Antonio Ferrer Lopez.

EXPORTANDO A BASE DE DATOS DESDE XML con php

INDICE

# Introducción.

# Análisis de datos.

# Diseño de base de datos.

# Implementación de servidor.

# Creación de la base de datos en el gestor de bases de datos.

# Creación del software mediante php.

# Comprobación de resultados (CONSULTAS A LA BBDD CREADA).

1. **Introducción.**

Se pretende crear una aplicación del lado del servidor mediante lenguaje de programación PHP que extraiga automáticamente los datos de un archivo xml y los inserte correctamente en una base de datos.

Dicho programa debe realizar las siguientes tareas:

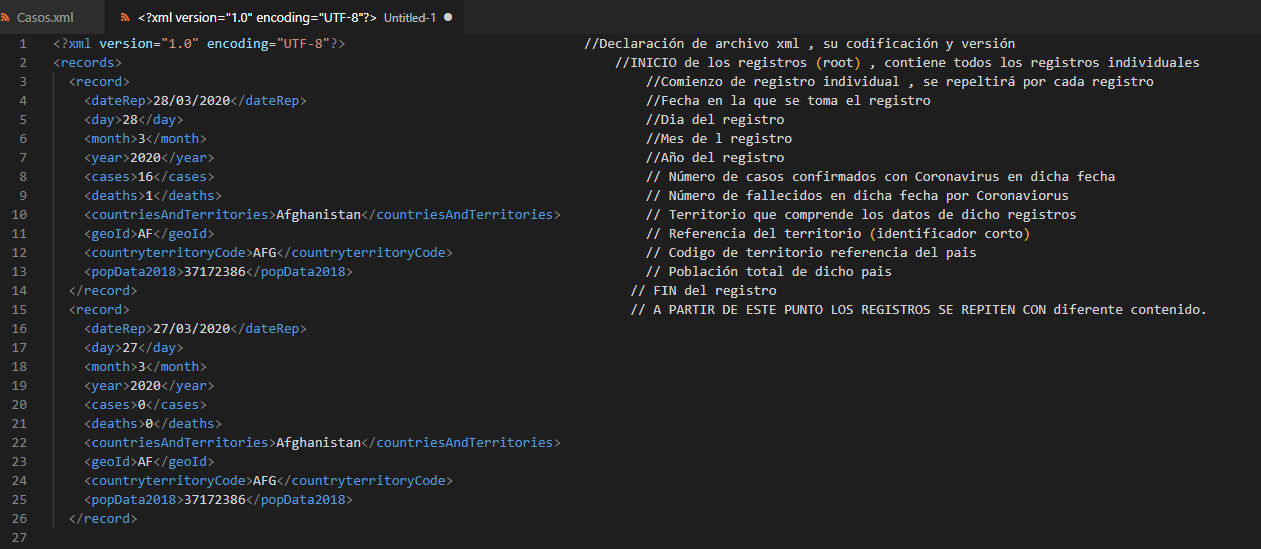
1. Creación de la estructura de la base de datos de acuerdo al diseño previo de esta.
2. Creación de las relaciones entre las tablas de la base de datos.
3. Lectura y serialización del archivo xml proporcionado.
4. Inserción de los datos en la base de datos.

Puesto que se trata de un backend, posteriormente realizaremos una serie de consultas a la base de datos para demostrar el correcto funcionamiento de la misma.

Posteriormente, sería posible diseñar un front-end para el consumo de dicha base de datos, la cual nos permitiría ofrecer dichos datos.

1. **Análisis de datos.**

Se nos proporciona un archivo de datos en formato XML compuesto por un gran número de registros, a continuación, mostramos la estructura de dicho archivo.



Estos datos, expresan los casos positivos, fallecimientos y la población total en varios países del mundo a la largo de unas fechas determinadas.

Dichos datos nos permitirían obtener estadísticas sobre la evolución del virus a nivel global.

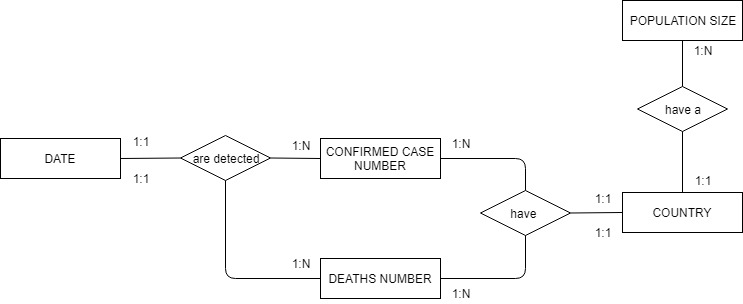
1. **Diseño de la base de datos.**

Una vez comprendidos los datos de debemos almacenar, procederemos al diseño de la base de datos para su implantación.

Para lo cual, partiremos del diseño del modelo entidad-relación, posteriormente crearemos el modelo relacional y aplicaremos las FN hasta la 3FN.

De esta forma obtendremos el esquema que nos permitirá crear nuestras consultas dentro de la aplicación que generarán las tablas y las relaciones entre ellas.

MODELO ENTIDAD RELACIÓN.

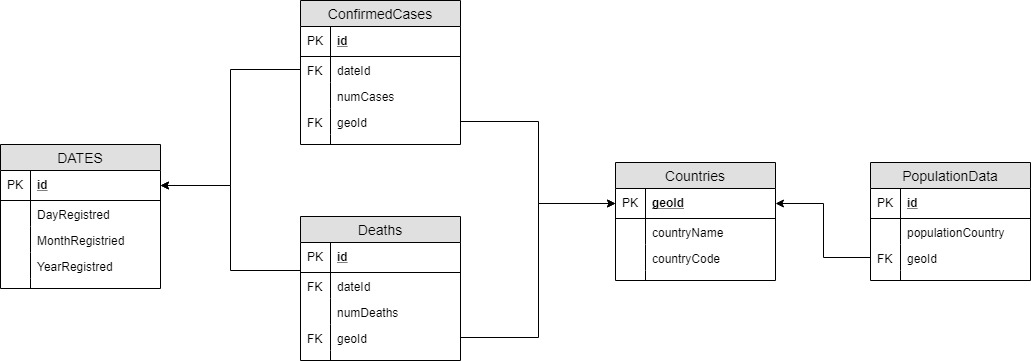


Pasemos a revisar las relaciones que se establecen en el modelo:

* En una fecha determinada (DATE) habrá una cantidad de casos confirmados y de fallecimientos (solo un número total por cada fecha).
* Un número de fallecidos o confirmados puede repetirse en varias fechas (si no hay nuevos contagios o fallecimientos).
* Por cada país hay un número total de fallecidos y de casos confirmados, dicho número de casos puede ser el mismo en diferentes países.
* Un país tiene una densidad de población concreta, puede haber países que posean la misma densidad de población.

Una vez establecidas las entidades que conforman el modelo, así como sus relaciones, pasamos dicho modelo a un nivel más de abstracción obteniendo el modelo relacional.

MODELO RELACIONAL



A partir de este modelo relacional podemos analizar si respeta la normalización buscada.

1FN

Podemos asegurar que la primera forma normal es respetada puesto que no hay atributos NO atómicos en las tablas, por lo que podemos pasar a analizar si cumple la segunda forma normal

2FN

Puesto que no hay claves primarias compuestas en ninguna tabla, este diseño cumple con la segunda forma normal.

3FN

Este diseño cumple la 3FN puesto que todos los atributos de las tablas que no son clave primaria no dependen de forma transitiva de la clave primaria.

A partir de este punto, y tras el análisis previo, podemos asegurar que estamos ante un diseño correcto por lo que podemos pasar a realizar los preparativos de nuestro sistema para correr aplicaciones diseñadas en php para, posteriormente desarrollar el programa propiamente dicho.

1. **Implementación mediante php.**

Para la preparación del entorno de trabajo precisamos un servidor web, en este caso emplearemos Apache, php (Emplearemos la versión 7.4) instalado en el sistema y un motor de bases de datos, en este caso emplearemos MariaDb.

El sistema operativo sobre el que trabajaremos es Linux Mint en su versión 19. Dispongo de un motor de bases de datos en servidor local (192.168.1.52:3306) por lo que haremos uso de ella.

Comenzaremos por instalar todo lo necesario en nuestro sistema.

Primero actualizaremos nuestros repositorios y paquetes mediante:

*apt update*

*apt upgrade*

Posteriormente instalaremos el servidor web Apache

*apt install apache2*

Posteriormente instalamos los paquetes de php y alguna de sus extensiones que pueden sernos útiles.

*apt install php-7.4 php7.4 libapache2-mod-php7.4 php7.4-cli php7.4-mysql*

Posteriormente reiniciamos nuestro servidor web mediante el comando

*service apache2 restart*

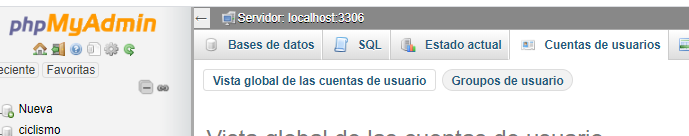
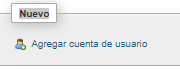
Y ya dispondremos de nuestro sistema listo para desarrollar nuestro software en PHP. Cade indicar que nuestra web debe estar en el directorio var/www/html para poder ser visto en el navegador.

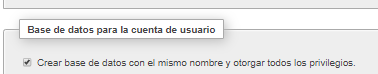
1. **Creación de la base de datos en el gestor de bases de datos.**

A continuación, debemos crear la base de datos que va a contener todos los datos, así como el usuario con el que configuraremos nuestra aplicación, el cual debe tener permisos para poder manipular dicha base de datos en concreto.

Se ha decidido nombrar a la base de datos covidxml y al usuario de la misma forma. La contraseña será autogenerada de forma aleatoria en el momento de la creación del usuario y la debemos guardar puesto que la requeriremos para hacer la conexión a la misma.

Una de las formas más rápidas para crear un usuario con su base de datos asociada es mediante phpMyadmin.

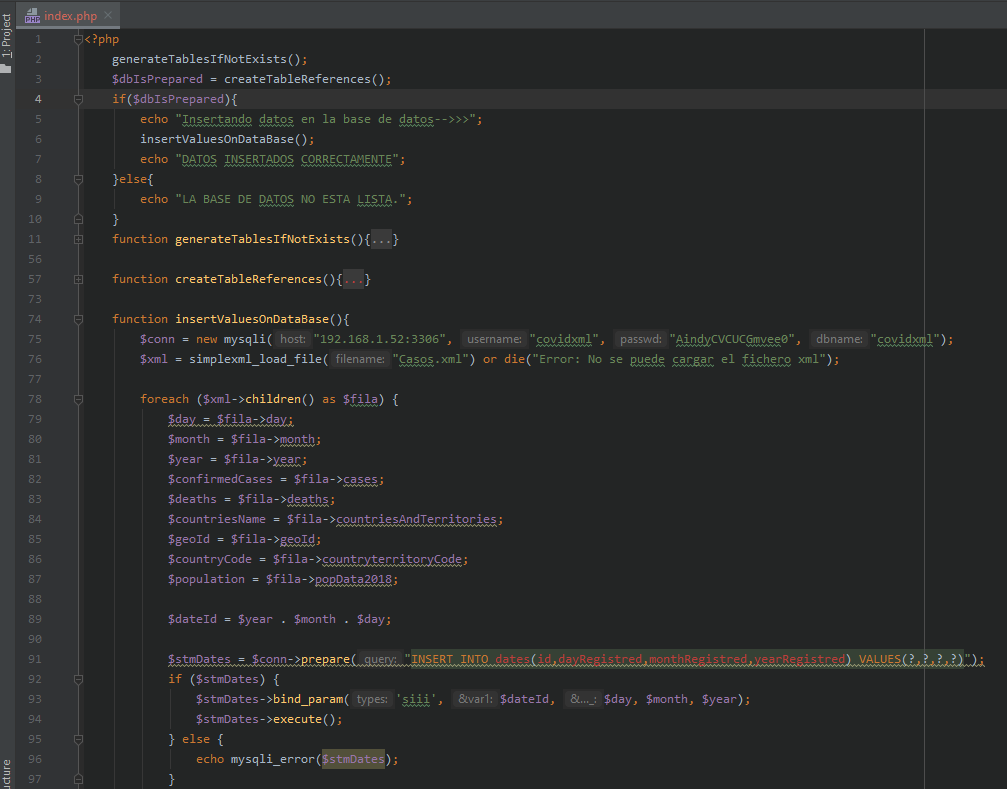
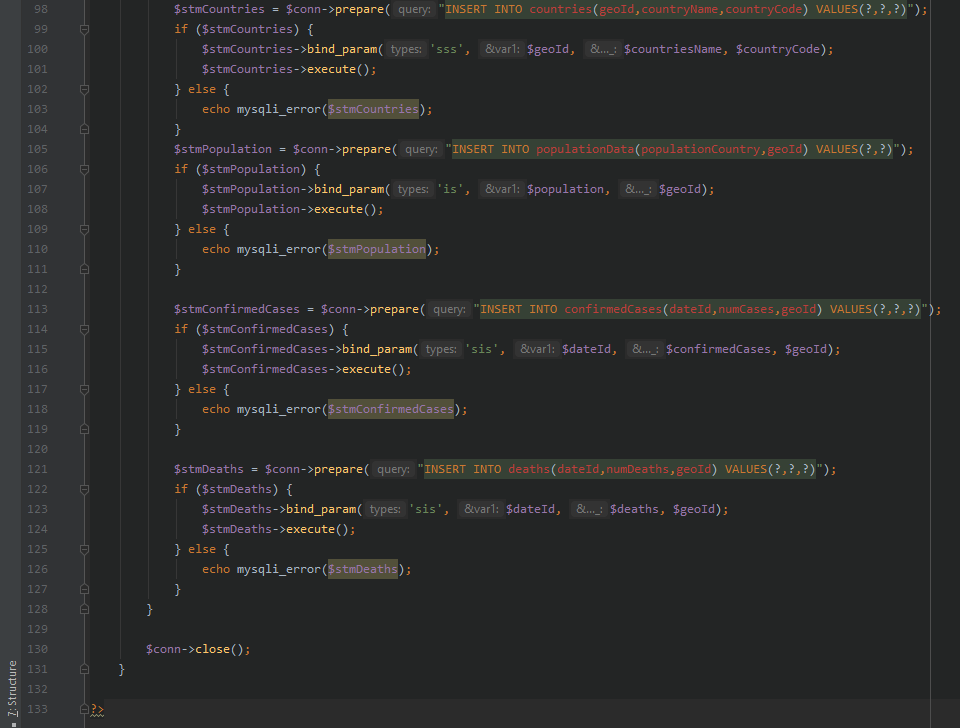
Una vez logeados, seleccionamos la opción Cuentas de usuario y añadimos un nuevo usuario.

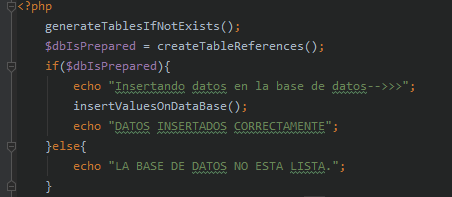
En el nuevo panel rellenamos el formulario de acuerdo a las preferencias que se requieran, y en el momento de insertar contraseña, dejamos que el sistema genere una aleatoria (debemos guardarla para su uso posterior). Nos aseguraremos de indicar que cree una base de datos para dicho usuario.

Con esto último ya dispondremos de una base de datos con la que trabajar.

1. **Creación del software mediante php.**

Pasemos a crear nuestra aplicación en php, abajo se muestra le código de la aplicación que se ha realizado y posteriormente a este, se detalla el flujo del programa y las funciones que lo componen.



Comencemos a analizar el código propuesto parte por parte.

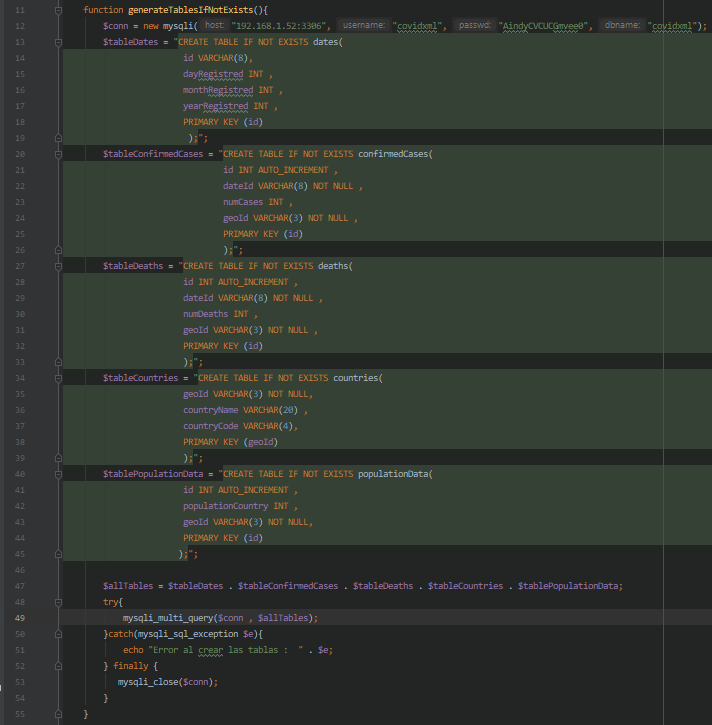
Al comienzo de la ejecución del código, lanzamos la función *generateTablesIfNotExists()* que nos generará las tablas de la base de datos.

Al terminar de ejecutarse esta función, ejecutaremos la función que se encarga de crear las referencias entre las tablas denominada *createTableReferences()* , esta función devuelve TRUE si se ha conseguido ejecutar correctamente, con lo que conseguimos saber si se ha creado correctamente la estructura de la base de datos correctamente, en caso contrario , nos muestra un mensaje de error . Si la respuesta fuera correcta continua con el flujo de ejecución ejecutando la función *insertValuesOnDAtabase()* que se encarga de serializar los datos xml e insertarlos en la base de datos.

Por motivos de simplicidad se ha optado por no separar las dos tareas que realiza esta última función pese a romper el principio de responsabilidad única ([Principios SOLID](https://enmilocalfunciona.io/principios-solid/)).

Una vez analizado el flujo de ejecución pasemos a revisar por encima las funciones que componen el programa.

***generateTablesIfNotExists()***

En 12 crea una conexión con la base de datos con la que realizaremos las consultas sql necesarias.

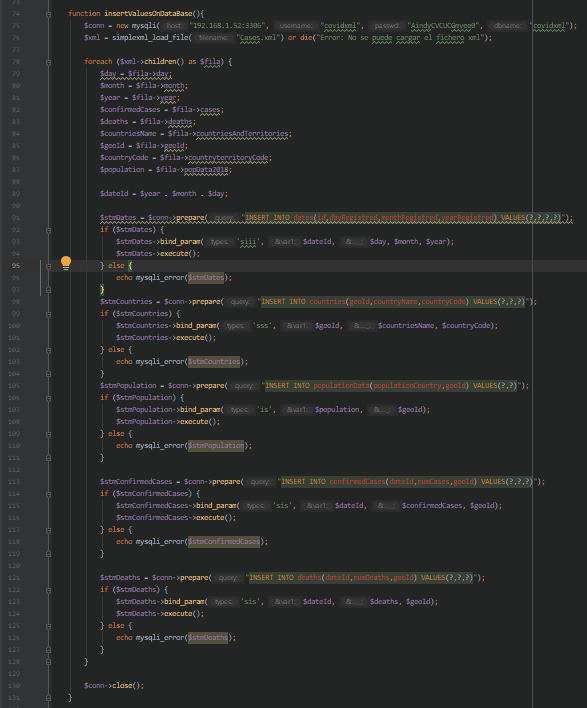
Posteriormente almacenamos en variables las consultas de creación de cada una de las tablas (String).

Emplearemos el helper mysqly\_multi\_query que nos permite realizar varias consultas seguidas, por lo que concatenaremos todas las consultas para formar un String multiconsulta (47) y posteriormente realizaremos la consulta a la base de datos. Puesto que las consultas a la base de datos pueden generar un error, debemos capturarlo mediante un try-catch y actuar en consecuencia. Finalmente cerramos la conexión a la BBDD para liberar recursos.

**GenerateTablesIfNotExists()**

Después de realizar una conexión a la base de datos en 58, almacenamos en la variable tableReferences el String de la consulta que nos creará las referencias entre las tablas de la base de datos, para , posteriormente realizar la consulta de la misma manera que podemos observar en la función *generateTablesIfNotExists()* , en una refactorización de código posterior sería interesante extraer esta parte del código en una función privada para poder utilizarla en las dos funciones sin repetir código.

**insertValuesOnDatabase()**



Después de crear nuestra conexión a la base de datos, Cargamos nuestro archivo xml mediante el método simple\_load\_file.

Mediante un foreEach recorremos todos los registros del archivo, accediendo a cada uno de los hijos (Estos son cada uno de los registros que queremos procesar)

Por cada uno de los registros, realizaremos tres tareas:

Primero extraemos y guardamos en una variable cada uno de los datos que nos interesan del registro (de 85 a 93).

Segundo creamos un String con la fecha para generar el id de la fecha (***Se ha optado por este método ya que para trabajar con Date en la base de datos deberíamos realizar varios casteos desde el String hasta el Datesql por lo que complicaríamos el código y no es el objetivo).***

El tercer paso vamos a crear las consultas SQL, pero, esta vez, ya que vamos a introducir datos, y por motivos de seguridad, realizaremos la creación de la consulta mediante un preparedStatement, con el que tenemos mayor control sobre los datos que vamos a introducir. Para ello:

Primero creamos la consulta preparada y en la posición en la que deben ir los valores a introducir ponemos “?”. Ya que puede haber fallos al generar la consulta preparada, mediante un condicional permitiremos que continúe con el Binding y la ejecución siempre que este correcto, en caso contrario imprimiremos el error para poder tracearlo.

Segundo, bindeamos los parámetros, es decir le indicamos en cada uno de los “?” que tipo de valor va a contener y la variable que contiene dicho valor.

Tercer paso, ejecutaremos la consulta preparada mediante ->execute().

Estos pasos se repiten con cada una de las tablas y los datos, insertando los datos que correspondan en cada uno de las tablas, a su vez, este proceso se repite por cada uno de los registros del archivo xml hasta llegar al final, llenando todos los datos que nos interesa tener almacenados.

Una vez tenemos todos los datos cargados cerramos la conexión (130).