Focus Consultores Asociados

Estándar de nomenclatura para Base de Datos (POSTGRES)

Manual de procedimientos

DATE \@ "d 'de' MMMM 'de' y" enero de 2015

Este documento describe el estándar de nomenclatura para base de datos utilizados en el departamento de Desarrollo.

|  |  |
| --- | --- |
| Archivo: | estandar\_nomenclatura\_bd.docx |
| Páginas: | NUMPAGES 1 |
| Fecha: | 25/12/2014 5:13:00 PM |
| Autor: | Javier Augusto Raffo Huamán |
| Contacto: | Raffo8924@gmail.com |
| Versión | 1.0 |

Tabla de contenido

TOC \t "heading 1, 1,heading 2, 2,heading 3, 3"

Introducción PAGEREF \_Toc \h 3

Objetivo PAGEREF \_Toc1 \h 3

Alcance PAGEREF \_Toc2 \h 3

Audiencia PAGEREF \_Toc3 \h 3

Fuentes utilizadas PAGEREF \_Toc4 \h 3

Condiciones de uso de este documento PAGEREF \_Toc5 \h 3

Convenciones utilizadas en este documento PAGEREF \_Toc6 \h 4

Terminología y definiciones PAGEREF \_Toc7 \h 4

Guía rápida PAGEREF \_Toc8 \h 5

Convenciones de nomenclatura PAGEREF \_Toc9 \h 5

Convenciones de Nomenclatura PAGEREF \_Toc10 \h 7

Guías genéricas y buenas prácticas PAGEREF \_Toc11 \h 7

Nomenclatura para los elementos de una base de datos PAGEREF \_Toc12 \h 7

La base de datos PAGEREF \_Toc13 \h 7

Schemas PAGEREF \_Toc14 \h 7

Tablas PAGEREF \_Toc15 \h 8

Vistas PAGEREF \_Toc16 \h 8

Columnas PAGEREF \_Toc17 \h 9

Funciones PAGEREF \_Toc18 \h 9

Triggers PAGEREF \_Toc19 \h 9

Tipos de datos definidos por el usuario PAGEREF \_Toc20 \h 9

Primary keys PAGEREF \_Toc21 \h 10

Foreign keys PAGEREF \_Toc22 \h 10

Indexes PAGEREF \_Toc23 \h 10

Variables PAGEREF \_Toc24 \h 10

GUIAS PAGEREF \_Toc25 \h 11

Relaciones 1:N PAGEREF \_Toc26 \h 11

Relaciones N:M PAGEREF \_Toc27 \h 11

BIBLIOGRAFÍA PAGEREF \_Toc28 \h 12

# Introducción

El presente documento describe la nomenclatura a utilizar en el diseño de base de datos en el departamento de desarrollo y consultoría de software de Focus Consultores Asociados.

## Objetivo

El objetivo de este documento es institucionalizar buenas prácticas y estandarizar la nomenclatura de nombres utilizada en el diseño y mantenimiento de bases de datos en el departamento de desarrollo de Focus Consultores Asociados.

## Alcance

Este documento aplica al diseño y mantenimiento de base de datos haciendo foco en el manejador de bases de datos en particular: Postgres.

## Audiencia

Este documento se encuentra dirigido a programadores, analistas, jefes de proyecto y especialistas técnicos del departamento de desarrollo de Focus Consultores Asociados, que tengan entre sus tareas realizar el diseño o mantenimiento de una base de datos.

## Fuentes utilizadas

Entre las fuentes utilizadas para la creación de este documento se encuentran diferentes publicaciones sobre nomenclatura de base de datos, las cuales son referenciadas en la sección de bibliografía, así como también se ha intentado seguir las prácticas utilizadas por Microsoft en el diseño de la base de datos Northwind.

## Condiciones de uso de este documento

Una regla puede romperse sólo ante razones justificadas, discutidas, con previa autorización del responsable del producto, y en caso que no pueda aplicarse ninguna alternativa razonable. El autor de la excepción, obligatoriamente debe documentar el código explicando la causa de la violación de la regla.

Las preferencias personales no se consideran una razón justificada.

## Convenciones utilizadas en este documento

|  |  |
| --- | --- |
| Abreviaciones | Descripción |
| OBL | Obligatorio |
| REC | Recomendado |
| **Negrita** | Texto con énfasis adicional que debe ser considerado importante. |
| Siempre | Indica que esta regla DEBE ser respetada, en los términos de este manual. |
| *Nunca* | Indica que esta acción NO DEBE ser realizada, en los términos de este manual. |
| *No* *hacer* | Indica que esta acción NO DEBE ser realizada, en los términos de este manual. |
| *Evitar* | Indica que esta práctica debe ser evitada siempre que sea posible, pero pueden existir excepciones AUTORIZADAS para su utilización. |
| *Intentar* | Indica que esta práctica debe aplicarse siempre que sea posible y apropiado. |
| Razón | Explica el propósito y las causas que motivan la regla o recomendación. |

## Terminología y definiciones

|  |  |
| --- | --- |
| Término | Descripción |
| Camel Case | Una palabra con la primera letra en minúsculas, y la primera letra de cada una de las palabras subsecuentes en mayúsculas.  Ejemplo: customerName |
| Magic Number | Cualquier literal numérico utilizado dentro de una expresión (o inicialización de variable) que no posea un significado claro. Usualmente este término no aplica a los valores 0 y 1 y cualquier otra expresión numérica equivalente que su evaluación resulte 0. |
| Pascal Case | Una palabra con la primera letra en mayúsculas, y la primera letra de cada palabra subsecuente también en mayúsculas.  Ejemplo: CustomerName |
| Hungarian Notation | Comienzan con una o mas letras en minúscula que denotan el tipo de la variable  Ejemplo: string sVariable |
| Underscore Separated | Indica palabras separadas con infraguión. Ejemplo: CUSTOMER\_DETAIL |

# Guía rápida

En esta sección se incluye un breve resumen de los principales estándares descriptos a los largo de este documento. Estas tablas no son detalladas en sus descripciones, pero brindan una rápida referencia a los elementos.

## Convenciones de nomenclatura

|  |  |
| --- | --- |
| C | Camel case |
| P | Pascal case |
| \_ | Prefijo con infraguión (underscore) |
| X | No aplica |
| [] | Lo se encuentre contenido entre paréntesis rectos significa que es opcional. |
| <VAR> | Indica que esa posición debe sustituirse por el valor del campo VAR. En el caso de la variable TABLE se hace la siguente distinción: TABLE\_S representa el nombre de una tabla en singular (ej: Customer), mientras que TABLE\_P indica el nombre de una tabla en plural (ej: Customers). |
| USU | Underscore Separated Upper Case (ej: factura\_producto). |

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Postgres** | **Observaciones** |
| **Base de datos** | <ENVIROMENT>\_<COUNTRY>\_<CUSTOMER>\_<SOLUTION>[\_AUX] | ***Ejemplo: ep\_pe\_sernanp\_erp***  ***Entornos de Ejecución:***  *EP : Entorno de Producción*  *ET : Entorno de Pruebas*  *ED : Entorno de Desarrollo* |
| **Schema** | <BUSINESS>[\_AUX] | ***Ejemplo: rrhh\_erp*** |
| **Tablas** | USU & singular | Evitar espacios en blanco ***Ejemplo: cliente*** |
| **Vistas** | VW\_<VIEW\_P> | Evitar espacios en blanco  ***Ejemplo: vw\_ventasxcliente*** |
| **Stored Procedures**  **O**  **User defined functions (Funciones)** | USU | ***Ejemplos:  FN\_<FUNCTIONALITY>\_<OPERATION>*[\_<AUX>]**  **fn\_personas\_validar\_documento** |
|  |
| **Triggers** | <TX>\_<TABLE\_S>\_<OPERATION>[\_<AUX>] | Un trigger esta siempre asociado con una tabla y una operación y no tiene sentido fuera de ellos. ***Ejemplos:***  ***tx\_orden\_insert\_validar\_datos*** |
| **Columns** | Para Comluna simple: USU  Para las claves (PK) : ID\_<TABLE\_S> | No nombrar de forma distinta campos que representen lo mismo. ***Ejemplos:  id\_orden, nombres\_completos, direccion, fecha\_registro*** |
| **User defined data types** | USU | ***Ejemplo: cliente\_id*** |
| **Primary keys** | PK\_<TABLE\_S> | ***Ejemplo: pk\_cliente*** |
| **Foreign keys** | FK\_<TABLE\_S><FIELD>\_<REF\_TABLE\_S><REF\_FIELD> | ***Ejemplo:***  ***fk\_cliente\_id\_orden\_cliente\_id\_cliente*** |
| **Indexes** | [IDX\_]<TABLE\_P>\_<FIELD>[\_AUX] | ***Ejemplo:***  ***idx\_orden\_detalle\_id\_orden\_u\_nc*** En el ejemplo presentado \_U correspondería a Unique y \_NC correspondería a NonClustered. |
| **Variables** | USU |  |

# Convenciones de Nomenclatura

A continuación se presentan un conjunto de guías y buenas prácticas, así como la nomenclatura para utilizar en el diseño de bases de datos.

## Guías genéricas y buenas prácticas

1. **OBL** – **Utilizar nombres en español para todos los elementos de la base de datos**, tablas, vistas, campos, etc.
2. **REC** – **Utilizar nombres descriptivos para los campos**. Utilizar nombres que resulten intuitivos y permitan entender el significado de los campos (mnemotécnicos). Evitar las abreviaciones, y si esto no es posible documentarlas bien.
3. **OBL**- **Utilizar solo minúsculas para nombrar los elementos de la base de datos, schemas, tablas y campos.**
4. **REC** – **No nombrar campos que representan lo mismo de forma distinta**. La forma en que se nombran iguales propiedades debe ser consistente en todo un esquema. Ejemplo: Nombrar al campo clave de la tabla Customers como Id, y después referenciarlo en otras tablas como CustomerId es INCORRECTO. El campo debe ser nombrado CustomerId en todos los casos que se quiera almacenar una clave de Customers.
5. **REC** – **Evitar tener demasiadas columnas NULLABLES en una tabla**. Esto es indicio de un esquema poco o nada normalizado. Falta de normalización puede conllevar problemas de consistencia en los datos en la medida que un mismo campo se puede terminar almacenando en varias tablas. Excesiva normalización puede tener asociada una perdida de performance en ciertas operaciones sobre la base de datos. Es necesario encontrar el equilibrio correspondiente a los requerimientos de cada proyecto en este punto. Como regla general la tercera forma normal es un buen punto intermedio.
6. **REC** – **Evitar tener tablas sin definición de primary keys**.
7. **REC** – **Evitar tener tablas innecesarias en el sistema**. Un buen diseño es uno simple (keep it simple ;)
8. **REC** – **Intentar evitar el uso de código propietario** en la definición de expresiones SQL.

Intentar utilizar código Standard SQL-92.

## Nomenclatura para los elementos de una base de datos

En esta sección se presenta la nomenclatura definida para los distintos elementos de una base de datos.

### La base de datos

La base de datos deberán nombrarse usando la siguiente nomenclatura:

**Entornos de Ejecución:**

EP : Entorno de Producción

ET : Entorno de Pruebas

ED : Entorno de Desarrollo

<ENVIROMENT>\_<PAIS>\_<CLIENTE>\_<SOLUCION>[\_AUX]

Donde se reserva AUX para algún texto adicional.

Ejemplo:

***ep\_pe\_sernanp\_erp***

### Schemas

Los sechemas de las bases de datos deberán nombrarse usando la siguiente nomenclatura:

<BUSINESS>[\_AUX]

Donde se reserva AUX para algún texto adicional.

Ejemplo:

***rrhh\_erp***

### Tablas

Las tablas deben nombrarse:

* en singular,
* en inglés
* sin utilizar espacios en blanco

Deben nombrarse con notación Underscore Separated, en mayúsculas.

Ejemplos: **CUSTOMER**, **ORDER**

En aquellos escenarios en donde se quiera agrupar tablas según cierta lógica del negocio se puede agregar un prefijo que permita esto. Por ejemplo, si en un mismo esquema se quieren almacenar empleados del departamento de recursos humanos se pueden definir de la siguiente manera:

HR\_EMPLOYE

### Vistas

Las vistas deben nombrarse con la misma notación definida para nombrar tablas, pero prefijadas usando **VW\_**.

Ejemplo:

vw\_ventas\_por\_pais

### Columnas

Los campos de una tabla corresponden a los atributos de una entidad, describen propiedades de la misma.

Las columnas deben ser nombradas según los lineamientos a continuación:

1. Los nombres deben ser simples, representativos e intuitivos.
2. Los nombres de las columnas de una tabla deben estar expresados **en singular**.
3. El campo **clave** de una tabla de nombrarse como el nombre de la tabla mas el sufijo **Id**. Ejemplo: Para una tabla de clientes (cliente), se definirían las claves:
   * id\_cliente.
4. Campos que representen la misma entidad del mundo real, deben estar nombrados de la misma manera en todas las tablas de un esquema. Por ejemplo nombrar la clave de la tabla *SALE* en una tabla como *id\_venta* y en otra *key\_venta* es incorrecto.
5. Se desaconseja prefijar sistemáticamente TODOS los campos de una tabla con el nombre de la tabla o una abreviación del mismo. Entendemos que esto agrega un nivel de redundancia y complejidad al sistema que no es necesario en manejadores modernos.

**POSTGRES**

Usar notación Underscore Separated

### Funciones

Las funciones son un espacio estándar para incluir lógica en la base de datos, expresada en un lenguaje de scripting que extiende SQL. Las funciones pueden ser invocados utilizando SQL estándar desde una aplicación, mediante la instrucción CALL.

Las funciones deben ser nombrados según la siguiente nomenclatura:

***FN\_<FUNCTIONALITY>\_<OPERATION>*[\_<AUX>]**

Usar notación Underscore Separated

Ejemplo:

fn\_gestionar\_persona\_insert

fn\_gestionar\_persona\_insert\_alterno

### Triggers

Un trigger es lógica alojada en la base de datos asociada a una determinada acción sobre una tabla. La lógica es disparada cuando ocurre la acción correspondiente.

Un trigger no tiene sentido fuera de una tabla y un trigger tiene asociada siempre una operación, por lo que dicha información debe estar asociada al nombre del trigger.

<TX>\_<TABLE>\_<OPERATION>[\_AUX]

Ejemplo: tx\_cliente\_insert\_validar

### Tipos de datos definidos por el usuario

Los tipos de datos definidos por el usuario son un mecanismo para mantener la consistencia de tipos en la base de datos. Cuando un mismo tipo de datos es utilizado en varias tablas, en vez de definirlo cada vez por separado, se define un “user defined data type” para luego referenciarlo desde todas ellas y mantener así centralizada su definición.

Ejemplo: cliente\_id

### Primary keys

La clave primaria es un conjunto de campos que identifica de forma única un registro en una tabla. Son un caso particular de un índice. La nomenclatura es la siguiente:

PK\_<TABLA>

Ejemplo: pk\_cliente

### Foreign keys

Las foreign keys son usadas para definir vínculos entre tablas relacionadas. Una foreign key establece una relación entre una o más columnas de una tabla y la clave primaria de la tabla referenciada. Como patrón para la nomenclatura de la foreign key elegimos el siguiente.

**FK\_<TABLA\_QUE\_REFERENCIA>+<CAMPO\_QUE\_REFERENCIA>\_<TABLA\_REFERENCIADA>+<CAMPO\_REFERENCIADO>**

Ej.: ***fk\_cliente\_id\_orden\_cliente\_id\_cliente***

### Indexes

Los índices son un mecanismo para aumentar la eficiencia de localización y acceso de un registro en una tabla en la base de datos, opcionalmente asegurando unicidad de los valores del índice. La definición de índices tiene un impacto positivo en los tiempos de consulta de registro y uno negativo en los de inserción y actualización de los campos del índice.

Los índices están asociados a una tabla y a un conjunto de campos de la tabla, a su vez pueden ser únicos o no y pueden estar definidos en cluster o no. La nomenclatura elegida para nombrarlos es la siguiente:

[IDX\_]<TABLA>\_<CAMPO>[\_AUX]

Prefijar el índice es opcional, pero de hacerlo se debe usar el prefijo especificado.

Ejemplo: idx\_orden\_detalle\_id\_orden\_u\_nc

El ejemplo corresponde a un índice definido sobre la tabla orden\_detalle, sobre el campo id\_orden, unico y nonclustered.

### Variables

Cuando las variables corresponden columnas de una tabla, deben ser nombrados de la misma manera que la columna. La notación elegida para definir las variables es camel.

# GUIAS

En esta sección se presentan guías para la definición de esquemas relacionales según un modelo conceptual.

## Relaciones 1:N

Para la definición de tablas que correspondan a una relación 1:N como puede ser la siguiente:

|  |  |
| --- | --- |
|  | **Una compañía emplea 1..N empleados** |

Se sugiere utilizar una estructura de tablas como la siguiente:

|  |  |
| --- | --- |
|  |  |

## Relaciones N:M

Para la definición de tablas que correspondan a relaciones N:M com puede ser la siguiente:

|  |  |
| --- | --- |
|  | **Un doctor atiende N pacientes, y un paciente es atendido por M doctores.** |

Se sugiere utilizar una estructura de tablas como la siguiente:

|  |  |
| --- | --- |
|  |  |

# BIBLIOGRAFÍA

1. Data Object Naming conventions by Kondreddi, Narayana Vyas (<http://vyaskn.tripod.com/object_naming.htm>)
2. Ten Things I hate about you by Celko, Joe (<http://www.intelligententerprise.com/001205/celko1_1.jhtml> )
3. Estándar de nomenclatura para bases de datos Oracle, Infocorp