BUENAS PRÁCTICAS

Manual para la aplicación de buenas prácticas de nomenclatura

Anexo II

Anexo II.- MANUAL DE BUENAS PRÁCTICAS EN LA GENERACIÓN DE LOS MODELOS DEL IEPNB

En este manual se proporcionan las normas básicas para la generación de los modelos de datos de los componentes del IEPNB.

Como es un proceso abierto, este manual se irá actualizando a medida que encontremos nuevas circunstancias.

Hay dos tipos de indicaciones:

- **Normas**: requisitos de obligado cumplimiento.
- **Buenas prácticas o recomendaciones**: no son de obligado cumplimiento aunque para un correcto funcionamiento se recomienda cumplirlas siempre que sea posible.

1. Normativa general

- No se pueden incluir objetos duplicados en el modelo de datos. Si se tienen varios objetos con el mismo nombre, al generar el script de creación, se intentaría crear dos veces el objeto y fallaría su ejecución.
- El tipo de base de datos para el que se han definido los objetos es Oracle.
- Los nombres de todos los objetos de base de datos (columnas, primary key, foreign key…) deben estar en mayúsculas, excepto las tablas y vistas, que se escribirán en minúsculas salvo la primera letra de cada palabra y sin espacios. Sólo pueden contener caracteres alfanuméricos no acentuados y guiones bajos. No pueden contener la letra 'ñ'.
- La *Integridad referencial* está basada en Primary Keys (PK) y Foreign Keys (FK) y siguiendo una nomenclatura específica para estas claves.
- Uso de catálogos comunes y acceso a otros esquemas: para no duplicar los datos se usarán los catálogos generales, así como los procedimientos y funciones de carácter genérico.

2. Buenas prácticas

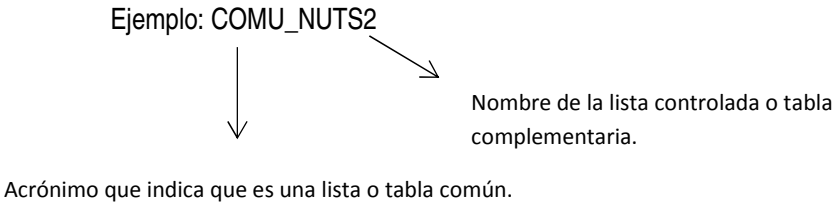
2.1. Nomenclatura

- Nombre del modelo de datos: IEPNB.
- Capas gráficas: YY_XXXX_(C,P)
  - YY es el número y letra del componente (3a, 6d, …)
  - XXXX es el acrónimo o sigla del componente (IFN, LUCDEME,…)
  - C: Islas Canarias; P: Península.
- Tablas:
  - Nomenclatura: (LC,TC)_XXXX_NombreTabla:
    - LC y TC son los acrónimos Lista Controlada o Tabla Complementaria respectivamente. Una LC puede estar abierta, si por ejemplo es una lista que se extrae para la normalización, o cerrada, si por ejemplo es una lista cuyo contenido se controlada desde la legislación).
    - XXXX es el acrónimo del componente.

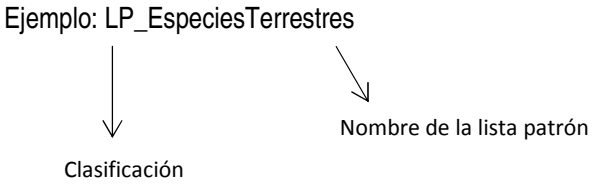
- NombreTabla es el nombre de la tabla en minúsculas, si se compone de varias palabras la primera letra de cada una irá en mayúsculas o si contiene varias palabras también podrían separarse por guiones bajos. Si se trata de un acrónimo o siglas irá en mayúsculas.

  Se utilizarán los acrónimos y siglas de los componentes del Inventario Español de Patrimonio Natural y Biodiversidad.

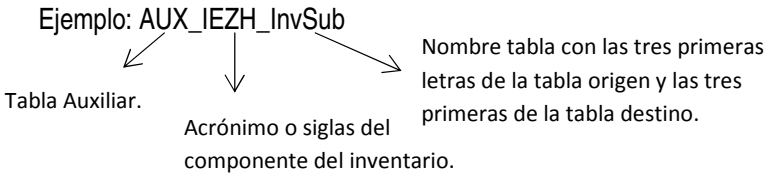
- Las tablas comunes a varios componentes se denominarán: COMU_NombreTabla. No se indicarán las siglas o acrónimo del nombre del componente ya que será común a varios.



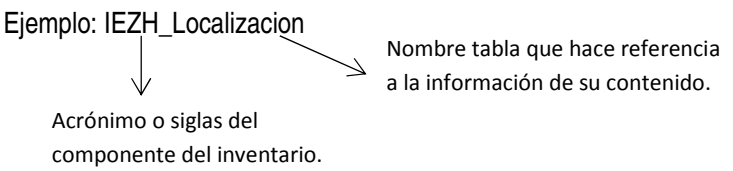
- Las Listas Patrón (LP) como son utilizadas por varios componentes se denominan: LP_NombreTabla. No se indicarán las siglas o acrónimo del nombre del componente ya que será común a varios.



- Las tablas auxiliares, necesarias para que el modelo funcione se denominarán: AUX_ACRONIMOCOMPONENTE_XXXSSS

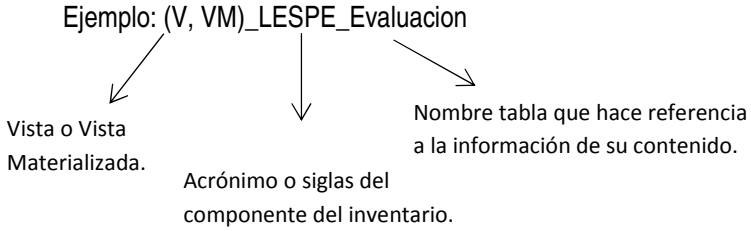


- Las tablas que se generen para simplificar la tabla principal del componente, es decir, para dividir de forma objetiva la información que contienen, se nombran con el acrónimo del componente seguido del nombre de la tabla en minúsculas con la primera/s letra/s en mayúsculas:

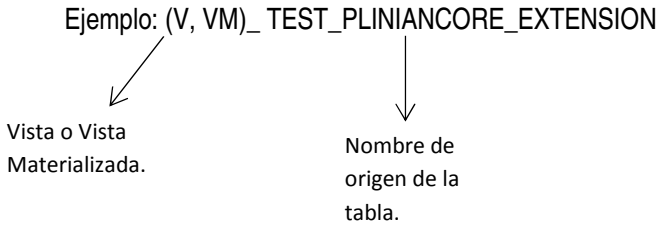


- Vistas: es un objeto de Oracle que al consultarlo ejecuta por detrás una query y nos devuelve el resultado (donde la query puede acceder a tablas, otras vistas, utilizar funciones o procedimientos, etc.). Cada vez que accedemos a la vista, la query se ejecuta y nos devuelve la información que en ese momento exista en el origen.
- Vistas Materializadas: es otro tipo de objeto de Oracle que aunque técnicamente es lo mismo que una *Vista* (ejecuta una consulta sql), lo que hace es almacenar físicamente en caché el resultado de ejecutar una query en un determinado momento, de forma que cada vez que consultemos la *Vista Materializada* lo que vamos a recuperar es lo que había en el origen en el momento en el que se creó o se refrescó la VM.

  - Nomenclatura: (V, VM)_XXX_NombreTabla:
    - V es la sigla de Vista; VM son las siglas de Vista Materializada
    - XXX es el acrónimo o sigla del componente.
    - NombreTabla es el nombre de la tabla en minúsculas, si se compone de varias palabras la primera letra de cada una irá en mayúsculas o si contiene varias palabras también podrían separarse por guiones bajos. Si se trata de un acrónimo o siglas irá en mayúsculas.



  - Las vistas que devuelven como resultado una tabla completa contienen el nombre de la tabla original. En nuestro caso como las vistas son sobre el modelo de datos de especies los nombres estarán en mayúsculas.



- En las tablas las columnas a crear deben ser de los siguientes tipos (Oracle):
  - BFILE: datos LOB sólo que los datos se almacenan en un archivo físico en el sistema operativo en lugar de en el servidor. Este tipo de datos proporciona acceso de sólo lectura a los datos. El tamaño máximo en BD es 4 Gigabytes.
  - BLOB: datos para guardar documentos-objetos binarios de longitud variable, para cadenas de más de 4000 gigabytes, es una versión más grande que RAW.
  - CHAR(n) y NCHAR(n): datos de cadena de longitud fija. *n* define la longitud de la cadena y debe ser un valor entre 1 y 8.000. Si no se especifica *n* Oracle le da un tamaño de 255 bytes.
  - **CLOB**: tipo de datos para almacenar bloques de texto ASCII, incluido el texto formateado como HTML o PostScript. Cadena de caracteres de longitud variable. Es una versión más grande de VARCHAR2. El tamaño máximo en BD es 4 Gigabytes.
  - **DATE**: almacena un valor para fecha y hora.
  - **FLOAT(x):** almacena un número en punto decimal sin restricción de dígitos decimales.

    Una columna FLOAT(126) es equivalente a NUMBER(38) aunque Number no podrá contener decimales y float si y con cualquier escala.
  - LONG: obsoleto. Almacena caracteres de longitud variable hasta 2gb, ahora se usa CLOB y NLOB para almacenar grandes cantidades de datos alfanuméricos.
  - LONG RAW: almacena cadenas binarias de ancho variable de hasta 2 gigabytes. Se recomienda su conversión en BLOB.
  - NCLOB: como CLOB pero este tipo proporciona además una forma de usar un NLS alternativo de la base de datos.
  - **NUMBER(p,s):** valores numéricos enteros o decimales, positivos o negativos. 'p' es el número total de dígitos, contando los decimales (de 1 a 38) y 'd' el número máximo de dígitos decimales (escala).
  - NVARCHAR y NVARCHAR2: máxima longitud permitida: 4000 caracteres. VARCHAR(2) Y NVARCHAR(30) son cadenas de longitud variable cuyo tamaño máximo es lo que está entre paréntesis.

    (Si se define una cadena de longitud 100 bytes y se introduce un valor de 10 bytes, la columna ocupará 10 y no 100 como haría con el char).
  - RAW(n): almacena cadenas binarias de ancho variable de hasta 2000 bytes. Oracle recomienda convertirlas en BLOB. Es obligatorio especificar el tamaño.
  - ROWID: almacena cadenas hexadecimales que representan de forma única una fila en una table (pero no única en cualquier tabla).
  - VARCHAR y **VARCHAR2**: [n|max] cuando los tamaños de las entradas de datos de columna varían de forma considerable. Datos de cadena no Unicode de longitud variable.

Máxima longitud permitida: 8000 caracteres.

Usando VARCHAR2 en lugar de CHAR ahorramos espacio de almacenamiento.

- Se recomienda utilizar la siguiente nomenclatura según el tipo de dato que representa la columna. Contiene la nomenclatura más utilizada en los modelos:
  - ID_: para el identificador del elemento, es normalmente la PK. Ejemplo: ID_USUARIO.

    También se usará ID para nombrar los identificadores de las listas controladas generadas para el correcto funcionamiento del modelo.
  - CD_: para códigos alfanuméricos. Ejemplo: CD_DNI.

    Se usará CD cuando el identificador de la lista controlada ya haya sido generado previamente.
    - CD_MUNI: Código del municipio
    - CD_NUTS1: Código de los NUTS1. Agrupación de comunidades Autónomas
    - CD_NUTS2: Código de los NUTS2. Comunidades y ciudades Autónomas
    - CD_NUTS3: Código de los NUTS3. Provincias, islas, Ceuta y Melilla
    - CD_TLF: Número de teléfono
    - CD_FAX: Número de fax
    - CD_TIPO: Código del tipo
    - CD_SIST_REF: Sistema de referencia
  - DS_: para descripciones alfanuméricas. Ejemplo: DS_EXPEDIENTE_ECONOMICO. (Describe un CD)
    - DS_TIPOLOGIA: Tipologías existentes. Nombre o descripción de un CD
    - DS_EST_CONSERV: Estado de conservación. Nombre o descripción de un CD
  - FC_: para fechas. Ejemplo: FC_ALTA_INVENTARIO.
    - FC_ENTRADA: Fecha de entrada
    - FC_SALIDA: Fecha de salida
    - FC_MODIF: Fecha de modificación
    - FC_ACT: Fecha actualización
  - BO_: para marcas, indicadores (Booleano). Ejemplo: BO_MASCULINO y se indica 'S' o 'N'. Utilizamos el campo VARCHAR2 (1).

- NM_: valor numérico o contador. Ejemplo: NM_PERSONAS_FISICAS.
    - NM_CP: Número del código costal
    - NM_HUSO: Huso en la malla de proyección UTM
    - NM_COORD_X: Coordenadas X
    - NM_COORD_Y: Coordenadas Y
    - NM_LAT_GRA: Latitud en grados
    - NM_LAT_MIN: Latitud en minutos
    - NM _LAT_SEG: Latitud en segundos
    - NM_LONG_GRA: Longitud en grados
    - NM_LONG_MIN: Longitud en minutos
    - NM_LONG_SEG: Longitud en segundos
    - NM_ALTITUD
    - NM_SUP: Extensión de la superficie
    - NM_DIR: Número de la dirección

- NB_: para nombres.
    - NB_CONTACTO: Nombre de contacto
    - NB_TIPO: Nombre del tipo
    - NB_DIR: Dirección (Vía, Calle, Avenida, Carretera,…)
    - NB_DPTO: Nombre de departamento

- TL_: (Texto largo) texto libre, observaciones, comentarios, etc. Ejemplo: TL_ MOTIVO_RECURSO.
    - TL_OBSERV: Observaciones

- TC_: (Texto corto) una palabra, un título, una frase corta, etc.
    - TC_ORIGEN
    - TC_NORMA_DECL: Norma de la declaración

- CL_: Texto libre de gran extensión. Equivale a un campo CLOB
- DR_: Se utiliza para direcciones de EMAIL o WEB
    - DR_WEB: Dirección Web
    - DR_EMAIL: Dirección de correo electrónico

- HR_: para tiempo sin fecha (hora, minuto, segundo).
- AN_: para años.
- EU_: para importe en euros.
- Además, si el nombre del campo contiene varias palabras pueden separarse por guiones bajos.
- Los valores numéricos tienen como separador de miles la ‘,’ y como separador de decimales el ‘.’.

- La nomenclatura de la *Foreign Key* debe ser de la forma FK_NombreTabla_NN, donde NN es un número consecutivo que se asigna a cada Foreign Key. (Se numera dentro de la tabla)
- No se han añadido a las tablas Checks, Valores por defecto y Triggers, en el caso de ser añadidos se deberá seguir la siguiente nomenclatura:
  - *Checks:* XXXX_CHECK_NOMBRECHECK.
  - *Valor por defecto*: XXXX_DF_NOMBRE_DEFAULT_VALUE.
  - *Triggers*: XXXX_NOMBRE_TABLA_TRIG_TIEMPO_ACCION (TIEMPO es B para Before o A para after) (ACCION es D para Delete, I para Insert o U para Update).

2.2. Generales

- Cada uno de los objetos (tabla, code list, feature class, etc.) debe tener una descripción del objeto al que hace referencia.
- Se pueden generar áreas de trabajo parciales con algunos de los objetos del esquema. Estas áreas son meramente informativas (para organizar visualmente los objetos) y no serán tenidas en cuenta para la generación de los scripts de base de datos. Esto se hace por ejemplo cuando el modelo de datos es grande (ej.: Plinian).
- Los objetos de un modelo deben estar dispuestos sobre el tapiz de diseño de forma clara, sin superponer unos a otros, de forma que puedan apreciarse visualmente todos los objetos del modelo así como sus relaciones.
- Si nuestro modelo de datos incluye tablas de otros esquemas que no tienen definida la Primary Key hay que deschequear la opción de generar las Foreign Keys a estas tablas para evitar que se generen.

2.3. Otros

- Particiones de tablas: por motivos de volumen se pueden particionar por rango (rango de fechas por ejemplo).
- Vistas: se recomienda crearlas incluyendo en la misma todos los campos de las claves primarias de cada una de las tablas que formen la vista de forma que la clave primaria de la vista será el subconjunto mínimo de claves primarias de las tablas que forman la vista o todas. En vistas con relaciones 1 a N bastará con incluir como identificador de la vista el identificador de la tabla principal.
- Si hay procedimientos y funciones se agruparán en packages.
- No se permite incluir en ningún paquete, procedimiento o función instrucciones del tipo COMMIT o ROLLBACK.
- Hay que hacer una correcta gestión de errores en los paquetes, procedimientos y funciones, asegurando que existe control sobre las posibles excepciones que puedan ocurrir durante su ejecución.