|  |
| --- |
| COSOF 2014 |
| Normas básicas para el diseño de bases de datos |
|  |
|  |
| **Domingo Andrés Cortez Jara** |
| **10/06/2013** |

|  |
| --- |
|  |

Contenido

[1. Introducción](#_gjdgxs)

[2. Normas de desarrollo](#_30j0zll)

[3. Normas básicas para nombrar nuevas tablas:](#_1fob9te)

[4. Nombra básicas para el nombramiento de columnas.](#_3znysh7)

[5. Normas básicas para creación de procedimientos almacenados.](#_2et92p0)

[6. Desarrollo de procedimientos almacenados.](#_tyjcwt)

# Introducción

Dentro de las distintas organizaciones es muy importante y necesario mantener un orden en la forma de trabajo, ya sea en cuanto a la documentación, contratos , reuniones o maneras de enfrentar distintas situaciones, todo esto con el objetivo de minimizar los tiempos para integrar nuevos elementos al grupo, generar facilidad en la comprensión, ya que, al enseñar las normas al nuevo personal nos aseguramos un rápido entendimiento de las metodologías, a la vez, esto se aplica también para los proyectos propios o que se remontan después de un tiempo.

# Normas de desarrollo

Vamos a definir algunos criterios que se deben utilizar para la creación de distintos elementos utilizados en desarrollo, tales como:

* Tablas
* Procedimientos almacenados
* Variables
* Archivos PHP, JS y TPL.
* AJAX y XAJAX.

Desarrollo de bases de datos

La parte fundamental dentro de un sistema es la base de datos, por lo tanto, cuando se requiere modificar o mejorar un sistema que ya existe, es muy importante poder entender con facilidad la lógica detrás de las tablas y sus relaciones.

# Normas básicas para nombrar nuevas tablas:

* Es importante que la tabla refleje exactamente lo que contiene, esto suena muy obvio, pero no siempre se aplica, sobre todo en las tablas de relaciones.

Como normas se definen los siguientes criterios:

* + El nombre debe comenzar con la sigla del sistema al que pertenecen, lo ideal es que no sean 3 caracteres, ya que, más información confunde y quita espacio a la descripción que es lo más importante. Luego de los 3 caracteres se aplica la descripción del objetivo de la tabla separado por “\_”, todo en mayúsculas y siempre en singular. La idea de ponerlo en singular es para tener claridad de cómo sería el nombre de la tabla, si nos ponemos de acuerdo nunca dudaremos si la tabla era persona o personas.

Ejemplo:

Tabla para almacenar oficinas en sistema de contratos (CNT)

El nombre de la tabla seria:

CNT\_OFICINA

* + Es importante nombrar las tablas relacionadas con una codificación que permita verlas rápidamente el buscar por nombres, por ejemplo, la tabla anterior es para almacenar oficinas, pero además vamos a agregar una tabla que tiene las posibles zonas para estas oficinas, entonces lo mas adecuado es generar la nueva tabla con nombre CNT\_OFICINA\_ZONA, esto permite entender en forma visual que la tabla de zonas corresponde a una descripción o relación de la tabla de oficinas.

# Nombra básicas para el nombramiento de columnas.

* Todos los nombres de campos se escriben con minúsculas.
* Al igual que los nombres de las tablas se debe mantener un orden y agrupamiento en el nombre de las columnas.
* Los nombres de las columnas deben reflejar fácilmente el contenido de estas, por lo tanto no se usaran abreviaciones a menos que sea estrictamente necesario por espacio, de ser así, no utilizara menos de 5 caracteres para dicha abreviación.
* Todas las columnas de las tablas deben ser definidas con prefijo de 2 letras que indican el tipo de datos que contiene, para codificar estos datos utilizaremos:
  + id (INT): Corresponde a códigos numéricos automáticos o correlativos
  + cd (varchar(n)): Corresponde al código propio de la fila, debe ser alfanumérico
  + fc (datetime): Corresponde a las fecha de sistema, si se requiere utilizar fecha y hora, se deben concatenar al momento de ser guardados
  + gl (varchar(n)): Corresponde a texto de cualquier tipo, ya sea nombre o descripciones
  + bo (bit o boolean): Corresponde a campos que tienen comportamiento binario, el 1 significará si o encendido y el cero significará no o apagado.
  + nr (int): Corresponde a cualquier número que se requiera almacenar, es importante definirlo bien, ya que en algunos motores de BD si el campo es vacío, significa error en el procedimiento. En algunos casos un numero puede ser considerado como una glosa, por ejemplo en números de teléfonos o números de direcciones, la idea es no utilizar campos números (nr) para campos que no requieren operaciones, algún tipo de cálculo o algún agrupamiento.

Ejemplo de columnas para una tabla de usuarios:

@gl\_nombre\_primero VARCHAR (50)

@gl\_nombre\_segundo VARCHAR (50)

@gl\_apellido\_paterno VARCHAR (50)

@gl\_apellido\_materno VARCHAR (50)

@gl\_direccion VARCHAR (120)

@gl\_fono VARCHAR (20)

@gl\_email VARCHAR (50)

@id\_perfil INT

@nr\_ficha INT

@cd\_empresa INT

@bo\_vigente BIT

@gl\_clave VARCHAR (20)

@gl\_login VARCHAR (20)

El largo de los campos de texto se fijara en:

* Nombres y apellidos (cada parte en columnas separadas): varchar 50
* Direcciones: varchar 120
* Teléfono fijo y móvil: varchar 20
* Todos los números: INT
* Todas las fechas: datetime

# Normas básicas para creación de procedimientos almacenados.

* El nombre debe ser escrito todo en minúsculas y separado por “\_”.
* Los nombres de los procedimientos almacenados deben comenzar con las letras spa (stored procedureauditing), luego se define en 3 letras correspondientes a la sigla del sistema, luego 4 caracteres que indiquen la operación principal, es decir:
  + - Ins (Insert)
    - Obt (select)
    - Act (update)
    - Eli (Delete o update para mantener integridad)

Luego lo ideal es definir la tabla principal sobre la cual se desarrolla la aplicación, si se conserva este orden se puede revisar rápidamente los procedimientos y localizar el que se requiere.

Por ejemplo:

Se requiere generar los 3 procedimientos básicos para un mantenedor de usuarios en el sistema de contratos (CNT):

Insertar

* + - spa\_cnt\_ins\_usuario

Consultar

* + - spa\_cnt\_obt\_usuario

Observacion: En el caso de los select a veces se requiere buscar por distintos criterios, por ejemplo, por id lógico o por código, en estos casos lo mejor es agregar al final del nombre el criterio de búsqueda a utilizar:

* + - spa\_cnt\_obt\_usuario\_id
    - spa\_cnt\_obt\_usuario\_login
    - spa\_cnt\_obt\_usuario\_oficina

Esto nos permitirá entender de inmediato cual es el criterio de busqueda y poder volver a utilizar el procedimiento en otra parte de nuestro código.

Actualizar:

* spa\_cnt\_act\_usuario

Eliminar

* spa\_cnt\_eli\_usuario

# Desarrollo de procedimientos almacenados.

* Los parámetros de los procedimientos deben tener el mismo nombre que la columna a la se hace referencia, en el caso de mysql se debe anteponer el prefijo “P\_”, el cual indica que se trata de un parámetro, en caso de otro motores como SYBASE, la columna tendrá el mismo nombre pero dentro del procedimiento se reconoce por ser antecedido por un “@”, ejemplo:

Tabla CNT\_USUARIO

Procedimiento de búsqueda de nombre mediante id de usuario (id\_usuario)

Detalle del procedimiento:

Create procedure spa\_cnt\_obt\_usuario\_id

(

P\_id\_usuario INT

)

/\*

Descripción: Obtener el nombre de un usuario mediante su id

Fecha: 11/06/2013

Autor: Domingo Cortez

\*/

BEGIN

Select gl\_nombre\_usuario

From CNT\_USUARIO

Where id\_usuario = P\_id\_usuario

END

* No se deben dejar consultas con “Select \*”, esto provoca problemas de performance al traer siempre todas las columnas aunque no se requieran, además puede traer problemas al recuperar campos cuando estos se llaman igual en tablas distintas. En algunos motores de BD no se puede utilizar esta sentencia, ya que, es reemplazada por el nombre de la tabla + el nombre del campo, esto significa que de todas maneras se debe eliminar las columnas que no se utilizaran.
* Todo sp debe llevar una cabecera indicando los datos de identificación, en algunos clientes existen formatos definidos, por lo que se debe consultar al momento de comenzar a desarrollar. Dentro de EBELEN se define el siguiente formato mínimo:

/\*

Descripción: Obtener el nombre de un usuario mediante su id

Fecha: 11/06/2013

Autor: Domingo Cortez

\*/

* Cada vez que se realiza una modificación a un procedimiento, se debe dejar constancia de los cambios efectuados, la fecha y el autor del cambio. Esto es necesario tanto en las modificaciones a producción como en las etapas de desarrollo, esto es necesario porque cuando varias personas están trabajando sobre los mismos objetos es necesario coordinar los cambios.

/\*

Descripción: Obtener el nombre de un usuario mediante su id

Fecha: 11/06/2013

Autor: Domingo Cortez

Modificaciones:

---------------

Modificación: Se agregar ordenamiento por gl\_nombre\_usuario

Fecha: 12/06/2013

Autor: Domingo Cortez

\*/

* Cada vez que se requiere desarrollar un nuevo sp, se debe revisar si ya existe algún procedimiento que cumpla con los que se necesita, en caso de existir, se debe usar y no generar copias con distinto nombre para hacer lo mismo.
* En caso que exista algún sp que puede ser mejorado para ser reutilizado, se debe modificar de tal manera que siga funcionando para los procesos actuales y para los nuevos requerimientos, la última persona en modificar se hace responsable de los cambios y del correcto funcionamiento del procedimiento.