
```

2.5 Instalación de RabbitMQ

RabbitMQ (http://www.rabbitmq.com/) es un sistema de mensajería empresarial completo y altamente confiable basado en el estándar AMQP Charla sobre RabbitMQ (http://www.symfony.es/noticias/2011/07/06/desymfony-2011-reduciendo-el-acoplamiento-entre-aplicaciones-con-rabbitmq/) . En este proyecto será utilizado para la carga masiva de datos y así evitar cuelgues o saturación del servidor.

- Agregar el repositorio

  ```
  # sh -c 'echo "deb http://www.rabbitmq.com/debian/ testing main" >>
  /etc/apt/sources.list'
  ```

- Agregar la clave pública

  ```
  # wget http://www.rabbitmq.com/rabbitmq-signing-key-public.asc
  ```

  ```
  # apt-key add rabbitmq-signing-key-public.asc
  ```

- Ejecutar

  ```
  # apt-get update
  ```

- Instalar el paquete

  ```
  # apt-get install rabbitmq-server
  ```

- Verificar que el servicio de rabbitmq esté corriendo

  # /etc/init.d/rabbitmq-server start

- Habilitar la interfaz web de administración

  # rabbitmq-plugins enable rabbitmq_management

  # /etc/init.d/rabbitmq-server restart

- Cargar la interfaz web: entrar a la dirección *http://localhost:55672/mgmt/*. El usuario por defecto es **guest** y la clave **guest**

- Iniciar las colas

  $ src/MINSAL/IndicadoresBundle/Util/iniciar_colas.sh

Pueden aparecer mensajes de aviso como "/usr/bin/nohup: redirecting stderr to stdout" solo debemos presionar ENTER

- Además es necesario configurar el **CRON** para que ejecute periódicamente la carga de datos, con esto se llamará al proceso origen-dato:cargar que verificará para cada indicador si le corresponde realizar la carga de datos según se haya configurado: diario, mensual, bimensual, trimestral, cuatrimestral, semestral o anual. Un ejemplo podría ser crear el archivo: /etc/cron.d/carga-php-siig con el siguiente contenido:

  #Ejecutar cada dia a las 00:00

  0 0 * * * www-data test -x /usr/bin/php && /usr/bin/php /var/www/siig/app/console origen-dato:cargar

2.6 Instalación de Servidor de Análisis Pentaho

Pentaho es un servidor de análisis (Business Inteligence) modular que ofrece herramientas para la carga de datos(ETL), análisis dimensional (OLAP), minería de datos y reportes entre otros. A continuación:

1-Instalaremos el modulo base de Pentaho - edición comunidad.

2- Configuraremos su servicio de análisis dimensional conocido como Mondrian.

3- Instalaremos la aplicación de visualización de cubos OLAP llamada SAIKU.

4- Modificaremos Apache: URL del SIIG apuntando a Pentaho.

5- Crear y Publicar Reportes por Indicador.

El objetivo es usar el servidor Pentaho+Saiku para analizar los datos del SIIG y a la vez integrar esta aplicación dentro de la plataforma del SIIG de forma que el usuario no se percate de que esta usando una aplicación externa.

2.6.1 Instalación de Pentaho

Pentaho es una aplicación escrita en JAVA que utiliza persistencia (Hibernate) un servidor de aplicaciones (Tomcat). Pentaho servirá como plataforma para ejecutar nuestra aplicación de análisis de datos.

- Instalar Java y soporte de Postgres:

```
# apt-get install openjdk-6-jre libpg-java
```

- Descargar la ultima versión del servidor de Pentaho en:

http://community.pentaho.com/projects/bi_platform/

Y luego descomprimir el archivo descargado en la carpeta que elijamos.

El archivo comprimido del servidor de Pentaho (biserver-ce-X.X-estable.tar) contiene dos carpetas con dos servicios diferentes:

- biserver-ce, la plataforma sobre la cual se instalaran nuevas aplicaciones visibles a los usuarios, accesible por el puerto 8080.

- aministration-console, la interfaz de administración del servidor que permite manejar cuentas de usuario, roles y conexiones a bases de datos, accesible desde el puerto 8099.

Cada uno de estos dos servicios tiene su script de inicio correspondiente y credenciales por defecto. A continuación eliminaremos el sistema de seguridad interno de Pentaho, para que no pida credenciales y así facilitar la integración con el resto del sistema SIIG. Estos cambios afectan ambos servicios: la plataforma de Pentaho y la consola de administración. Para eliminar el uso de credenciales basta seguir las instrucciones del manual oficial de Pentaho:

http://wiki.pentaho.com/display/ServerDoc2x/Removing+Security

Luego de hacer esos cambios estamos listos para iniciar el servidor:

```
# cd biserver-ce
```

```
# ./start-pentaho.sh
```

En este punto deberíamos poder abrir la aplicación sin usar credenciales usando la dirección del servidor:

http://localhost:8080/pentaho

```
Nota: Si fuesen necesarias, las credenciales por
defecto son
usuario: joe,
contraseña: password

Los errores del sistema son registrados en:
Log de Pentaho:  biserver-ce/tomcat/logs/pentaho.log
Log de Servidor Tomcat: biserver-ce/tomcat/logs/
catalina.out
```

A continuación, conectaremos Pentaho a la base de datos del SIIG usando la consola de administración. La consola de administración no incluye soporte para Postgres. El primer paso es copiar el manejador de Postgres:

```
# cp biserver-ce/tomcat/lib/postgresql-9.1-902.jdbc4.jar administration-
console/jdbc/
```

Luego arrancamos la consola de administración usando el script dentro de la carpeta administration-console:

```
# ./start-pac.sh
```

En este punto ya podemos conectarnos a:

http://localhost:8099/

```
Nota: Si fuese necesario las credenciales por defecto
son:
usuario: admin
contraseña: password
```

Una vez dentro de la consola, podemos crear nueva conexión de bases de datos, asegurándonos de usar estos valores.

nombre: conexion_bd_pentaho

driver: org.postgres.Driver

URL: jdbc:postgresql://localhost:5432/database_name

Los parámetros conexion_bd_pentaho y database_name deben de ser los que se establecieron anteriormente al ejecutar el comando composer install.

Asegúrese de probar la conexión usando el botón "Test/Probar" al pie de esta misma ventana. Finalmente guarde sus cambios y detener la consola de administración:

./stop-pac.sh

2.6.2 Configuración de Mondrian

Ahora que Pentaho ya puede conectarse a nuestra base datos, procederemos a configurar el servicio de Mondrian para la gestión de cubos OLAP. Para esto es necesario:

- Crear un archivo para definir nuestro cubo OLAP. Mondrian conoce estos archivos como ‘schemas’ y puede ser creado usando la siguiente plantilla:

https://github.com/erodriguez-minsal/SIIG/wiki/PlantillaIndicadorOLAP

Alternativamente el mismo archivo puede ser editado/creado usando la aplicación Mondrian Schema Workbench disponible aquí:

http://sourceforge.net/projects/mondrian/files/schema%20workbench/

Finalmente guardamos el archivo de la siguiente forma:

Biserver-ce/pentaho-solutions/admin/resources/metadata/
NOMBRE_CUBO.mondrian.xml

- Y Agregamos el nuevo cubo al listado de cubos de Mondrian. Este listado esta descrito en el archivo:

biserver-ce/pentaho-solutions/system/olap/datasources.xml

En este archivo cada cubo esta definido de la siguiente forma:

```
<Catalog name="NOMBRE_CUBO">

<DataSourceInfo>Provider=mondrian;DataSource=conexion_bd_pentaho
```

```
    <Definition>solution:admin/resources/metadata/
NOMBRE_CUBO.mondrian.xml</Definition>
</Catalog>
```

Este listado puede incluir varios cubos, por cada cubo que se agregue al sistema habrá que crear su archivo/esquema correspondiente y agregarlo a este listado. Alternativamente, la aplicación Mondrian Workbench, puede generar el esquema del cubo y luego publicarlo/agregarlo a este listado.

2.6.3 Instalar SAIKU

Para poder manipular visualmente los cubos que hemos creado usaremos SAIKU. Esta es una aplicación que permite hacer consultas al cubo y mostrar resultados usando peticiones REST y AJAX. SAIKU procesa la respuesta devuelta por Pentaho en formato JSON para generar representaciones visuales de los datos. Para saber mas cerca de SAIKU puede visitar:

http://analytical-labs.com/downloads.php

Para instalar SAIKU debemos primero instalar las librerías de CTOOLS. Estas librerías se pueden instalar automáticamente usando este Script:

https://github.com/pmalves/ctools-installer

Guardamos este archivo como: biserver-ce/instalar_ctools.sh y le damos permisos de ejecución:

```
#chmod +x instalar_ctools.sh
```

Y ejecutamos el script:

```
# ./instalar_ctools.sh –s pentaho-solutions/
```

El Script preguntará si queremos instalar todas las librerías, incluyendo el paquete SAIKU, respondemos que si a todo.

Reiniciar Pentaho:

```
# ./stop-pentaho.sh

# ./start-pentaho.sh
```

2.6.4 Modificar Apache: URL del SIIG apuntando a SAIKU

Para enmascarar la URL de Pentaho debemos activar el proxy de Apache para esto debemos activar un par de módulos de Apache:

```
#a2enmod proxy proxy_http
```

Luego editamos la sección VirtualHost dentro de /etc/apache2/sites-enabled/000-default:

```
<Location /admin/minsal/indicadores/saiku/>
    ProxyPass http://localhost:8080/pentaho/content/saiku/
    ProxyPassReverse http://localhost:8080/pentaho/content/saiku/
    SetEnv proxy-chain-auth
</Location>
```

En este punto ya tenemos SAIKU disponible como una URL del SIIG en:

http://localhost/admin/minsal/indicadores/saiku/

El servidor OLAP/Mondrian puede ser consultado a través de SAIKU usando su API HTTP/REST. Esta API permite obtener información sobre los cubos existentes en el servidor OLAP así como efectuar consultas, los resultados son devueltos en formato JSON. La documentación de la API puede ser consultada en:

http://dev.analytical-labs.com/saiku/serverdocs/

2.6.5 4.5 Generación de Reportes

Los reportes son generados usando el servidor de análisis Pentaho. Cada indicador puede tener un reporte individual que incluye menús de selección, gráficos, texto y tablas de datos según se requiera. Debido a que los requerimientos de análisis y presentación varían entre indicadores, estos reportes deben ser diseñados y publicados manualmente por el administrador del sistema.

Los reportes existentes están disponibles desde el menú del SIIG en administración->Indicadores->Ficha Técnica (http://SIIG/admin/minsal/indicadores/fichatecnica/list).

El listado de Ficha Técnica incluye un botón Mostrar Reporte que carga el reporte para el indicador correspondiente usando los datos disponibles mas recientes.

Todos los indicadores en el listado tienen el botón 'Mostrar Reporte', sin embargo solo los reportes creados y publicados por el administrador del sistema están disponibles.

El proceso para crear y publicar reportes incluye:

1. **Fijar clave para la publicación** de contenidos al editar el archivo:

   pentaho-solutions/system/publisher_config.xml

para modificar esta linea

```
<publisher-password>NUEVA_CLAVE</publisher-password>
```

1. **Crear Reporte** para el Indicador deseado usando la aplicación de Reportes de Pentaho. Esta aplicación debe ser instalada en la terminal del administrador del sistema y puede ser descargada en:

http://reporting.pentaho.com/report_designer.php

Una guía completa sobre el diseño de reportes usando esta aplicación esta disponible en

http://wiki.pentaho.com/display/Reporting/01.+Creating+Your+First+Report

Los reportes pueden ser creados y editados de forma local, y una vez terminados pueden ser publicados en el servidor de Pentaho. Al publicar un reporte, este inmediatamente esta disponible para el sistema SIIG. Si es necesario modificar un reporte existente, la modificación debe hacerse en forma local y luego publicar la nueva versión del reporte.

1. **Publicar reporte**. Al seleccionar la opción publicar, aparece un ventana que nos pide elegir una carpeta en la cual se publicará el reporte. Para que pueda ser encontrada por el SIIG, asegúrese de guardar todos sus reportes en una carpeta llamada 'reportes'. Si esta carpeta no existe puede crearla al momento de guardar su primer reporte usando el botón 'Nueva Carpeta'.

Una vez dentro de la carpeta 'reportes' deberá asignar un nombre de archivo (Ejem: indicador15.prpt), un titulo e ingresar la clave de publicación que se fijo en el primer paso.

```
NOTA: El SIIG  esta configurado para leer reportes
tales como: reportes/indicadorX.prpt
```

```
Por esto, si el reporte es publicado en una carpeta
diferente o si el nombre
asignado al archivo es diferente, no podrá ser leído
por el SIIG.
```

2.7 Instalación de librería wkhtmltopdf

wkhtmltopdf (http://code.google.com/p/wkhtmltopdf/) Es una utilidad de línea de comando para convertir html a pdf

1. Descargar wkhtmltopdf desde http://code.google.com/p/wkhtmltopdf/downloads/list elegir la versión adecuada al sistema operativo
2. Descomprimir. Ej.: tar xjf wkhtmltopdf-0.11.0_rc1-static-amd64.tar.bz2
3. Mover y renombrar el archivo: mv wkhtmltopdf-amd64 /usr/bin/wkhtmltopdf
4. Dar permisos de ejecución: chmod +x /usr/bin/wkhtmltopdf

2.8 OPCIONAL: Validación de Usuarios desde directorios LDAP

Si fuese necesario validar usuarios contra un directorio LDAP, se deben seguir los pasos descritos en esta sección. Si un usuario aun no esta creado dentro del sistema, se hará una búsqueda en el directorio LDAP especificado en el archivo app/config/config.yml. A continuación se muestran las lineas relevantes para especificar que directorio usar:

```
#Fr3d_LDAP
fr3d_ldap:
    driver:
        host:   10.10.20.2 # IP del Servidor LDAP
institucional
        port:   389    # Opcional
    user:
        baseDn: ou=people,dc=salud,dc=gob,dc=sv #
contenedor de usuarios
        filter: (objectClass=organizationalPerson) #
esquema comun para todos los usuarios del directorio
```

2.8.1 Cargar la aplicación

En este punto estamos listos para cargar la aplicación desde:

http://siig.localhost

Esta página se ha dejado vacía a propósito

Capítulo 3

Modelo De Datos

3.0.1 Esquema general de la Aplicacion

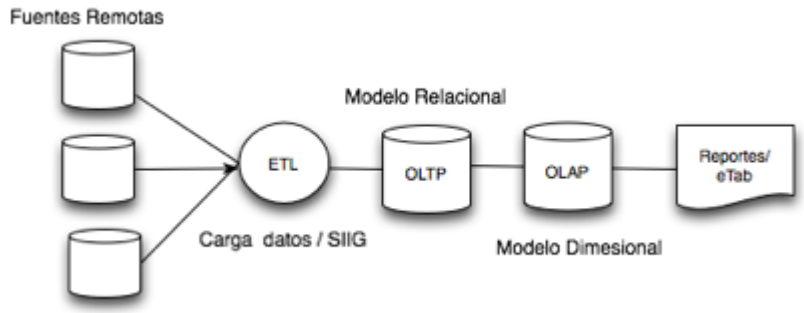

Figura 3.1 Esquema de la aplicación

Los datos que maneja el sistema provienen de distintas fuentes y son de una naturaleza tal que es necesario utilizar el modelo de base datos sin esquema/ genérico EAV. Las tabla EAV (Fila_origen_dato) y demás tablas auxiliares son parte del almacenamiento de datos transaccional (OLTP) de la aplicación. Esto facilita el manejo de datos de cualquier indicador sin importar sus propiedades. Los cubos de análisis multidimensional (OLAP) son generados usando estos valores genéricos y estan descritos en la seccion de Gesition de Cubos OLAP. Las tablas de los cubos OLAP usan un esquema de estrella mientras que las tablas del almacenamiento OLTP usan un modelo relacional. El Siguiente Diccionario de Datos y Diagrama ER describen la estructura del almacenamiento transaccional (OLTP) de la Aplicacion.

3.0.2 Diagrama Entidad Relacion

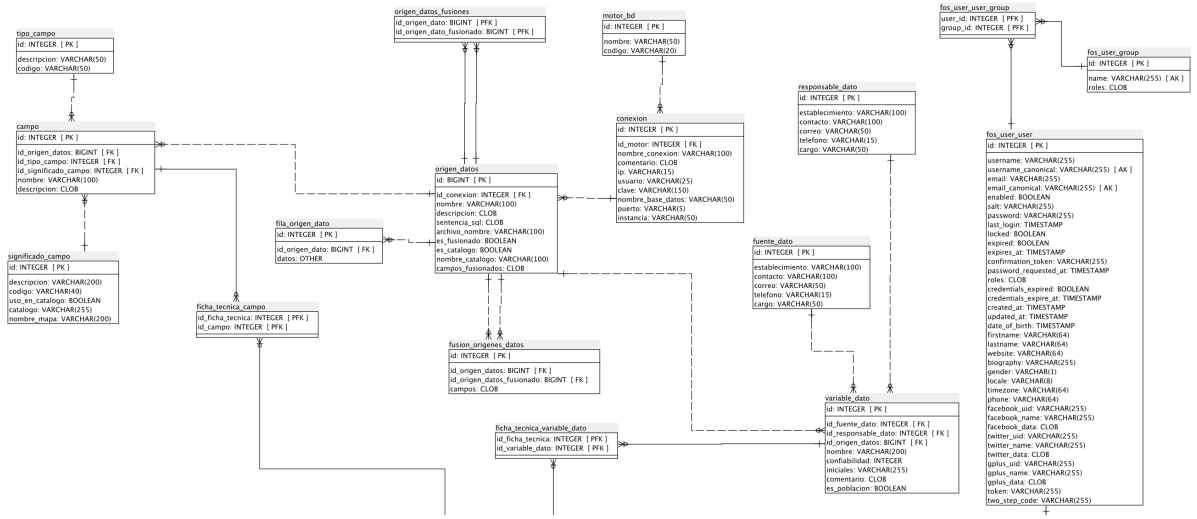

Figura 3.2 Diagrama ER1

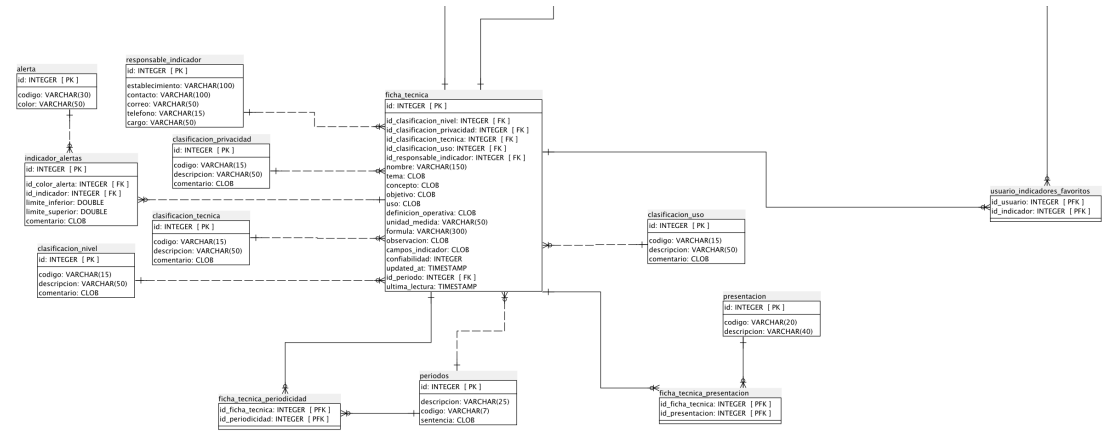

Figura 3.3 Diagrama ER2

3.0.3 Diccionario de Datos

3.0.3.1 Lista de tablas

1. alerta ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
codigo	codigo	VARCHAR(30)		NOT NULL
color	color	VARCHAR(50)		NOT NULL

Esta tabla es usada por:

- indicador_alertas (página 41) hace referencia la campo (id)

2. campo ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
id_origen_datos (FK (página 42))	id_origen_datos	BIGINT		NOT NULL
id_tipo_campo (FK (página 45))	id_tipo_campo	INTEGER		NOT NULL
id_significado_campo (FK (página 45))	id_significado_campo	INTEGER		NOT NULL
nombre	nombre	VARCHAR(100)		NOT NULL
descripcion	descripcion	CLOB		

Esta tabla depende de:

- origen_datos (página 42) por medio de (id_origen_datos)
- tipo_campo (página 45) por medio de (id_tipo_campo)
- significado_campo (página 45) por medio de (id_significado_campo)

Esta tabla es usada por:

- ficha_tecnica_campo (página 36) hace referencia la campo (id)

3. clasificacion_nivel ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
codigo	codigo	VARCHAR(15)		NOT NULL
descripcion	descripcion	VARCHAR(50)		NOT NULL
comentario	comentario	CLOB		

Esta tabla es usada por:

- ficha_tecnica (página 34) hace referencia la campo (id)

4. clasificacion_privacidad ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
codigo	codigo	VARCHAR(15)		NOT NULL
descripcion	descripcion	VARCHAR(50)		NOT NULL
comentario	comentario	CLOB		

Esta tabla es usada por:

- ficha_tecnica (página 34) hace referencia la campo (id)

5. clasificacion_tecnica ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
codigo	codigo	VARCHAR(15)		NOT NULL
descripcion	descripcion	VARCHAR(50)		NOT NULL
comentario	comentario	CLOB		

Esta tabla es usada por:

- ficha_tecnica (página 34) hace referencia la campo (id)

6. clasificacion_uso ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
codigo	codigo	VARCHAR(15)		NOT NULL
descripcion	descripcion	VARCHAR(50)		NOT NULL

comentario	comentario	CLOB		

Esta tabla es usada por:

- ficha_tecnica (página 34) hace referencia la campo (id)

7. conexion ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
id_motor (FK (página 42))	id_motor	INTEGER		NOT NULL
nombre_conexion	nombre_conexion	VARCHAR(100)		NOT NULL
comentario	comentario	CLOB		
ip	ip	VARCHAR(15)		NOT NULL
usuario	usuario	VARCHAR(25)		NOT NULL
clave	clave	VARCHAR(150)		NOT NULL
nombre_base_datos	nombre_base_datos	VARCHAR(50)		NOT NULL
puerto	puerto	VARCHAR(5)		
instancia	instancia	VARCHAR(50)		

Esta tabla depende de:

- motor_bd (página 42) por medio de (id_motor)

Esta tabla es usada por:

- origen_datos (página 42) hace referencia la campo (id)

8. ficha_tecnica ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL

id_clasificacion_nivel (FK (página 32))	id_clasificacion_nivel	INTEGER		NOT NU
id_clasificacion_privacidad (FK (página 33))	id_clasificacion_privacidad	INTEGER		NOT NU
id_clasificacion_tecnica (FK (página 33))	id_clasificacion_tecnica	INTEGER		NOT NU
id_clasificacion_uso (FK (página 33))	id_clasificacion_uso	INTEGER		NOT NU
id_responsable_indicador (FK (página 45))	id_responsable_indicador	INTEGER		NOT NU
nombre	nombre	VARCHAR(150)		NOT NU
tema	tema	CLOB		NOT NU
concepto	concepto	CLOB		
objetivo	objetivo	CLOB		
uso	uso	CLOB		
definicion_operativa	definicion_operativa	CLOB		
unidad_medida	unidad_medida	VARCHAR(50)		NOT NU
formula	formula	VARCHAR(300)		NOT NU
observacion	observacion	CLOB		
campos_indicador	campos_indicador	CLOB		
confiabilidad	confiabilidad	INTEGER		
updated_at	updated_at	TIMESTAMP		
id_periodo (FK (página 43))	id_periodo	INTEGER		NOT NU
ultima_lectura	ultima_lectura	TIMESTAMP		

Esta tabla depende de:

- periodos (página 43) por medio de (id_periodo)

- clasificacion_privacidad (página 33) por medio de (id_clasificacion_privacidad)
- clasificacion_tecnica (página 33) por medio de (id_clasificacion_tecnica)
- clasificacion_uso (página 33) por medio de (id_clasificacion_uso)
- clasificacion_nivel (página 32) por medio de (id_clasificacion_nivel)
- responsable_indicador (página 45) por medio de (id_responsable_indicador)

Esta tabla es usada por:

- ficha_tecnica_periodicidad (página 36) hace referencia la campo (id)
- ficha_tecnica_campo (página 36) hace referencia la campo (id)
- ficha_tecnica_presentacion (página 37) hace referencia la campo (id)
- indicador_alertas (página 41) hace referencia la campo (id)
- ficha_tecnica_variable_dato (página 37) hace referencia la campo (id)
- usuario_indicadores_favoritos (página 46) hace referencia la campo (id)

9. ficha_tecnica_campo ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id_ficha_tecnica (PK) (FK (página 34))	id_ficha_tecnica	INTEGER	PK	NOT NULL
id_campo (PK) (FK (página 32))	id_campo	INTEGER	PK	NOT NULL

Esta tabla depende de:

- campo (página 32) por medio de (id_campo)
- ficha_tecnica (página 34) por medio de (id_ficha_tecnica)

10. ficha_tecnica_periodicidad ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable

id_ficha_tecnica (PK) (FK (página 34))	id_ficha_tecnica	INTEGER	PK	NOT NULL
id_periodicidad (PK) (FK (página 43))	id_periodicidad	INTEGER	PK	NOT NULL

Esta tabla depende de:

- periodos (página 43) por medio de (id_periodicidad)
- ficha_tecnica (página 34) por medio de (id_ficha_tecnica)

11. ficha_tecnica_presentacion ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id_ficha_tecnica (PK) (FK (página 34))	id_ficha_tecnica	INTEGER	PK	NOT NULL
id_presentacion (PK) (FK (página 44))	id_presentacion	INTEGER	PK	NOT NULL

Esta tabla depende de:

- ficha_tecnica (página 34) por medio de (id_ficha_tecnica)
- presentacion (página 44) por medio de (id_presentacion)

12. ficha_tecnica_variable_dato ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id_ficha_tecnica (PK) (FK (página 34))	id_ficha_tecnica	INTEGER	PK	NOT NULL
id_variable_dato (PK) (FK (página 46))	id_variable_dato	INTEGER	PK	NOT NULL

Esta tabla depende de:

- ficha_tecnica (página 34) por medio de (id_ficha_tecnica)
- variable_dato (página 46) por medio de (id_variable_dato)

13. fila_origen_dato ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
id_origen_dato (FK (página 42))	id_origen_dato	BIGINT		NOT NULL
datos	datos	[1111]		

Esta tabla depende de:

- origen_datos (página 42) por medio de (id_origen_dato)

14. fos_user_group ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
name	name	VARCHAR(255)		NOT NULL
roles	roles	CLOB		NOT NULL

(DC2Tipo:array)

Esta tabla es usada por:

- fos_user_user_group (página 40) hace referencia la campo (id)

15. fos_user_user ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
username	username	VARCHAR(255)		NOT NULL
username_canonical	username_canonical	VARCHAR(255)		NOT NULL
email	email	VARCHAR(255)		NOT NULL
email_canonical	email_canonical	VARCHAR(255)		NOT NULL

enabled	enabled	BOOLEAN		NOT NULL
salt	salt	VARCHAR(255)		NOT NULL
password	password	VARCHAR(255)		NOT NULL
last_login	last_login	TIMESTAMP		
locked	locked	BOOLEAN		NOT NULL
expired	expired	BOOLEAN		NOT NULL
expires_at	expires_at	TIMESTAMP		
confirmation_token	confirmation_token	VARCHAR(255)		
password_requested_at	password_requested_at	TIMESTAMP		
roles	roles	CLOB		NOT NULL
credentials_expired	credentials_expired	BOOLEAN		NOT NULL
credentials_expire_at	credentials_expire_at	TIMESTAMP		
created_at	created_at	TIMESTAMP		NOT NULL
updated_at	updated_at	TIMESTAMP		NOT NULL
date_of_birth	date_of_birth	TIMESTAMP		
firstname	firstname	VARCHAR(64)		
lastname	lastname	VARCHAR(64)		
website	website	VARCHAR(64)		
biography	biography	VARCHAR(255)		
gender	gender	VARCHAR(1)		
locale	locale	VARCHAR(8)		
timezone	timezone	VARCHAR(64)		
phone	phone	VARCHAR(64)		
facebook_uid	facebook_uid	VARCHAR(255)		
facebook_name	facebook_name	VARCHAR(255)		
facebook_data	facebook_data	CLOB		

twitter_uid	twitter_uid	VARCHAR(255)		
twitter_name	twitter_name	VARCHAR(255)		
twitter_data	twitter_data	CLOB		
gplus_uid	gplus_uid	VARCHAR(255)		
gplus_name	gplus_name	VARCHAR(255)		
gplus_data	gplus_data	CLOB		
token	token	VARCHAR(255)		
two_step_code	two_step_code	VARCHAR(255)		

Esta tabla es usada por:

- fos_user_user_group (página 40) hace referencia la campo (id)
- usuario_indicadores_favoritos (página 46) hace referencia la campo (id)

16. fos_user_user_group ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
user_id (PK) (FK (página 38))	user_id	INTEGER	PK	NOT NULL
group_id (PK) (FK (página 38))	group_id	INTEGER	PK	NOT NULL

Esta tabla depende de:

- fos_user_group (página 38) por medio de (group_id)
- fos_user_user (página 38) por medio de (user_id)

17. fuente_dato ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
establecimiento	establecimiento	VARCHAR(100)		NOT NULL

contacto	contacto	VARCHAR(100)		NOT NULL
correo	correo	VARCHAR(50)		NOT NULL
telefono	telefono	VARCHAR(15)		NOT NULL
cargo	cargo	VARCHAR(50)		NOT NULL

Esta tabla es usada por:

- variable_dato (página 46) hace referencia la campo (id)

18. fusion_origenes_datos ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
id_origen_datos (FK (página 42))	id_origen_datos	BIGINT		NOT NULL
id_origen_datos_fusionado (FK (página 42))	id_origen_datos_fusionado	BIGINT		NOT NULL
campos	campos	CLOB		

Esta tabla depende de:

- origen_datos (página 42) por medio de (id_origen_datos)
- origen_datos (página 42) por medio de (id_origen_datos_fusionado)

19. indicador_alertas ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
id_color_alerta (FK (página 31))	id_color_alerta	INTEGER		NOT NULL
id_indicador (FK (página 34))	id_indicador	INTEGER		NOT NULL
limite_inferior	limite_inferior	DOUBLE		NOT NULL

limite_superior	limite_superior	DOUBLE		NOT NULL
comentario	comentario	CLOB		

Esta tabla depende de:

- alerta (página 31) por medio de (id_color_alerta)
- ficha_tecnica (página 34) por medio de (id_indicador)

20. motor_bd ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
nombre	nombre	VARCHAR(50)		NOT NULL
codigo	codigo	VARCHAR(20)		

Esta tabla es usada por:

- conexion (página 34) hace referencia la campo (id)

21. origen_datos ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	BIGINT	PK	NOT NULL
id_conexion (FK (página 34))	id_conexion	INTEGER		NOT NULL
nombre	nombre	VARCHAR(100)		NOT NULL
descripcion	descripcion	CLOB		
sentencia_sql	sentencia_sql	CLOB		
archivo_nombre	archivo_nombre	VARCHAR(100)		
es_fusionado	es_fusionado	BOOLEAN		
es_catalogo	es_catalogo	BOOLEAN		
nombre_catalogo	nombre_catalogo	VARCHAR(100)		

campos_fusionados	campos_fusionados	CLOB		

Esta tabla depende de:

- conexion (página 34) por medio de (id_conexion)

Esta tabla es usada por:

- campo (página 32) hace referencia la campo (id)
- origen_datos_fusiones (página 43) hace referencia la campo (id)
- origen_datos_fusiones (página 43) hace referencia la campo (id)
- variable_dato (página 46) hace referencia la campo (id)
- fila_origen_dato (página 38) hace referencia la campo (id)
- fusion_origenes_datos (página 41) hace referencia la campo (id)
- fusion_origenes_datos (página 41) hace referencia la campo (id)

22. origen_datos_fusiones ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id_origen_dato (PK) (FK (página 42))	id_origen_dato	BIGINT	PK	NOT NULL
id_origen_dato_fusionado (PK) (FK (página 42))	id_origen_dato_fusionado	BIGINT	PK	NOT NULL

Esta tabla depende de:

- origen_datos (página 42) por medio de (id_origen_dato_fusionado)
- origen_datos (página 42) por medio de (id_origen_dato)

23. periodos ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
descripcion	descripcion	VARCHAR(25)		NOT NULL

codigo	codigo	VARCHAR(7)		NOT NULL
sentencia	sentencia	CLOB		

Esta tabla es usada por:

- ficha_tecnica_periodicidad (página 36) hace referencia la campo (id)
- ficha_tecnica (página 34) hace referencia la campo (id)

24. presentacion ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
codigo	codigo	VARCHAR(20)		NOT NULL
descripcion	descripcion	VARCHAR(40)		NOT NULL

Esta tabla es usada por:

- ficha_tecnica_presentacion (página 37) hace referencia la campo (id)

25. responsable_dato ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
establecimiento	establecimiento	VARCHAR(100)		NOT NULL
contacto	contacto	VARCHAR(100)		NOT NULL
correo	correo	VARCHAR(50)		NOT NULL
telefono	telefono	VARCHAR(15)		NOT NULL
cargo	cargo	VARCHAR(50)		NOT NULL

Esta tabla es usada por:

- variable_dato (página 46) hace referencia la campo (id)

26. responsable_indicador ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
establecimiento	establecimiento	VARCHAR(100)		NOT NULL
contacto	contacto	VARCHAR(100)		NOT NULL
correo	correo	VARCHAR(50)		NOT NULL
telefono	telefono	VARCHAR(15)		NOT NULL
cargo	cargo	VARCHAR(50)		NOT NULL

Esta tabla es usada por:

- ficha_tecnica (página 34) hace referencia la campo (id)

27. significado_campo ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
descripcion	descripcion	VARCHAR(200)		NOT NULL
codigo	codigo	VARCHAR(40)		NOT NULL
uso_en_catalogo	uso_en_catalogo	BOOLEAN		
catalogo	catalogo	VARCHAR(255)		
nombre_mapa	nombre_mapa	VARCHAR(200)		

Esta tabla es usada por:

- campo (página 32) hace referencia la campo (id)

28. tipo_campo ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL

descripcion	descripcion	VARCHAR(50)		
codigo	codigo	VARCHAR(50)		NOT NULL

Esta tabla es usada por:

- campo (página 32) hace referencia la campo (id)

29. usuario_indicadores_favoritos ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id_usuario (PK) (FK (página 38))	id_usuario	INTEGER	PK	NOT NULL
id_indicador (PK) (FK (página 34))	id_indicador	INTEGER	PK	NOT NULL

Esta tabla depende de:

- ficha_tecnica (página 34) por medio de (id_indicador)
- fos_user_user (página 38) por medio de (id_usuario)

30. variable_dato ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id (PK)	id	INTEGER	PK	NOT NULL
id_fuente_dato (FK (página 40))	id_fuente_dato	INTEGER		NOT NULL
id_responsable_dato (FK (página 44))	id_responsable_dato	INTEGER		NOT NULL
id_origen_datos (FK (página 42))	id_origen_datos	BIGINT		NOT NULL
nombre	nombre	VARCHAR(200)		NOT NULL
confiabilidad	confiabilidad	INTEGER		
iniciales	iniciales	VARCHAR(255)		NOT NULL

comentario	comentario	CLOB		
es_poblacion	es_poblacion	BOOLEAN		

Esta tabla depende de:

- origen_datos (página 42) por medio de (id_origen_datos)
- fuente_dato (página 40) por medio de (id_fuente_dato)
- responsable_dato (página 44) por medio de (id_responsable_dato)

Esta tabla es usada por:

- ficha_tecnica_variable_dato (página 37) hace referencia la campo (id)

Esta página se ha dejado vacía a propósito

Capítulo 4

Gestión de Cubos OLAP

4.0.1 Introducción

La cantidad de reportes que pueden generarse depende de los catálogos que se usen en cada indicador. Al interior de una estructura para reportes (cubo OLAP), cada catalogo se convierte en dimensión. Así por ejemplo el catalogo municipio se convierte en una dimensión dentro de la estructura de reportes y posibilita hacer búsquedas usando cualquier campo que exista dentro de este catálogo. Por ejemplo: Si el catalogo municipio tiene los campos municipio, departamento y región esto nos permitiría hacer las siguientes consultas: numero de casos por municipio, número de casos por departamento, número de casos por región.

La ventaja principal de usar un gestor de cubos OLAP es aislar la lógica de las búsquedas para analizar los datos. De esta forma el sistema se enfoca en presentar al usuario la mayor cantidad de información de forma flexible sin preocuparse de la lógica para obtener los datos.

4.0.2 Indicadores y Cubos OLAP

El sistema cuenta con un servidor de gestión de cubos OLAP que se conecta directamente a la base de datos de Indicadores. La definición de cubos esta descrita en el archivo biserver-ce/pentaho-solutions/system/olap/datasources.xml, los contenidos de este archivo se muestran a continuación:

```
<?xml version="1.0" encoding="UTF-8"?>

<DataSources>
```

```
  <DataSource>

<DataSourceName>Provider=Mondrian;DataSource=Pentaho</DataSourceName
    <DataSourceDescription>Pentaho BI Platform
Datasources</DataSourceDescription>
    <URL>http://localhost:8080/pentaho/
Xmla?userid=joe&amp;password=password</URL>
    <DataSourceInfo>Provider=mondrian</DataSourceInfo>
    <ProviderName>PentahoXMLA</ProviderName>
    <ProviderType>MDP</ProviderType>

<AuthenticationMode>Unauthenticated</AuthenticationMode>
    <Catalogs>
      <Catalog name="Indicador 23">

<DataSourceInfo>Provider=mondrian;DataSource=Minsal</DataSourceInfo>
        <Definition>solution:/admin/resources/metadata/
indicador23.mondrian.xml</Definition>
      </Catalog>
      <Catalog name="Indicador 24">

<DataSourceInfo>Provider=mondrian;DataSource=Minsal</DataSourceInfo>
        <Definition>solution:/admin/resources/metadata/
indicador24.mondrian.xml</Definition>
      </Catalog>
      <Catalog name="Indicador 25">

<DataSourceInfo>Provider=mondrian;DataSource=Minsal</DataSourceInfo>
        <Definition>solution:/admin/resources/metadata/
indicador25.mondrian.xml</Definition>
      </Catalog>
      <Catalog name="indicador 7">

<DataSourceInfo>Provider=mondrian;DataSource=Minsal</DataSourceInfo>
        <Definition>solution:/admin/resources/metadata/
indicador7.mondrian.xml</Definition>
      </Catalog>
    </Catalogs>
  </DataSource>
</DataSources>
```

Como puede verse en este código cada indicador es un catalogo/cubo cuya descripción esta contenida en otro archivo XML. Para facilitar la creación de nuevos cubos a continuación se muestra el código base de un nuevo indicador, esta plantilla esta disponible en https://github.com/erodriguez-minsal/SIIG/wiki/PlantillaIndicadorOLAP

Para crear un indicador nuevo se debe:

1. Cambia el nombre del archivo usando el numero del indicador correspondiente ej: indicador5.mondrian.xml

2. Modificar los valores de este archivo (comentario N.1 - N.5)

3. Copiar este archivo al servidor en: /home/siig/biserver-ce/pentaho-solutions/admin/resources/metadata

4. Agregar el indicador como un nuevo catalogo: el Archivo de Fuentes de datos /home/siig/biserver-ce/pentaho-solutions/system/olap/datasources.xml

Se deben agregar estas lineas:

```
<Catalog name="Indicador 5">

<DataSourceInfo>Provider=mondrian;DataSource=Minsal</DataSourceI
    <Definition>solution:/admin/resources/metadata/
indicador5.mondrian.xml</Definition>
</Catalog>
```

Refrescar la aplicación.

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- N.1 - Asignar el numero del indicador segun la
tabla Ficha_tecnica-->
<Schema name="Indicador X">

<!--Definicion de dimensiones, pueden estar definidas
aun si no se utilizan en este indicador-->

<!-- Definicion de la dimension departamento-->
<Dimension visible="true" highCardinality="false"
name="Departamento">
```

```
    <Hierarchy name="Departamento" visible="true"
hasAll="true" primaryKey="id">
      <Table name="ctl_departamento" schema="public">
      </Table>
      <Level name="Departamento" visible="true"
column="descripcion" type="String"
uniqueMembers="false" levelType="Regular"
hideMemberIf="Never">
      </Level>
    </Hierarchy>
    <Hierarchy name="Digestyc" visible="true"
hasAll="true" primaryKey="id">
      <Table name="ctl_departamento" schema="public">
      </Table>
      <Level name="Digestyc" visible="true"
column="digestyc" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
      </Level>
    </Hierarchy>
    <Hierarchy name="Gis" visible="true" hasAll="true"
primaryKey="id">
      <Table name="ctl_departamento" schema="public">
      </Table>
      <Level name="Gis" visible="true" column="gis"
type="Numeric" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
      </Level>
    </Hierarchy>
  </Dimension>

  <!-- Definicion de la dimension Municipio-->
  <Dimension visible="true" highCardinality="false"
name="Municipio">
    <Hierarchy name="Municipio" visible="true"
hasAll="true" primaryKey="id">
      <Table name="ctl_municipio" schema="public">
      </Table>
      <Level name="Municipio" visible="true"
column="descripcion" type="String"
uniqueMembers="false" levelType="Regular"
hideMemberIf="Never">
```

```
      </Level>
    </Hierarchy>
  </Dimension>


  <!-- Definicion de la dimension Region-->
  <Dimension visible="true" highCardinality="false"
name="Region">
    <Hierarchy name="Region" visible="true"
hasAll="true" primaryKey="id">
      <Table name="ctl_regiones" schema="public">
      </Table>
      <Level name="Region" visible="true"
column="descripcion" type="String"
uniqueMembers="false" levelType="Regular"
hideMemberIf="Never">
      </Level>
    </Hierarchy>
  </Dimension>

  <!-- Definicion de la dimension Area-->
  <Dimension visible="true" highCardinality="false"
name="Area">
    <Hierarchy name="Area" visible="true" hasAll="true"
primaryKey="inicial">
      <Table name="ctl_area" schema="public">
      </Table>
      <Level name="Area" visible="true"
column="descripcion" type="String"
uniqueMembers="false" levelType="Regular"
hideMemberIf="Never">
      </Level>
    </Hierarchy>
  </Dimension>

 <!-- Definicion de la dimension Genero-->
   <Dimension visible="true" highCardinality="false"
name="Genero">
    <Hierarchy name="Genero" visible="true"
hasAll="true" primaryKey="inicial">
      <Table name="ctl_sexo" schema="public">
```

```
      </Table>
      <Level name="Genero" visible="true"
column="descripcion" type="String"
uniqueMembers="false" levelType="Regular"
hideMemberIf="Never">
      </Level>
    </Hierarchy>
  </Dimension>

 <!-- Definicion de la dimension Sibasi-->
   <Dimension visible="true" highCardinality="false"
name="Sibasi">
    <Hierarchy name="Sibasi" visible="true"
hasAll="true" primaryKey="id">
      <Table name="ctl_sibasi" schema="public">
      </Table>
      <Level name="Sibasi" visible="true"
column="descripcion" type="String"
uniqueMembers="false" levelType="Regular"
hideMemberIf="Never">
      </Level>
    </Hierarchy>
  </Dimension>

  <!-- Definicion de la dimension Tiempo-->
  <Dimension visible="true" highCardinality="false"
name="Tiempo">
    <Hierarchy name="Anio" visible="true" hasAll="true"
primaryKey="fecha">
      <Table name="ctl_tiempo" schema="public">
      </Table>
      <Level name="Anio" visible="true" column="anio"
type="Numeric" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
      </Level>
    </Hierarchy>
    <Hierarchy name="Mes" visible="true" hasAll="true"
primaryKey="fecha">
      <Table name="ctl_tiempo" schema="public">
      </Table>
      <Level name="Mes" visible="true" column="mes"
```

```
type="Numeric" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
      </Level>
      <Level name="Mesanio" visible="true"
column="mesanio" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
      </Level>
      <Level name="Finmes" visible="true"
column="finmes" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
      </Level>
    </Hierarchy>
    <Hierarchy name="Feriadoelsalvador" visible="true"
hasAll="true" primaryKey="fecha">
      <Table name="ctl_tiempo" schema="public">
      </Table>
      <Level name="Feriadoelsalvador" visible="true"
column="feriadoelsalvador" type="String"
uniqueMembers="false" levelType="Regular"
hideMemberIf="Never">
      </Level>
    </Hierarchy>
    <Hierarchy name="Periodo" visible="true"
hasAll="true" primaryKey="fecha">
      <Table name="ctl_tiempo" schema="public">
      </Table>
      <Level name="Periodo" visible="true"
column="periodo" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
      </Level>
    </Hierarchy>
    <Hierarchy name="Semanaaniocalendario"
visible="true" hasAll="true" primaryKey="fecha">
      <Table name="ctl_tiempo" schema="public">
      </Table>
      <Level name="Semanaaniocalendario" visible="true"
column="semanaaniocalendario" type="String"
uniqueMembers="false" levelType="Regular"
hideMemberIf="Never">
      </Level>
      <Level name="Semanacalendario" visible="true"
```

```
column="semanacalendario" type="Numeric"
uniqueMembers="false" levelType="Regular"
hideMemberIf="Never">
      </Level>
      <Level name="Findesemana" visible="true"
column="findesemana" type="String"
uniqueMembers="false" levelType="Regular"
hideMemberIf="Never">
      </Level>
    </Hierarchy>
    <Hierarchy name="Trimestre" visible="true"
hasAll="true" primaryKey="fecha">
      <Table name="ctl_tiempo" schema="public">
      </Table>
      <Level name="Trimestre" visible="true"
column="trimestre" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
      </Level>
      <Level name="Trimestreanio" visible="true"
column="trimestreanio" type="String"
uniqueMembers="false" levelType="Regular"
hideMemberIf="Never">
      </Level>
    </Hierarchy>
  </Dimension>

  <!--N.2 - Definicion del cubo-->

  <Cube name="Nombre del indicador">
    <Table name="Nombre de la tabla del indicador"
schema="public"/>

    <!--N.3 - Listado de dimensiones disponibles en
este indicador. Para cada dimension el formato a seguir
es:
    DimensionUsage name=Etiqueta
source=DimensionPreDefinida
foreignKey=ColumnaTablaInidcador-->

    <DimensionUsage name="Departamento"
source="Departamento" foreignKey="id_departamento"/>
```

```
    <DimensionUsage name="Municipio" source="Municipio"
foreignKey="id_municipio"/>
    <DimensionUsage name="Region" source="Region"
foreignKey="id_region"/>
    <DimensionUsage source="Tiempo" name="Tiempo"
visible="true" foreignKey="fecha"
highCardinality="false"/>

 <!--N.4 - Definicion de variables del indicador-->
    <Measure name="nnici_p" column="nnici_p"
aggregator="sum" formatString="#"/>
    <Measure name="nnici_simmow" column="nnici_simmow"
aggregator="sum" formatString="#"/>

    <!--N.5 - Definicion de la formula unidad de
medida-->
<CalculatedMember name="Porcentaje"
formula="([Measures].[nnici_p]/[Measures].[nnici_simmow])*
100" dimension="Measures" visible="true"/>
 </Cube>
</Schema>
```

4.0.3 Crear/Actualizar Catalogo Tiempo

Cada indicador/cubo puede utilizar la dimensión tiempo, esta dimensión es un tabla/catalogo especial que es creada por el administrador del sistema usando la función especial crear_ctl_tiempo. A continuación se muestra el código de esta función:

```
CREATE OR REPLACE FUNCTION crear_ctl_tiempo(inicio
integer DEFAULT 2006, anios integer DEFAULT 8)
RETURNS VOID AS $$
DECLARE
dias integer;
myquery text;
BEGIN
dias:=365*$2;

DROP TABLE IF EXISTS ctl_tiempo;

myquery:='CREATE TABLE ctl_tiempo AS SELECT *  from
(SELECT
```

```
            datum AS fecha,
            extract(year FROM datum)::int  AS Anio,
            extract(month FROM datum)::int AS Mes,
            to_char(datum, ''TMMonth'')::character(12)
AS NombreMes,
            extract(day FROM datum)::int AS Dia,
            extract(doy FROM datum)::int AS DiaAnio,
              to_char(datum, ''TMDay'')::character(12)
AS NombreDiaSemana,
            extract(week FROM datum)::int AS
SemanaCalendario,
            to_char(datum, ''dd. mm.
yyyy'')::character(12) AS FechaCorriente,
            ''T'' || to_char(datum, ''Q'')::int  AS
Trimestre,
            to_char(datum, ''yyyy/Q'')::character(6) AS
TrimestreAnio,
            to_char(datum, ''yyyy/mm'')::character(12)
AS MesAnio,
            -- ISO calendar year and week
            to_char(datum, ''iyyy/IW'')::character(8)
AS SemanaAnioCalendario,
            -- Weekend
            CASE WHEN extract(isodow FROM datum) IN (6,
7) THEN ''FinDeSemana'' ELSE ''DiaDeSemana'' END AS
FinDeSemana,
            -- Feriados para El Salvador
        CASE WHEN to_char(datum, ''MMDD'') IN
(''0801'', ''0802'', ''0803'', ''084'')
                        THEN ''Feriado'' ELSE ''Dia
Laboral'' END
                        AS FeriadoElSalvador,
            -- Periodos festivos del calendario
            CASE WHEN to_char(datum, ''MMDD'') BETWEEN
''0701'' AND ''0831'' THEN ''Vacación de Verano''
                WHEN to_char(datum, ''MMDD'') BETWEEN
''1115'' AND ''1225'' THEN ''Temporada Navideña''
                WHEN to_char(datum, ''MMDD'') >
''1223'' OR to_char(datum, ''MMDD'') <= ''1231'' THEN
''Vacación Navideña''
                        ELSE ''Normal'' END
```

```
                        AS Periodo,
            -- Fecha de inicio de fin de semana
            datum + (1 - extract(isodow FROM
datum))::integer AS IncioSemana,
            datum + (7 - extract(isodow FROM
datum))::integer AS FinSemana,
            -- Fecha de inicio de fin de Mes
            datum + (1 - extract(day FROM
datum))::integer AS InicioMes,
            (datum + (1 - extract(day FROM
datum))::integer + ''1 month''::interval)::date - ''1
day''::interval AS FinMes
FROM ( SELECT '''||$1||'-01-01''::DATE + sequence.day
AS datum
            FROM generate_series(0,'||dias||') AS
sequence(day)
            GROUP BY sequence.day
     ) DQ
ORDER BY 1) as foo;';
EXECUTE myquery;

ALTER TABLE ctl_tiempo ADD PRIMARY KEY (fecha);

RAISE NOTICE 'Se creo tabla Tiempo de % anios a partir
de %',anios,inicio;

END;
$$  LANGUAGE plpgsql;
```

Como se puede ver en el código, los intervalos y periodos de tiempo (feriados, etc) que se quieran usar para analizar datos pueden pueden ser configurados al crear esta función. Luego de que se ha agregado esta función, procedemos a crear el catalogo tiempo dentro de la base de datos indicadores:

```
postgres=# select * from crear_ctl_tiempo(2008,5);
```

NOTA: Se creo tabla Tiempo de 5 años a partir de 2008

Esta página se ha dejado vacía a propósito

Lista de figuras