## Sistemas Bancarios para Universidades

Conjunto de instituciones de una economía que ayudan a conectar el ahorro y la inversión.

Está integrado por las entidades de depósito: bancos, cajas de ahorro, incluida la Confederación Española de Cajas de Horro (CECA), las cooperativas de crédito y cajas rurales. La característica fundamental de estas instituciones al es tener como principal fuente la financiación la obtención de depósitos trasferibles, procedentes principalmente del sector privado.

Realizan inversiones y captación de recursos por cuenta propia e intermedian la gran mayoría de las transacciones financieras. Dicho en otras palabras: canalizan el ahorro que generan fundamentalmente las familias (unidades de gasto con superávit de ahorro), hacia las empresas (prestatarios o unidades de gasto con déficit de ahorro).

LA INDUSTRIA BANCARIA Y FINANCIERA  
El Sistema Bancario Financiero BYTE, ha sido diseñada con el objeto de proveer una herramienta de operación, control y mercadeo, a toda institución financiera que requiera mantenerse a la vanguardia en un mercado que cada día es más competitivo.   
  
El Sistema Bancario Financiero BYTE, es un conjunto de módulos independientes e integrables, que permiten la automatización de todos los departamentos, según sean las necesidades y posibilidades de inversión, lo que quiere decir que para iniciar un proyecto no se requiere realizar una gran inversión, sino en forma gradual según sea el avance de su proyecto.  
  
Este sistema contempla todas las operaciones que una institución financiera realiza, tanto administrativas como operativas, y esto lo hace con sus más de 80 módulos, los cuales, integralmente, son la única opción en el mercado que le permite un superlativo grado de automatización con la más alta tecnología.   
  
Aumento de la Productividad   
Hoy en día los sistemas de información no se limitan únicamente a ayudar en la automatización de operaciones, van más allá, es decir, conforman una herramienta que ayuda a la institución en general, aumentando la productividad de las áreas operativas y de negocio. La Solución Bancaria de BYTE lo realiza aprovechando las múltiples ventajas que le caracterizan, destacando lo siguiente:   
  
Sistema altamente paramétrico, que permite a la institución reaccionar rápidamente ante los cambios y nuevas necesidades.   
  
Estándares de diseño en todos sus módulos, tanto en operación como en presentación, lo que permite fácil adaptación a los usuarios de los diferentes módulos.   
  
Ambiente gráfico que permite que el proceso de capacitación de los usuarios sea más rápido.   
  
Integración de las transacciones operativas al sistema contable, por medio de interfaces amigables y flexibles.   
  
Manuales en línea al alcance de todos los usuarios, que brinda una ayuda rápida y eficiente, reduciendo el requerimiento de personal especializado para soporte a usuarios.   
 Normalización nivel 2 en Base de Datos

Normalización de [bases de datos](http://es.wikipedia.org/wiki/Base_de_datos) consiste en aplicar una serie de reglas a las relaciones obtenidas tras el paso del [modelo entidad-relación](http://es.wikipedia.org/wiki/Modelo_E-R) al [modelo relacional](http://es.wikipedia.org/wiki/Modelo_relacional).

Las bases de datos relacionales se normalizan para:

* Evitar la [redundancia](http://es.wikipedia.org/wiki/Redundancia) de los datos.
* Disminuir problemas de actualización de los datos en las tablas.
* Proteger la [integridad](http://es.wikipedia.org/wiki/Integridad_de_datos) de los datos.

En el modelo relacional es frecuente llamar [tabla](http://es.wikipedia.org/wiki/Tabla_(base_de_datos)) a una relación, aunque para que una tabla sea considerada como una relación tiene que cumplir con algunas restricciones:

* Cada tabla debe tener su nombre único.
* No puede haber dos [filas](http://es.wikipedia.org/wiki/Fila_(base_de_datos)) iguales. No se permiten los duplicados.
* Todos los datos en una [columna](http://es.wikipedia.org/wiki/Columna_(base_de_datos)) deben ser del mismo tipo.

## Normalización de Bases de Datos y Técnicas de diseño.

Uno de los factores más importantes en la creación de páginas web dinámicas es el diseño de las Bases de Datos (BD). Si tus tablas no están correctamente diseñadas, te pueden causar un montón de dolores de cabeza cuando tengas de realizar complicadísimas llamadas SQL en el código PHP para extraer los datos que necesitas. Si conoces como establecer las relaciones entre los datos y la normalización de estos, estarás preparado para comenzar a desarrollar tu aplicación en PHP. Si trabajas con MySQL o con Oracle, debes conocer los métodos de normalización del diseño de las tablas en tu sistema de BD relacional. Estos métodos pueden ayudarte a

Hacer tu código PHP más fácil de comprender, ampliar, y en determinados casos, incluso hacer tu aplicación más rápida. Básicamente, las reglas de Normalización están encaminadas a eliminar redundancias e inconsistencias de dependencia en el diseño de las tablas. Más tarde explicaré lo que esto significa mientras vemos los cinco pasos progresivos para normalizar, tienes que tener en cuenta que debes crear una BD funcional y eficiente. También detallaré los tipos de relaciones que tu estructura de datos puede tener. Digamos que queremos crear una tabla con la información de usuarios, y los datos a guardar son el nombre, la empresa, la dirección de la empresa y algún e-mail, o bien URL si las tienen. En principio comenzarías definiendo la estructura de una tabla como esta:

Formalización CERO

Usuarios

Nombre empresa direccion\_empresa url1 url2

Joe ABC 1 Work Lane abc.com xyz.com

Jill XYZ 1 Job Street abc.com xyz.com

Diríamos que la anterior tabla esta en nivel de Formalización Cero porque ninguna de nuestras reglas de normalización ha sido aplicada. Observa los campos url1 y url2 -- ¿Qué haremos cuando en nuestra aplicación necesitemos una tercera url? ¿Quieres tener que añadir otro campo/columna a tu tabla y tener que reprogramar toda la entrada de datos de tu código PHP? Obviamente no, tú quieres crear un sistema funcional que pueda crecer y adaptarse fácilmente a los nuevos requisitos. Echemos un vistazo a las reglas del Primer Nivel de Formalización-Normalización, y las aplicaremos a nuestra tabla.

Primer nivel de Formalización/Normalización. (F/N)

1. Eliminar los grupos repetitivos de la tablas individuales.

2. Crear una tabla separada por cada grupo de datos relacionados.

3. Identificar cada grupo de datos relacionados con una clave primaria. ¿Ves que estamos rompiendo la primera regla cuando repetimos los campos

url1 y url2 ? ¿Y qué pasa con la tercera regla, la clave primaria? La regla tres básicamente significa que tenemos que poner un campo tipo contador autoincrementable para cada registro. De otra forma, ¿Qué pasaría si tuviéramos dos usuarios llamados Joe y queremos diferenciarlos. Una vez que aplicáramos el primer nivel de F/N nos encontraríamos con la siguiente tabla:

Usuarios

UserId nombre empresa direccion\_empresa url

One Joe ABC 1 Work Lane abc.com

One Joe ABC 1 Work Lane xyz.com

2 Jill XYZ 1 Job Street abc.com

2 Jill XYZ 1 Job Street xyz.com

Ahora diremos que nuestra tabla está en el primer nivel de F/N. Hemos solucionado el problema de la limitación del campo url. Pero sin embargo vemos otros problemas. Cada vez que introducimos un nuevo registro en la tabla usuarios, tenemos que duplicar el nombre de la empresa y del usuario. No sólo nuestra BD crecerá muchísimo, sino que será muy fácil que la BD se corrompa si escribimos mal alguno de los datos redundantes. Aplicaremos pues el segundo

Nivel de F/N:

Segundo nivel de F/N

1. Crear tablas separadas para aquellos grupos de datos que se aplican a varios

Registros.

2. Relacionar estas tablas mediante una clave externa.

Hemos separado el campo url en otra tabla, de forma que podemos añadir más en el futuro si tener que duplicar los demás datos. También vamos a usar nuestra clave primaria para relacionar estos campos:

Usuarios

UserId nombre empresa direccion\_empresa

1 Joe ABC 1 Work Lane

2 Jill XYZ 1 Job Street

urls

urlId relUserId url 1 1 abc.com

2 1 xyz.com

3 2 abc.com

4 2 xyz.com

Vale, hemos creado tablas separadas y la clave primaria en la tabla usuarios, userId, está relacionada ahora con la clave externa en la tabla urls, relUserId. Esto está mejor. ¿Pero qué ocurre cuando queremos añadir otro empleado a la empresa ABC? ¿O 200 empleados? Ahora tenemos el nombre de la empresa y su dirección duplicándose, otra situación que puede inducirnos a introducir errores en nuestros datos. Así que tendremos que aplicar el tercer nivel de F/N: Tercer nivel de F/N.

1. Eliminar aquellos campos que no dependan de la clave. Nuestro nombre de empresa y su dirección no tienen nada que ver con el campo userId, así que tienen que tener su propia empresa Id:

Usuarios

userId nombre relEmpresaId

1 Joe 1

2 Jill 2

Empresas

emprId empresa direccion\_empresa

1 ABC 1 Work Lane

2 XYZ 1 Job Street

urls

urlId RelUserId url

1 1 abc.com

2 1 xyz.com

3 2 abc.com

4 2 xyz.com

Ahora tenemos la clave primaria emprId en la tabla empresas relacionadas con la clave externa recEmpresaId en la tabla usuarios, y podemos añadir 200 usuarios mientras que sólo tenemos que insertar el nombre 'ABC' una vez. Nuestras tablas de usuarios y urls pueden crecer todo lo que quieran sin duplicación ni corrupción de datos. La mayoría de los desarrolladores dicen que el tercer nivel de F/N es suficiente, que nuestro esquema de datos puede manejar fácilmente los datos obtenidos de una cualquier empresa en su totalidad, y en la mayoría de los casos esto será cierto.

## Query en php

Del inglés, consulta realizada contra una base de datos. Se usa para obtener datos,

modificarlos o bien borrarlos.

## ¿Cómo puedo hacer un query a la base de datos usando PHP?

Instrucciones básicas para hacer un query de SQL desde un script en PHP.

Para ejecutar este ejemplo, es necesario que guardes los comandos en un archivo con extensión .php y que tu cuenta web tenga soporte para bases de datos.

El primer paso necesario es conectar tu script a la base de datos, esto se logra usando la función mysql\_connect que toma como parametros el servidor, el nombre de usuario y el password:

mysql\_connect("localhost","userid","passwd");

(Reemplaza el texto en azul con los datos correspondientes a tu cuenta).

Segundo paso. Una vez conectada la base de datos, procede a pasar el query en SQL usando la funcion mysql() de PHP. Esta función recibe como parámetro la base de datos y el query en lenguaje SQL, por ejemplo:

$q = mysql("userid","[QUERY DE SQL]");

(Reemplaza el texto azul con tu userid y el verde con tu query en SQL).

En este punto, el query está hecho. Para saber cuantas líneas regreso tu consulta, puedes usar la funcion mysql\_numrows() que toma como parámetro el objeto que regreso tu mysql():

$num = mysql\_numrows($q);

Ahora, $num tiene asignado el número de líneas que regreso el query.

Finalmente para sacar los datos que regresó el query (en caso de queries de SELECT), usa la función mysql\_Result(), ésta toma como parámetros el objeto del query, el número de línea y el campo que quieres ver:

print mysql\_Result($q,num\_linea,"campo");

Una vez más, reemplaza los textos coloreados. Recuerda que el número de la línea siempre deberá ser menor al valor regresado por mysql\_numrows() o de lo contrario PHP sacará un error cuando no pueda leer una línea inexistente. Igualmente, campo deberá ser un campo válido del query que realizaste para poder aceptarlo (y entre comillas!).

Esta es una descripción muy básica para realizar un query. Consulta la [documentación oficial de PHP](http://mx.php.net/manual/es/index.php) para conocer el gran poder de este lenguaje

### Ejemplos[¶](http://www.php.net/manual/es/function.mysql-query.php#refsect1-function.mysql-query-examples)

**Ejemplo #1 Consulta inválida**

La siguiente consulta es sintácticamente inválida, por lo que **mysql\_query()** fallará y retornará FALSE.

<?php  
$resultado = mysql\_query('SELECT \* WHERE 1=1');  
if (!$resultado) {  
    die('Consulta no válida: ' . mysql\_error());  
}  
  
?>

**Ejemplo #2 Consulta válida**

La siguiente consulta es válida, por lo que **mysql\_query()** retornará un [resource](http://www.php.net/manual/es/language.types.resource.php).

<?php  
// Lo siguiente podría ser proporcionado por un usuario, como por ejemplo  
$nombre = 'fred';  
$apellido  = 'fox';  
  
// Formular la consulta  
// Este es el mejor método para formular una consulta SQL  
// Para más ejemplos, consulte mysql\_real\_escape\_string()  
$consulta = sprintf("SELECT nombre, apellido, direccion, edad FROM amigos   
    WHERE nombre='%s' AND apellido='%s'",  
    mysql\_real\_escape\_string($nombre),  
    mysql\_real\_escape\_string($apellido));  
  
// Ejecutar la consulta  
$resultado = mysql\_query($consulta);  
  
// Comprobar el resultado  
// Lo siguiente muestra la consulta real enviada a MySQL, y el error ocurrido. Útil para depuración.  
if (!$resultado) {  
    $mensaje  = 'Consulta no válida: ' . mysql\_error() . "\n";  
    $mensaje .= 'Consulta completa: ' . $consulta;  
    die($mensaje);  
}  
  
// Usar el resultado  
// Si se intenta imprimir $resultado no será posible acceder a la información del recurso  
// Se debe usar una de las funciones de resultados de mysql  
// Consulte también mysql\_result(), mysql\_fetch\_array(), mysql\_fetch\_row(), etc.  
while ($fila = mysql\_fetch\_assoc($resultado)) {  
    echo $fila['nombre'];  
    echo $fila['apellido'];  
    echo $fila['direccion'];  
    echo $fila['edad'];  
}  
  
// Liberar los recursos asociados con el conjunto de resultados  
// Esto se ejecutado automáticamente al finalizar el script.  
mysql\_free\_result($resultado);  
?>

## Análisis y modelado de Base de Datos

En la siguiente página esta un documento que contiene todas las características necesarias para la etapa de modelado:

<http://ict.udlap.mx/people/carlos/is341/bases02.html>

NALISIS DE BASE DE DATOS

Para ello necesitaremos de:

➢

LDD: Con el que (por ejemplo) codificar las sentencias para lacreación de las distintas tablas de la base de datos.

➢

LMD: Para codificar las instrucciones (que por ejemplo) se encargarán de realizar: Consultas, Adiciones, Eliminaciones de Registros. Diseño físico

El diseño físico es el proceso de producir la descripción de la implementación de la base de datos en memoria secundaria: estructuras de almacenamiento y métodos de acceso que garanticen un acceso eficiente a los datos. En general, el propósito del diseño físico es describir cómo se va a implementar físicamente el esquema lógico obtenido en la fase anterior. Concretamente, en el modelo relacional, esto consiste en:

➢

Obtener un conjunto de relaciones (tablas) y las restricciones que se deben cumplir sobre ellas.

➢

Determinar las estructuras de almacenamiento y los métodos deacceso que se van a utilizar para conseguir unas prestacionesóptimas.

➢

Diseñar el modelo de seguridad del sistema. Diseño conceptual

En esta etapa se debe construir un esquema de la información que se usa en la empresa, independientemente de cualquier consideración física. A este esquema se le denomina esquema conceptual. Al construir el esquema, los diseñadores descubren la semántica (significado) de los datos de la empresa: encuentran entidades, atributos y relaciones. El objetivo es comprender:

➢

La perspectiva que cada usuario tiene de los datos.

➢

La naturaleza de los datos, independientemente en su representación física.

➢

El uso de los datos a través de las áreas de aplicación.

El esquema conceptual se puede utilizar para que el diseñador transmita a la empresa lo que ha entendido sobre la información que ésta maneja. Para ello, ambas partes deben estar familiarizadas con la notación utilizada en el esquema. La más popular es la notación del modelo entidad-relación, que se describirá en el capítulo dedicado al diseño conceptual.

Diseño lógico

El diseño lógico es el proceso de construir un esquema de la información que utiliza la empresa, basándose en un modelo de base de datos específico, independiente del SGBD concreto que se vaya a utilizar y de cualquier otra consideración física. En esta etapa, se transforma el esquema conceptual en un esquema lógico queutilizará las estructuras de datos del modelo de base de datos en el que se basa el SGBD que se vaya a utilizar, como puede ser el modelo relacional, el modelo de red, el modelo jerárquico o el modelo orientado a objetos. Conforme se va desarrollando el esquema lógico, éste se va probando y validando con los requisitos de usuario. La normalización

es una técnica que se utiliza para comprobar la validez de losesquemas lógicos basados en el modelo relacional, ya que asegura que las relaciones(tablas) obtenidas no tienen datos redundantes. Esta técnica se presenta en el capítulodedicado al diseño lógico de bases de datos.

## Características principales de las Cajas de Ahorro:

* Las cajas de ahorro se constituyen bajo la forma jurídica de fundaciones de naturaleza privada con finalidad social, lo que las diferencia de los bancos que son sociedades anónimas,
* Peculiaridades en la composición de sus órganos de gobierno, en los que habrá una representación adecuada de las Corporaciones Locales, de los impositores y de grupos que representen intereses sociales y colectivos (Ley de Órganos Rectores de Cajas de Ahorro).
* Actúan bajo criterios mercantiles pero un porcentaje significativo de sus beneficios revierten en su obra social.
* Las cajas tienen prohibido acudir a los mercados de capitales para aumentar sus recursos propios.
* Las cajas tienen una fuerte implantación local, que se deriva en parte de la participación de las Corporaciones Locales en sus órganos de

Órganos Rectores de las Cajas de Ahorro

* Asamblea General de una Caja de Ahorros, es el órgano constituido por las representaciones de los intereses sociales y colectivos del ámbito de actuación de la caja de ahorro que asume el supremo gobierno y decisión de la entidad. La representación de los intereses colectivos en la Asamblea General se llevará a efectos mediante la participación de los Consejeros, que serán designados en un mínimo de 60 y máximo de 160 siguiendo los siguientes criterios: 40% por las Corporaciones Municipales en cuyo término tenga abierta oficina la entidad; 44% por los impositores de la caja; 11% por las personas o entidades fundadoras de la caja y 5% por los representantes de los empleados trabajadores en la caja.
* Consejo de Administración de una Caja de Ahorros, el órgano que tiene encomendada la administración y gestión financiera, así como la de la obra benéfico social de la caja. Al igual que en la Asamblea General, en el Consejo de Administración también están representados los intereses colectivos en similares porcentajes a cada grupo.
* Comisión de Control de una Caja de Ahorros, tiene por objeto cuidar que la gestión del Consejo de Administración se cumpla con la máxima eficacia y precisión, dentro de las líneas generales de actuación señaladas por la Asamblea General y de las directrices emanadas de la normativa financiera