**INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA**

**3D (Definição Dicionário de Dados)**

**Modelo de Tipos**

*Objectivo: Criar uma estrutura que especifique os tipos oferecidos pela plataforma e um conjunto de elementos que permitam redefinir esses tipos.*

**.: Tipos da Plataforma :.**

*Descrição: Conjunto de tipos que são sempre disponibilizados para o desenho de soluções (tipos base) e que são transversais à maioria das linguagens de programação. Possibilidade de se criarem tipos extendidos em que poderão ser redefinidas propriedades relativas a cada tipo.*

**Estratégia**

Com o objectivo de se disponibilizar um modelo de tipos por omissão com correspondência a tipos utilizados nas linguagens de programação (e.g. C#), bem como, em motores de bases de dados relacionais (e.g. SQL), optou-se por utilizar parte do modelo de tipos oferecido pela maioria destes últimos, no caso concreto do Microsoft SQL Server.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tipo | Eq. C# | Eq. SQL | max\_length | Precision | scale |
| guid | Guid | uniqueidentifier | 16 bytes | 0 | 0 |
| datetime | datetime | datetime | 8 bytes | 23 digits | 3 |
| int | int | int | 4 bytes | 10 digits | 0 |
| long | long | bigint | 8 bytes | 19 digits | 0 |
| double | double | real | 4 bytes | 24 digits | 0 |
| float | float | float | 8 bytes | 53 digits | 0 |
| decimal | decimal | decimal | 17 bytes | 38 digits | 38 |
| boolean | boolean | bit | 1 bytes | 1 digits | 0 |
| string(n) | string | varchar | 8000 chars | 0 | 0 |
| binary | byte[] | binary | 8000 bytes | 0 | 0 |
| char | char | char | 8000 chars | 0 | 0 |

Dependendo da natureza do tipo (numérico, caracter, decimal), o valor de cada um dos parâmetros referidos assume significados diferentes:

|  |  |
| --- | --- |
| Parâmetro | Significado |
| *Precision* | Número de dígitos num número |
| *Length* | Para tipos numéricos e binários representa o número de bytes ocupados.  Para tipos caracter representa o número de caracteres |
| *Scale* | Número de dígitos à direita do ponto decimal |

**Representação em XML e Regras de Validação**

Os tipos da plataforma têm representação em XML através do elemento *platformTypes*. Este irá conter elementos *platformType* especificando, por via de atributos, os valores acima indicados.

No que respeita à validação, o XML squema associado ao ficheiro descritor do modelo de tipos impõe a unicidade de nomes e limita o número de elementos *platformType* para o número exacto de tipos oferecidos.

**.: Tipos Custom :.**

*Descrição: Conjunto de elementos que extendem tipos da plataforma, redefinindo atributos adequados ao tipo base.*

**Estratégia**

A criação de tipos custom visa permitir a redefinição dos atributos do tipo base, entro os quais, dominio de valores, intervalo de valores possíveis, dimensão e máscara.

Neste sentido, identificaram-se, para cada tipo base passível de ser extendido, o conjunto de restrições aplicáveis.

|  |  |
| --- | --- |
| Tipo | Restrições |
| string | length, minLength, maxLength, pattern, enumeration |
| decimal | totalDigits, fractionDigits, pattern, enumeration, maxInclusive, maxExclusive, minInclusive, minExclusive |
| float | pattern, enumeration, maxInclusive, maxExclusive, minInclusive, minExclusive |
| double | pattern, enumeration, maxInclusive, maxExclusive, minInclusive, minExclusive |
| datetime | pattern, enumeration, maxInclusive, maxExclusive, minInclusive, minExclusive |
| binary | length, minLength, maxLength, pattern, enumeration |
| long | totalDigits, pattern, enumeration, maxInclusive, maxExclusive, minInclusive, minExclusive |
| int | totalDigits, pattern, enumeration, maxInclusive, maxExclusive, minInclusive, minExclusive |
| short | totalDigits, pattern, enumeration, maxInclusive, maxExclusive, minInclusive, minExclusive |
| byte | totalDigits, pattern, enumeration, maxInclusive, maxExclusive, minInclusive, minExclusive |

**Descrição das Restrições**

O significado de uma restrição depende do tipo a que se refere. Assim, com vista a facilitar a definição de tipos *custom*, criou-se o XML squema que especifica um elemento especifico para cada tipo base extensível (e.g. *customLong*, *customInt*, *customShort*).

**length**

Número de unidades de comprimento. No caso de aplicação ao tipo string significa o número exacto de caracteres e, no tipo binário significa o número exacto de bytes. O seu valor deverá ser um número inteiro positivo e será sujeito a validação quando utilizado em conjunto com *minLength* e *maxLength*..

**minLength**

Número mínimo de unidades de comprimento. No caso de aplicação ao tipo string significa o número mínimo de caracteres e, no tipo binário significa o número mínimo de bytes. O seu valor deverá ser um número inteiro positivo e será sujeito a validação quando utilizado em conjunto com *length* e *maxLength*.

**maxLength**

Número máximo de unidades de comprimento. No caso de aplicação ao tipo string significa o número máximo de caracteres e, no tipo binário significa o número máximo de bytes. O seu valor deverá ser um número inteiro positivo e será sujeito a validação quando utilizado em conjunto com *length* e *minLength*.

**pattern**

Permite caracterizer uma expressão regular que define a máscara de introdução. O seu valor será do tipo string.

**enumeration**

Restringe os valores possíveis de um tipo aos indicados na restrição, todavia, não impondo relações de ordem no domínio.

**maxInclusive**

Valor máximo incluido no intervalo de valores possíveis do dominio. O seu valor deverá ser do tipo a que a restrição diz respeito e será sujeito a validação quando utilizado em conjunto com *minExclusive* e *minInclusive*. A presença desta restrição invalida a utilização de *maxExclusive*.

**maxExclusive**

Valor máximo excluido do intervalo de valores possíveis do dominio. O seu valor deverá ser do tipo a que a restrição diz respeito e será sujeito a validação quando utilizado em conjunto com *minExclusive* e *minInclusive*. A presença desta restrição invalida a utilização de *maxInclusive*.

**minInclusive**

Valor mínimo incluido no intervalo de valores possíveis do dominio. O seu valor deverá ser do tipo a que a restrição diz respeito e será sujeito a validação quando utilizado em conjunto com *maxExclusive* e *maxInclusive*. A presença desta restrição invalida a utilização de *minExclusive*.

**minExclusive**

Valor mínimo excluido do intervalo de valores possíveis do dominio. O seu valor deverá ser do tipo a que a restrição diz respeito e será sujeito a validação quando utilizado em conjunto com *maxExclusive* e *maxInclusive*. A presença desta restrição invalida a utilização de *minInclusive*.

**totalDigits**

Define o número máximo de digitos do tipo que representa.

**fractionDigits**

Define o número máximo de digitos à direita do ponto decimal do tipo que representa

**.: XML Squema - (*typeModelValidation.xsd)* :.**

*Descrição: Regras para a criação do ficheiro XML representativo do modelo de tipos da solução.*

**Validação de *platformTypes***

Uma vez que se pretende disponibilizar tipos da plataforma para utilização na solução, é obrigatório que o XML referente aos tipos inclua todos os 11 elementos *platformType*.

A validação de *platformTypes* é então explícita mas, considerando a utilização do mesmo “bloco” de elementos em todas as soluções, acaba por nunca ser utilizada.

**Validação de *customTypes***

Os tipos *custom* são descritos em *squema* da seguinte forma:

<xs:complexType name="t\_customType">

<xs:choice>

<xs:element name="customLong" type="t\_customLong"/>

<xs:element name="customInt" type="t\_customInt"/>

<xs:element name="customShort" type="t\_customShort"/>

<xs:element name="customByte" type="t\_customByte"/>

<xs:element name="customBinary" type="t\_customBinary"/>

<xs:element name="customDecimal" type="t\_customDecimal"/>

<xs:element name="customFloat" type="t\_customFloat"/>

<xs:element name="customDouble" type="t\_customDouble"/>

<xs:element name="customString" type="t\_customString"/>

<xs:element name="customDatetime" type="t\_customDatetime"/>

</xs:choice>

<xs:attribute name="name" type="xs:string" use="required"/>

</xs:complexType>

Exige-se assim a presença do atributo *name,* que identifica o tipo na solução, e a presença de um elemento que indique qual o tipo que estamos a extender.

No que respeita ao nome do tipo, este deverá ser único e é validado no âmbito do elemento *customTypes*:

<xs:element name="customTypes" type="t\_customTypeContainer">

<xs:unique name="un\_customType\_name">

<xs:selector xpath="customType"/>

<xs:field xpath="@name"/>

</xs:unique>

</xs:element>

Dos elementos que deverão estar presentes em *customType* pretende-se que possibilitem a definição de restrições, conforme tabela em cima.

**.: Exemplos de Utilização :.**

**Tipo que representa o NIF:**

<customType name="nif">

<customString>

<length>9</length>

</customString>

</customType>

**Tipo que representa os pontos cardeais:**

<customType name="pontosCardeais">

<customString>

<enumeration>Norte</enumeration>

<enumeration>Sul</enumeration>

<enumeration>Este</enumeration>

<enumeration>Oeste</enumeration>

</customString>

</customType>

**Tipo que representa um semestre:**

<customType name="primeiroSemestre">

<customDatetime>

<minInclusive>2010-01-01T00:00:00</minInclusive>

<maxInclusive>2010-06-30T23:59:00</maxInclusive>

</customDatetime>

</customType>

**Tipo que representa um mês:**

<customType name="junho">

<customDatetime>

<pattern>2010-06-([0-2]{1}[0-9]{1}|30)T00:00:00</pattern>

</customDatetime>

</customType>