**Padrão para Nomenclatura de Banco de Dados­­­**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sumário de Informações do Documento** | | | |
| Versão | Data | Mudanças | Autor |
| 0.1 | 30/05/2013 | Adaptação da Metodologia utilizada pela Celepar Informática do Paraná. | Roberto Soares Bochnia |
| 0.2 | 02/06/2013 | Alterada nomenclatura para minúscula no início do nome. | Roberto Soares Bochnia |
| 0.3 | 03/06/2013 | Alterada nomenclatura de tabela excluindo a sigla tb\_ no início do nome. | Roberto Soares Bochnia |
| 0.4 | 04/06/2013 | Alterada nomenclatura dos nomes para tudo em minúsculo. | Roberto Soares Bochnia |

Sumário

[1. INTRODUÇÃO 2](#__RefHeading__870_834607251)

[2. BANCO DE DADOS 2](#__RefHeading__872_834607251)

[3. CHAVE DO USUÁRIO DO BANCO DE DADOS 2](#__RefHeading__874_834607251)

[4. TABELAS 3](#__RefHeading__876_834607251)

[5. NOME DOS ATRIBUTOS (COLUNAS) 5](#__RefHeading__878_834607251)

[6. VIEWS 8](#__RefHeading__880_834607251)

[7. ÍNDICES 9](#__RefHeading__882_834607251)

[7.1 Chaves primárias 10](#__RefHeading__884_834607251)

[7.2 Chaves estrangeiras 10](#__RefHeading__886_834607251)

[8. STORED PROCEDURES 11](#__RefHeading__888_834607251)

[9. TRIGGER 13](#__RefHeading__890_834607251)

# 1. INTRODUÇÃO

O objetivo deste documento é prover informações para formalizar a nomenclatura de objetos de banco de dados, bem como apresentar regras para sua utilização, evitando assim, o hábito de existir diferentes nomenclaturas dentro de uma mesma aplicação.

O documento apresenta os objetos de banco de dados com três itens: sintaxe, regras e exemplo. Isto para facilitar o entendimento do desenvolvedor.

# 2. BANCO DE DADOS

O nome do banco de dados deverá identificar o negócio que está sendo automatizado ou deverá refletir a sigla da aplicação.

**Sintaxe:**

{[a.z]} -> xxxxxx , onde:

xxxxxx - indica o nome da aplicação que o banco de dados irá atender.

Para o nome da aplicação utilizar tudo em minúsculo com no máximo 15 caracteres.

# 3. CHAVE DO USUÁRIO DO BANCO DE DADOS

A chave do usuário do banco de dados deverá refletir o nome do banco de dados utilizado.

**Sintaxe:**

sa\_{[a.z]} -> xxxxxxx , onde:

sa\_ - indica que é uma chave do banco de dados

xxxxxx – indica o nome da aplicação.

Para o nome da aplicação utilizar tudo em minúsculo com no máximo 15 caracteres.

# 4. TABELAS

O nome de uma tabela deverá ser sugestivo, por exemplo: se no departamento Financeiro for necessário manter uma tabela de feriados, esta tabela deve ser nomeada identificando claramente seu conteúdo, isto significa que seu nome então deverá ser feriado.

**Sintaxe:**

{[A..Z][{a.z}]} xxxxXxxxx, onde:

xxxxxxx – indica o nome da tabela.

**Regras:**

· Tudo em minúsculo;

· Limite-se a usar 30 caracteres - padrão da maioria dos bancos. Em caso de dúvida, consulte esta informação no manual do banco de dados que estiver utilizando;

· O nome da tabela deve estar sempre no singular (Exemplo: *Cliente* no lugar de *Clientes*);

· Evite usar abreviações, se necessário use as conhecidas;

· Não utilize acentuações ou caracteres especiais;

**Exemplos:**

***pessoa*** - Tabela de pessoas;

***funcionario*** - Tabela de funcionários;

***cliente*** - Tabela de clientes;

***feriado*** - Tabela de feriados;

***laudo\_necroterio*** - Tabela de laudos do necrotério.

# 5. NOME DOS ATRIBUTOS (COLUNAS)

Seguindo o mesmo padrão utilizado para tabelas, deve-se identificar a coluna da tabela de maneira clara e descritiva, somente use abreviações quando se tratar de domínio público, exemplo FGTS, fundo de garantia por tempo de serviço.

**Sintaxe:**

{ [A..Z][{a.z}] }  xxxxxxxxx, onde

xxxxx – indica o nome do campo

**Regras:**

· Tudo em minúsculo;

· Não usar preposições;

· Usar palavras no singular e sem acentuação;

· Usar nome que identifique e individualize o dado dentro do “ambiente” do cliente;

· Dar nomes distintos para dados distintos;

· Utilizar o nome sempre no singular;

· Evite utilizar abreviações, caso seja extremamente necessário utilize as de domínio público ou limite-se a 4 letras;

**Exemplos:**

Sem abreviações:

***nome***

***nome\_funcionario***

***data\_nascimento***

***valor\_salario***

Com abreviações***:***

· ***nome\_func\_resp***

· ***gestor\_resp***

· ***cod\_termo***

**Nota 1:** Para campos de chave primária utilizar a seguinte regra:

Identificador-da-chave[nome da tabela]

Exemplos:

Nome da Tabela Identificador

id\_pessoa

id\_cliente

id\_historico

**Nota 2:** Caso a chave seja composta utilize um nome “simples” que identifique a composição, evitando usar abreviações.

**Nota 3:** Nomenclatura -

Um aspecto importante quanto a **nomenclatura dos dados**, é que a regra para codificação dos nomes seja tal que, a partir do significado dos dados, se possa chegar ao seu nome. Este aspecto é importante para que se possa aproveitar o potencial dos dados armazenados, através da identificação da existência ou não dos dados necessários à uma aplicação, evitando-se a criação de redundâncias desnecessárias.

Uma estratégia que utilizada para este fim, é ter uma regra para formação de nome:

Formato geral de um nome de dado

**TIPO-QUALIFICADO 1 .**.. **QUALIFICADOR N -IDENTIFICADOR**

Onde:

**TIPO-QUALIFICADO:** identifica o tipo de dado. É o termo mais geral no nome.

Exemplos: código, data, nome, valor.

**QUALIFICADORES:** especificam detalhes que qualificam o tipo de dado. Não são obrigatórios. Quando utilizados, devem ser codificados do mais geral para o mais especifico.

Exemplo: provento, nascimento, salario.

**IDENTIFICADOR:** identifica a entidade ou relacionamento que possui o dado como um de seus atributos.

Exemplos: funcionários, consumidor, fornecedor, departamento.

**Exemplo de Nomes:**

código\_provento\_funcionario

data\_nascimento\_consumidor

nome\_filho\_funcionario

nome\_consumidor

valor\_salario\_funcionario

Existe uma forma de criar nome de campos:

- nos bancos de dados onde houver diferenciação de letras maiúsculas e minúsculas usar a notação Húngara, exceto para algumas siglas que ficam em maiúscula. Exemplo: códigoProventoFuncionario, numeroCGC.

- Para os bancos de dados onde não se consiga usar este tipo de notação, usar letras minúsculas separando os elementos do nome através de underscore “\_” ou hífen “-”. Exemplo:

código\_provento\_funcionario.

Em todos os casos, quando criar as bases de dados, não usar acentuação mesmo que o banco de dados permita, pois torna difícil a portabilidade para outra plataforma.

**Notação Húngara** é a diferenciação dos elementos de uma dada nomenclatura através do uso de letras maiúsculas e letras minúsculas. Tem a sua origem na codificação de programas.

# 6. VIEWS

Deve-se utilizar a mesma semântica utilizada para as tabelas. Deve ser prefixada a palavra “vw” seguido do nome da tabela.

**Sintaxe:**

vw\_{ [A..Z][{a.z}] } -> vw\_xxxxxxxxxxxxxx, onde:

vw\_ - indica que é uma view

xxxxxx – indica o nome da view

**Regras:**

· Tudo em minúsculo;

· Limite-se a usar 30 caracteres - padrão da maioria dos bancos. Em caso de dúvida consulte esta informação no manual do banco de dados que estiver utilizando;

· O nome da visão (*view)* deve estar sempre no singular;

· Evite usar abreviações, se necessário use as conhecidas;

· Não utilize acentuações ou caracteres especiais.

**Exemplos:**

***vw\_pessoa*** *- View de pessoas.*

***vw\_funcionario*** *- View de funcionários.*

***vw\_cliente*** *- View de clientes.*

***vw\_feriado*** *- View de feriados.*

***vw\_laudo\_iml*** *- View de laudos do IML*

# 7. ÍNDICES

Deve-se utilizar a mesma semântica utilizada para as tabelas.

**Sintaxe:**

ix\_{ [A..Z][{a.z}] } {[1..99] | [identificador da chave]} -> ix\_xxxxxxxxxxxxx, ou seja,

ix\_<nome\_da\_tabela><identificador\_da\_chave>nn, onde:

ix\_ - prefixo para identificar que se trata de um índice

xxxxxxx - nome da tabela a qual o índice está associado

nn – número seqüencial da tabela

xxxxxxx - número seqüencial dos índices que se referem a tabela ou identificador da chave

## 7.1 Chaves primárias

Para nomear as chaves primárias utilizar:

**Sintaxe**:

pk\_{[A..Z][{a..z}]} -> pk\_xxxxxx, onde

pk\_ - indica que é uma chave primária

xxxxx – indica o nome da tabela a qual o índice está associado

## 7.2 Chaves estrangeiras

Para nomear as chaves estrangeiras utilizar:

**Sintaxe**:

fk\_{[A..Z][{a..z}]}\_{[A..Z][{a..z}]} -> fk\_xxxxxx\_xxxxx, ou seja,

fk\_<nome da tabela origem(pai)>\_<nome da tabela destino(filho)>, onde:

fk\_ - indica que se trata de uma chave estrangeira

xxxxx - indica o nome da tabela origem(pai)

xxxxx - indica o nome da tabela destino(filho)

**Regras:**

· Tudo em minúsculo;

· Limite-se a usar 15 caracteres - Em caso de dúvida consulte esta informação no manual do banco de dados que estiver utilizando;

· O nome da tabela deve estar sempre no singular. Evite usar abreviações, se necessário use as conhecidas;

· Não utilize acentuações ou caracteres especiais.

# 8. STORED PROCEDURES

Deve-se utilizar a mesma semântica utilizada para as tabelas. Deve ser prefixada com a sigla “stp\_” seguida do nome da tabela para a *stored procedure*. Para *stored procedures* especifica como inclusão (i), exclusão (d), alteração (u) e consulta (s) utilizar a notação especifica, apresentada na sintaxe.

**Sintaxe:**

stp\_{[ d | i | s | u ]}\_{[A..Z][{a..z}]} -> stp\_x\_xxxxxx, ou seja,

stp\_<identificador\_da\_ação\_principal><nome\_da\_stored\_procedure>, onde:

stp – indica que se trata de uma *stored procedure*

x - Identificador-da-Ação Principal -

d – DELETE, i – INSERT, s – SELECT e u – UPDATE , quando houver operações compostas utilizar o identificador em ordem alfabética.

xxxxxx – nome da *stored procedure*

**Regras:**

· Tudo em minúsculo;

· O nome da *stored procedure* deve estar sempre no singular. Evite usar abreviações, se necessário use as conhecidas;

· Não utilize acentuações ou caracteres especiais;

· Definir uma área de identificação geral, onde deverão existir informações tais como:

· Autor;

· Função;

· Alterações;

· Data;

· Autor;

· Descrição da alteração.

· Identificar as alterações efetuadas na área apropriada, para permitir fácil localização de problemas no código;

· Endentar de modo a tornar a codificação clara e facilitar o trabalho de manutenção;

· Inserir comentários sempre que necessário. Comentários adicionais que auxiliem a compreensão de processamentos complexos. Não poluir o código com comentários desnecessários, que descrevam procedimentos óbvios;

· Evitar o aninhamento excessivo de comandos, o que costuma dificultar a manutenção do código. Dar preferência à codificação mais longa, porém mais clara, desde que não prejudique a performance.

**Exemplos:**

stp\_i\_[nome-da-sp] - Stored procedure de inclusão.

stp\_d\_[nome-da-sp] - Stored procedure de exclusão.

stp\_u\_[nome-da-sp] - Stored procedure de alteração.

stp\_s[nome-da-sp] - Stored procedure de consulta.

# 9. TRIGGER

Deve-se utilizar a mesma semântica utilizada para as tabelas. Deve ser prefixada a sigla “trg\_” seguida do nome da tabela para a trigger. Para trigger especifica como inclusão (i), exclusão (d), alteração (u) e consulta (s) utilizar a notação especifica, apresentada na sintaxe.

**Sintaxe:**

trg\_{[ d | i | u ]}\_{[A..Z][{a..z}]} -> trg\_x\_xxxxxx, ou seja,

trg\_<identificador\_da\_ação\_principal><nome\_da\_trigger>, onde:

trg\_ - indica que se trata de uma trigger

x - Identificador-da-ação\_principal -

d – DELETE, i – INSERT e u – UPDATE, quando houver operações compostas utilizar o identificador em ordem alfabética.

xxxxxx – nome da trigger

**Regras:**

· Tudo em minúsculo;

· O nome da trigger deve estar sempre no singular. Evite usar abreviações, se necessário use as conhecidas;

· Não utilize acentuações ou caracteres especiais;

· Definir uma área de identificação geral, onde deverão existir informações tais como:

· Autor;

· Função;

· Alterações;

· Data;

· Autor;

· Descrição da alteração.

· Identificar as alterações efetuadas na área apropriada, para permitir fácil localização de problemas no código;

· Endentar de modo a tornar a codificação clara e facilitar o trabalho de manutenção;

· Inserir comentários sempre que necessário. Comentários adicionais que auxiliem a compreensão de processamentos complexos. Não poluir o código com comentários desnecessários, que descrevam procedimentos óbvios;

· Evitar o aninhamento excessivo de comandos, o que costuma dificultar a manutenção do código. Dar preferência à codificação mais longa, porém mais clara, desde que não prejudique a performance;

**Exemplos:**

trg\_i\_[nome-da-trg] - Trigger de inclusão.

trg\_d\_[nome-da-trg] - Trigger de exclusão.

trg\_u\_[nome-da-trg] - Trigger de alteração.