Nomenclatura BDDAD Projeto Integrador

Conteúdo

[Nível Conceptual 2](#_Toc120211481)

[Entidade 2](#_Toc120211482)

[Atributo 2](#_Toc120211483)

[Relação 3](#_Toc120211484)

[Nível Lógico 3](#_Toc120211485)

[Tabelas 3](#_Toc120211486)

[Nível Físico 4](#_Toc120211487)

[Regras 4](#_Toc120211488)

[Geral 4](#_Toc120211489)

[Criação de Tabelas 4](#_Toc120211490)

[Queries/Procedures/Functions 5](#_Toc120211491)

Existem 3 fases pelas quais temos de passar para construir uma base de dados

* Nível **conceptual**
* Nível **lógico**
* Nível **físico**

Para cada nível existem regras que devem ser seguidas, tanto a nível de execução como nomenclatura.

# Nível Conceptual

Este modelo consiste em entidades, atributos e relações e, para cada elemento, existem as seguintes regras.

## Entidade

Caracteriza-se por um nome (frequentemente um substantivo) e um conjunto de atributos.

O nome é sempre escrito com as iniciais de cada palavra em **maiúscula**, e no **singular**, por exemplo

* Piloto
* Piloto Prova

Para designar uma entidade no modelo conceptual, usamos o **quadrado**.

## Atributo

Caracteriza-se por um nome, e, tal como as entidades, as iniciais de cada palavra têm letra **maiúscula**, e estão no **singular**:

* Idade
* Data de Nascimento

Para designar um atributo no modelo conceptual, usamos o **círculo oval**.

## Relação

Esta caracteriza-se por num nome e dois ou mais argumentos (entidades).

O seu nome tem de ser um **verbo** e apresenta a mesma sintaxe que a entidade, ou seja, letra **maiúscula** nas iniciais das palavras e no **singular**.

Temos como exemplo:

* Participa
* Concorre

A cardinalidade é usada para representar melhor as relações existentes. Por exemplo, podemos dizer que uma escola tem vários professores, e para representar isso usamos cardinalidade.

Aqui, no modelo conceptual usamos apenas a cardinalidade máxima, ou seja 1 ou N.

Para designar uma relação no modelo conceptual, usamos o **losango**.

# Nível Lógico

É constituído apenas por **tabelas** e as suas **ligações**.

## Tabelas

As tabelas são constituídas pelo seu nome identificador e por uma série de atributos.

O nome é escrito com a inicial **maiúscula** e os atributos em **camelCase.**

Para conectar tabelas, usamos novamente a cardinalidade, no entanto, no modelo lógico, somos mais específicos em relação a essa associação.

* Um
* Vários
* Um e apenas um
* Zero ou Um
* Um ou vários
* Zero ou vários

Neste nível, ainda, devemos não usar acentos nas palavras, ou seja, uma tabela com o nome *Promoção* ficaria ***Promocao.***

No modelo lógico, acontece inúmeras vezes existir a necessidade de criar tabelas intermédias para guardar dados, quando duas tabelas apresentam uma relação de ***n*** para ***n****.*

Quando isto acontecer, devemos seguir a seguinte nomenclatura:

* Tabela Prova
* Tabela Piloto
* Nova Tabela -> ProvaPiloto

\*Na eventualidade de existir algum caso particular no qual seja possível utilizar um nome mais intuitivo, é perfeitamente válida a sua utilização.

**Chaves primárias:**

No caso de não ser possível eleger uma chave primária natural é necessário criar uma artificial, nestes casos o nome deve seguir a sintaxe: ***idTabela.***

**Chaves estrangeiras:**

No momento da criação de uma foreign key, deve ser seguida a seguinte sintaxe: ***atributoTabelaproveniente***.

No entanto, no caso de a foreign key (ou uma das suas partes) se tratar de uma chave primária artificial (na sua tabela principal), a foreign key deve manter o seu nome.

No momento de criação de chave estrangeiras

# Nível Físico

Neste nível existe um mundo de regras e boas práticas que devemos seguir.

## Regras

### Geral

* Todas as palavras reservadas devem ser escritas em letra **maiúscula**.
* Todos os comandos devem acabar com **ponto e vírgula**.
* Todo o código, exceto *insert*, deve estar devidamente afastado e indentado.

### Criação de Tabelas

* O nome de *Tables*, *Views*, entre outros, devem apresentar **inicial maiúscula** e a convenção **camelCase**
* Os atributos destas devem apresenta a convenção **camelCase**

#### Constraints

* Todas as constraints, excluindo *foreign key*, devem apresentar o seguinte formato:
  + xx\_nomeTabela\_nomeAtributo1\_nomeAtributo2, onde xx representa o tipo de constraint
    - pk – Primary Key
    - nn – Not Null
    - uk – Unique
    - ck – Check
  + Introduzir apenas o número de argumentos necessários, pois não é obrigatório ter dois atributos
* Todas as constraints do tipo *foreign key* devem apresentar o seguinte formato
  + fk\_nomeTabela\_nomeAtributo1\_nomeTabelaReferenciada
* Sempre que possível, escrever as constraints **inline**, exceto *foreign key*
* Sempre que possível, escrever as constraint **outline** dentro da criação da tabela
  + Estas devem estar devidamente separadas do resto da criação da tabela (enter)

### Queries

* Todos estes componentes devem, obrigatoriamente, apresentar **indentação**
* Todos os comandos devem ser finalizados com **ponto e vírgula**
* Os comandos *Select* devem apresentar a seguinte formatação
  + SELECT \*

FROM Marinheiro

WHERE idMarinheiro = 2;

### Procedures/Functions

* Todas as palavras reservadas devem estar em maiúscula
* Indentação é obrigatório
* Todos os procedimentos e funções devem começar por
  + CREATE OR REPLACE
* O nome destes devem apresentar o seguinte formato
  + proc\_CreateSetor or func\_CreateSetor
* Os parâmetros devem apresentar o seguinte formato
  + designação\_parameter