ESTANDARES DE NOTACION

|  |
| --- |
| **ESTANDARES DE DISEÑO DE BASE DE DATOS**  Reglas generales  Los nombres de tablas y campos deben especificarse bajo el estándar camelCase, este estándar especifica escribir palabras compuestas eliminando los espacios poniendo mayúscula en la primera letra de cada palabra.  Unicamente se utilizaran caracteres alfabéticos, salvo que por naturaleza del nombre s e necesite dígitos numéricos, se prohíbe el uso de caracteres de puntuación y símbolos:  Ej. nombreConferencista  Las letras acentuadas se reemplazaran con las equivalentes no aceptadas, en lugar de la letra ñ se utilizara ni:  Ej anioConferencia  El nombre debe ser lo más descriptivo posible, evitando términos ambiguos o que presenten distintos interpretaciones:  Ej. categorisConferencia => tipoConferencia  El nombre no debe abreviarse de manera entendible y específica con al menos las primeras tres letras de la palabra.  Tablas  Los nombres de las tablas deben especificarse de forma plural y de acuerdo a las reglas generales:  Ej. Estudiantes, conferencistas, cursos, etc.  Todas las tablas de las BD deben relacionarse con al menos una tabla.  Campos clave (Atributos)  Toda tabla debe poseer una o más llaves referenciales:  Ej. Tabla Estudiantes (codEst).  Toda relación entre tablas debe implementarse mediante constraints (Llaves foraneas) con integridad referencial.  La integridad referencial deberá actualizar en cascada en todos los casos y restringir el borrado, salvo para las entidades débiles.  *Ej. No se podrá eliminar un registro de la tabla estudiantes que tenga ocurrencias en otras tablas, si sucediera el caso deberá implementarse el borrado lógico, por el contrario si podrá habilitarse el borrado en cascada, si la relación fuera (tablas estudiantes conferencista).*  Las llaves referenciales deben ubicarse al inicio de la definición de la tabla (Los primeros).  El nombre de la llave referencial debe estar compuesto por: “id + nombre de la tabla en singular”:  Ej. Tabla conferencista => idConf  Otros campos  Todo campo o tributo que presente un nombre o descripción se colocará inmediatamente después de las llaves referenciales:  Ej. Tabla Conferencistas => idConf, **nomConf**  En caso de los campos que presenten datos de acuerdo a su representación conceptual en el ámbito del negocio deberá prefijarse de la siguiente manera:  Numeros: **num** (Ejemplo: numeroEstudiantes => **numEst**)  Fechas: **fecha** (Ejemplo: fecha de incripcion => **fechaInsc**)  Codigos: **Codigo** (Ejemplo: Código de estudiante => codEst)  Los campos de relación (Llaves foráneas) deben nombrarse de la misma manera que las llaves primarias (Usando el nombre de la tabla que hace referencia). |
| **ESTANDARES DE CODIFICACIÓN**  Declaración de variables:   * Las variables generales deben ser descriptivos, y los locales no necesariamente descriptivos pero si cortos. * Los nombres de las constantes de las clases deberían escribirse todo en mayúscula con las palabras separadas por un guion bajo (“\_”). * En el caso de las variables generales, deben ser encapsulados. * En la declaración de variables se manejaran el método camelCase, que implica declarar una variable de la siguiente manera. El inicio del nombre de la variable debe ser en minúscula, y en caso de tener alguna otra palabra.   Nombre de los métodos:   * Los métodos deben ser encapsulados. * Los métodos deberán ser verbos (en infinitivo), en mayúsculas y minúsculas, con las primeras letras del nombre en minúscula, y con las primeras letras de cada palabra interna en mayúscula (lowerCamelCase). * No se permiten caracteres especiales. * El nombre debe ser lo suficiente descriptivo, no importando a priori la longitud del mismo.   Comentarios:   * Los comentarios deben hacerse al inicio de cada clase, antes de cada método de la siguiente manera:   /\*Nombre del método (Descripción)  \* Parámetros que recibe (Descripción)  \* Dato de retorno (Descripción)  \*/   * Los comentarios más cortos se las debe realizar en una sola línea (“//”) |
| **ESTANDARES DEL VERSIONADOR**  GIT HUB   * Todo el proyecto se trabajará solamente en una rama. * Las sugerencias dentro el desarrollo deberán se podrán hacer en una rama interna. * La descripción de los commit’s deben ser especificas con detalle de los cambios realizados. |
|  |