Secretaria da Fazenda do Estado do Tocantins

Centro Interamericano de Administrações Tributárias

Produto: Padrão para o Banco de Dados

**Setembro/2015**

Tabela de conteúdo

1. Introdução 3

2. Schema 3

3. Nomenclatura de objetos 4

4. Modelo Lógico 6

5. Documentação 7

# Introdução

Esse documento descreve um conjunto de normas, convenções e recomendações para o desenvolvimento de aplicações para a SEFAZ-TO ao nível de banco de dados.

O objetivo fundamental é o de proporcionar um padrão que será usado para todo tipo de desenvolvimento realizado pelo pessoal interno ou externo na SEFAZ-TO, e será incorporado como um padrão para a aceitação ou rejeição dos resultados finais produzidos durante um projeto de software.

É vital destacar a importância de manter uma constante atualização dessas diretrizes, seguindo os desenvolvimentos tecnológicos e o surgimento de novos paradigmas no campo da informática.

O seguimento de padrões de codificação é importante porque levam a uma maior coerência dentro do código na equipe de trabalho. Maior consistência leva que o código seja mais fácil de entender, isso significa maior capacidade de desenvolver e manter. Isto reduz o custo total das aplicações criadas. Lembre-se que o código de programação vai existir muito tempo depois que o equipe de programação seja transferido para outros projetos.

Uma meta importante durante o desenvolvimento é garantir que qualquer programador pode transferir seu trabalho para outro desenvolvedor ou outra equipe de desenvolvimento, para que possam realizar a manutenção e melhorias para o sistema, sem ter que investir um tempo razoável para entender o código.

O código que é difícil de entender corre o risco de ser rejeitado e reescrito. Se cada desenvolvedor segue seus próprios critérios, então é muito difícil compartilhar código entre desenvolvedores, elevando o custo de desenvolvimento e manutenção.

# Schema

Um schema é uma coleção de objetos de banco de dados normalmente associada a um usuário de um banco de dados em particular. Este usuário é mais conhecido como dono do schema

Os nomes dos schemas devem ser representativos da aplicação (ou módulo) representado. As abreviaturas mostradas na seguinte tabela:

|  |  |
| --- | --- |
| **ESQUEMA** | **DESCRIPÇÃO** |
| SEFAZ\_ARR | Arrecadação |
| SEFAZ\_CAT | Catálogos do Sistema |
| SEFAZ\_CCI | Cadastro de Contribuinte |
| SEFAZ\_CCC | Conta Corrente |
| SEFAZ\_COB | Cobrança |
| SEFAZ\_COM | Consulta |
| SEFAZ\_ECF | Equipamento Emissor de Cupom Fiscal |
| SEFAZ\_FIE | Fiscalização Estabelecimento |
| SEFAZ\_FTR | Fiscalização de Trânsito de Mercadorias |
| SEFAZ\_GED | Gerenciamento Eletrônico de Documentos |
| SEFAZ\_ISE | Isenções |
| SEFAZ\_NFE | NFE |
| SEFAZ\_OTR | Outras Receitas |
| SEFAZ\_REE | Regimes Especiais |
| SEFAZ\_RES | Restituições |
| SEFAZ\_SEG | Segurança |
| SEFAZ\_WRK | Gestão de Fluxo de Trabalho |

# Nomenclatura de objetos

Nas primeiras linguagens de programação como Basic, Pascal, Cobol, Fortran, etc ... o uso de memória para executar o programa foi muito importante. Em seguida, uma das opções era reduzir o uso de memória no armazenamento de programas. Foram usadas abreviaturas nos nomes para descrever campos. Por exemplo: data de nascimento, foi DT\_NASC. Existem muitos sistemas legados com estas nomenclaturas e até hoje há desenvolvedores que continuam usando esses mesmos padrões legados nos programas como no banco de dados.

Atualmente, não há tantas restrições sobre o uso de memória, como existem grandes servidores e recursos de computação. Então, a recomendação é usar a melhor descrição para um campo para a de legibilidade e manutenção dos programas. Assim, para o exemplo: data de nascimento será DATA\_NASCIMENTO. Outro exemplo muito exagerado pode ser DATA\_DE\_NASCIMENTO\_PESSOA\_FISI (não é recomendável).

Para criar um objeto no banco de dados deve levar em consideração a seguinte estrutura do nome:

1.- Os primeiros caracteres para definir o objeto, você pode levar em consideração as seguintes abreviaturas:

|  |  |
| --- | --- |
| **OBJETO** | **ABREVIATURA** |
| TABELA | TA |
| VIEW (VISÃO) | VW |
| ÍNDICE | IX |
| TRIGGER | TR |
| SEQUÊNCIA | SQ |
| CONSTRAINT | PK,FK,NN,UN,CK |
| PACOTE | PKG |
| PROCEDURE | PRC |
| FUNCION | FNC |
| TRABALHO | JB |
| ROLES | RL |
| TABLESPACES | TBS\_XXX\_DADOS; TBS\_XXX\_INDICES |

2.- Os próximos caracteres que representam o esquema ao qual pertence o objeto, como para a criação de esquemas é uma boa alternativa para um esquema é gerado para cada aplicação é feita com base nos desta forma vai nos dar uma melhor organização. Exemplo: SEFAZ\_ARR ... teria como esquema de ARR.

3.- Os outros caracteres que compõem o nome do objeto devem ser somente alfabéticos, ou seja, não utilizar números em seus nomes, ou quaisquer outros caracteres especiais, exceto o caso de sublinhado “underscore” (\_).

4.- O nome de uma Tabela deve ser alusivo à natureza dos dados que serão armazenados nela. O nome da tabela deve ser feita em singular.

5.- Os nomes dos índices são assim: IX\_<NOME\_TABELA>#

6.- O nome de uma trigger deve ter o seguinte formato. Onde: ocasião\_disparo pode ser PRE/POS e operações pode ser INS/UPD/DEL/AUD.

TR\_<nome\_da\_tabela>\_<ocasião\_disparo>\_<operações>

7.- É necessário para a formação dos nomes são feitas apenas letras, exceto no caso de índices, que também pode usar números

8.- Oracle suporta até 30 caracteres para os nomes de esquemas, tabelas, objetos, colunas (campos).

Alguns exemplos:

|  |  |  |
| --- | --- | --- |
| **Exemplo** | **Válido** | **Inválido** |
| Data de Nascimento | DATA\_NASCIMENTO | DT\_NASC, DT\_NS, DT\_NASCI |
| Código País | CODIGO\_PAIS | CO\_PA, COD\_PAIS |
| Zona de Vigilância Aduaneira | ZONA\_VIGILANCIA\_ADUANEIRA | ZO\_VIG\_ADUANEIRA |
| Documento Tributário | DOCUMENTO\_TRIBUTARIO | DOC\_TRIB |
| Valor de Imposto | VALOR\_IMPOSTO | VLR\_IMPO |
| Obrigação tributária | OBLIGACAO\_TRIBUTARIA | OBLIG\_TRIB, OB\_TRIBUTA |

9.- Exemplo para View. Precisa-se uma View para as declarações de ICMS del 2015. Entaõ será VW\_DECLARAÇÃO\_ICMS\_2015.

# Modelo Lógico

1.- Cada campo deve ser representado pelo tipo de dados que corresponde, isto é:

* Quando a informação é alfanumérica será representada por VARCHAR2.
* Quando são datas será representado por o tipo DATE.
* Quando são números com que as operações matemáticas são realizadas será usado o tipo NUMBER de outro modo será VARCHAR2.
* Poder ser usado números para representar sequencias.

2.- Descrever os domínios de cada campo com seu valor default.

3.- Se um campo ocorre em mais do que uma tabela para representar a integridade referencial, o nome de campo que deve ser o mesmo em cada tabela representa o relacionamento.

4.- Evitar padronização (transformação) das informações, especialmente nos comandos que fazem parte dos procedimentos armazenados (NVL, NVL2, TO\_CHAR, TO\_NUMBER).

5 .- Utilização mais eficiente dos predicados. Por exemplo: BETWEEN … AND … no lugar de … >= ... AND ... < ...

6.- Uso de valores default 0 quando o tipo de dados é número servindo em operações matemáticas, no contrário é NULL.

7.- Em vez de usar default '' em campos utilizar NULL, as pesquisas são melhores.

8.- Sempre utilizar os domínios nos campos; sejam conjuntos de valores ou condições.

9.- Para os nomes compostos utilizar (\_), exemplo: Valor de Imposto ... VALOR\_IMPOSTO.

10.- Utilizar singular para os nomes, não plural, exemplo de tabela: DECLARAÇÃO\_TRIBUTARIA é melhor que DECLARAÇÕES\_TRIBUTARIAS.

# Documentação

Cada objeto no banco de dados deve ser documentado, seja nas descrições do modelador E-R ou no documento Word. Também devem ser descritos no banco de dados de meta-dados, para Oracle utilizar o comando COMMENT.