

Desvendando o CSN

A Anatomia dos Modelos CDS (Core Data Services)



A Espinha Dorsal dos Modelos CDS

O que é CSN?

Uma notação compacta e otimizada para modelos CDS. Pronuncia-se 'Season' (do inglês).

Mais que um JSON Schema

Ele vai além para capturar modelos de Entidade-Relacionamento (ERM) completos e suas extensões.

A Ponte Universal

CSN é a fonte perfeita para gerar artefatos-alvo, como:

- Interfaces (OData/EDM, OpenAPI)
- Modelos de Persistência (SQL, NoSQL)



A Anatomia de um Modelo CSN

Um modelo CSN é um objeto JS/JSON com até quatro propriedades principais, todas opcionais:

requires: Um array listando os modelos importados (dependências).

definitions: Um dicionário com todas as definições nomeadas (tipos, entidades, serviços). **Este é o coração do modelo.**

extensions: Um array de aspectos e anotações a serem aplicados.

i18n: Dicionários para textos traduzidos.

```
{  
  "requires": [ "@sap/cds/common", "./db/schema" ],  
  "definitions": {  
    "some.type": { "type": "cds.String", "length": 11 }  
    // ... mais definições aqui  
  },  
  "extensions": [  
    { "extend": "Foo", "elements": { "bar": { "type": "cds.String" } } }  
  ]  
}
```

O Coração do Modelo: `definitions`

A propriedade `definitions` é um dicionário que contém todos os artefatos do seu modelo.

Cada entrada é um par `chave: valor`, onde:

- **Chave:** É o nome absoluto e totalmente qualificado da definição (ex: `\"my.bookshop.Books\"`).
- **Valor:** É um objeto que descreve a definição em detalhes.

Dentro do objeto de valor, a propriedade `kind` diferencia o tipo do artefato: `entity`, `type`, `service`, `action`, etc.

```
{  
    definitions: {  
        'Currency': {  
            type: "cds.String",  
            length: 3  
        }, // kind "type" é隐式的  
  
        'Products': {  
            kind: "entity",  
            elements: {  
                'ID': {  
                    type: "cds.Integer",  
                    key: true  
                }  
            }  
        }  
    }  
}
```

'Products': {} Chave (Nome Qualificado)

kind: "entity", Diferencia o tipo do artefato

Fundamentos: Definindo Tipos de Dados

Scalar

Um tipo base com facetas (propriedades adicionais). Definido pela propriedade type.

```
{ "type": "cds.String", "Length": 3 }
```



Structured

Uma coleção de elementos nomeados. Definido pela propriedade elements.

```
{ "elements": { "foo": { "type": "cds.Intege
```

Arrayed

Uma lista de itens de um tipo específico. Definido pela propriedade items.

```
{ "items": { "type": "cds.Integer" } }
```



Enumeration

Um conjunto de valores fixos. Definido pela propriedade enum.

```
{ "enum": { "asc": {}, "desc": {} } }
```

Modelando o Domínio: Entidades

- Entidades são tipos estruturados com `kind: "entity"`.
- Assim como tipos estruturados, possuem uma propriedade `elements`.
- **Propriedades Essenciais dos Elementos:**
 - `key: true`: Sinaliza que o elemento é (parte da) chave primária.
 - `notNull: true`: Impõe uma restrição de não-nulidade, como em SQL.
 - `virtual: true`: O elemento é ignorado no mapeamento de persistência genérico.

A propriedade `includes` pode ser usada para reutilizar definições de aspectos ou tipos.



```
{  
  definitions: {  
    'Products': {  
      kind: "entity",  
      elements: {  
        key: true 'ID': { type: "cds.Integer",  
                           'title': { type: "cds.String",  
                                       notNull: true },  
                           'price': { type: "Amount",  
                                      virtual: true } },  
        }  
      }  
    }  
  })
```

Restrição de
não-nulidade

Ignorado na
persistência

Derivando Dados: Views e Projeções

- Views são entidades definidas como uma projeção sobre outras entidades.
- A presença da propriedade query (ou projection) as identifica no CSN.
- query: Captura a consulta de origem no formato CQN (Core Query Notation).
- **Flexibilidade:** Views podem ter uma assinatura de elementos explícita (elements) e aceitar parâmetros de entrada (params).

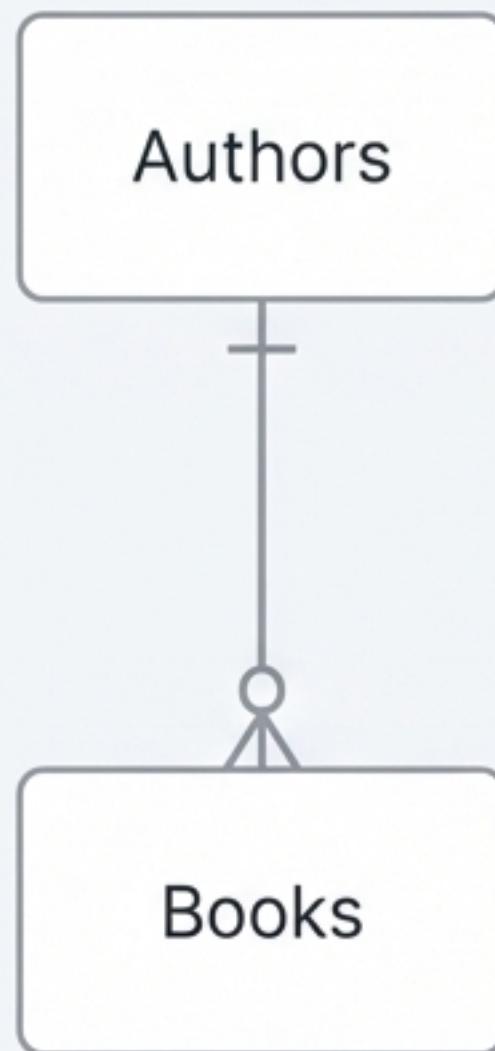
View Simples

```
'SimpleView': {  
    kind: "entity",  
    query: {  
        SELECT: { from: { ref: ['Bar'] } }  
    }  
}
```

View Parametrizada

```
'ParamView': {  
    kind: "entity",  
    params: { 'ID': { type: 'cds.Integer' } },  
    elements: { 'title': { type: "cds.String" } },  
    query: { SELECT:{...} }  
}
```

Conectando Tudo: Associações



Associações são definidas com `type: "cds.Association"` ou `type: "cds.Composition"`.

Propriedades Essenciais:

`target`: O nome da entidade de destino do relacionamento.

Controle Fino:

`cardinality`: Um objeto `{max: "*"}` define uma relação to-many.
`on`: Uma condição de junção explícita para associações não gerenciadas.
`keys`: Mapeamento explícito de chaves estrangeiras.

```
'Authors': {  
    kind: "entity",  
    elements: {  
        'books': {  
            type: "cds.Association",  
            target: "Books",  
            cardinality: { max: "*" }  
        }  
    }  
}
```

Enriquecendo o Modelo com Anotações

- Anotações adicionam metadados personalizados a qualquer definição ou elemento.
- São representadas como propriedades com o prefixo `@`.
- O prefixo atua como uma proteção contra conflitos com propriedades padrão do CSN.
- Exemplos: `@title": "\"Funcionários\"", `@readonly": true`.
- Podem ser aplicadas em qualquer nível: entidade, elemento, serviço, etc.

```
'Employees': {
    kind: "entity",
    '@title': "Mitarbeiter",
    '@readonly': true,
    elements: {
        'firstname': { type: "cds.String",
            '@title': "Vorname" },
        'surname': { type: "cds.String",
            '@title': "Nachname" }
    }
}
```

Modularidade com `extensions` (Aspects)

A propriedade de alto nível `extensions` contém um array de modificações a serem aplicadas dinamicamente ao modelo. Existem duas formas principais de modificação:

- **extend**: Adiciona novos elementos e/ou anotações a uma definição existente. Permite a composição de modelos.
- **annotate**: Adiciona ou sobrescreve *apenas* anotações em uma definição existente ou em seus elementos.



Adicionando Estrutura e Metadados

```
{  
  "extend": "Foo",  
  "@foo": true,  
  "elements": {  
    "bar": ...  
  }  
}
```



Adicionando Apenas Metadados

```
{  
  "annotate": "Foo",  
  "@foo": true,  
  "elements": {  
    "boo": { '@boo': true }  
  }  
}
```

A Camada de API: Serviços, Ações e Funções

- **Serviços:** São definições com `kind: ""service\"`. Atuam como contêineres para entidades e operações expostas.
- **Ações e Funções:** Definem operações executáveis, com `kind: "action"` ou `kind: "function"`.
 - `params`: Um dicionário que define os parâmetros de entrada da operação.
 - `returns`: Uma definição de tipo que descreve a resposta da operação (pode ser um tipo simples ou estruturado).

```
'OrderService.cancelOrder': {  
    kind: "action",  
  
    Parâmetros de Entrada  
    params: {  
        'orderID': { type: "cds.Integer" },  
        'reason': { type: "cds.String" },  
    },  
  
    Tipo de Retorno  
    returns: {  
        elements: {  
            'ack': { enum: { 'succeeded':{}, 'failed':{} } },  
            'msg': { type: "cds.String" },  
        }  
    }  
}
```

Organização e Globalização do Modelo

Imports (`requires`)

Um array de strings que especifica as dependências do modelo. Equivalente à diretiva `using` do CDL. Os caminhos podem ser módulos absolutos ou arquivos relativos (iniciando com `./` ou `../`).

```
"requires": [  
    "@sap/cds/common",  
    "./db/schema"  
]
```

Internacionalização (`i18n`)

Um dicionário opcional de alto nível para armazenar textos traduzidos. A estrutura esperada é `'{ "language-key": { "text-key": "translated string\" } }'`.

```
"i18n": {  
    "de": {  
        "PRODUCT_NAME": "Produktname"  
    }  
}
```

CSN: O Coração Compilado do CDS

Desenvolvedor (CDL):

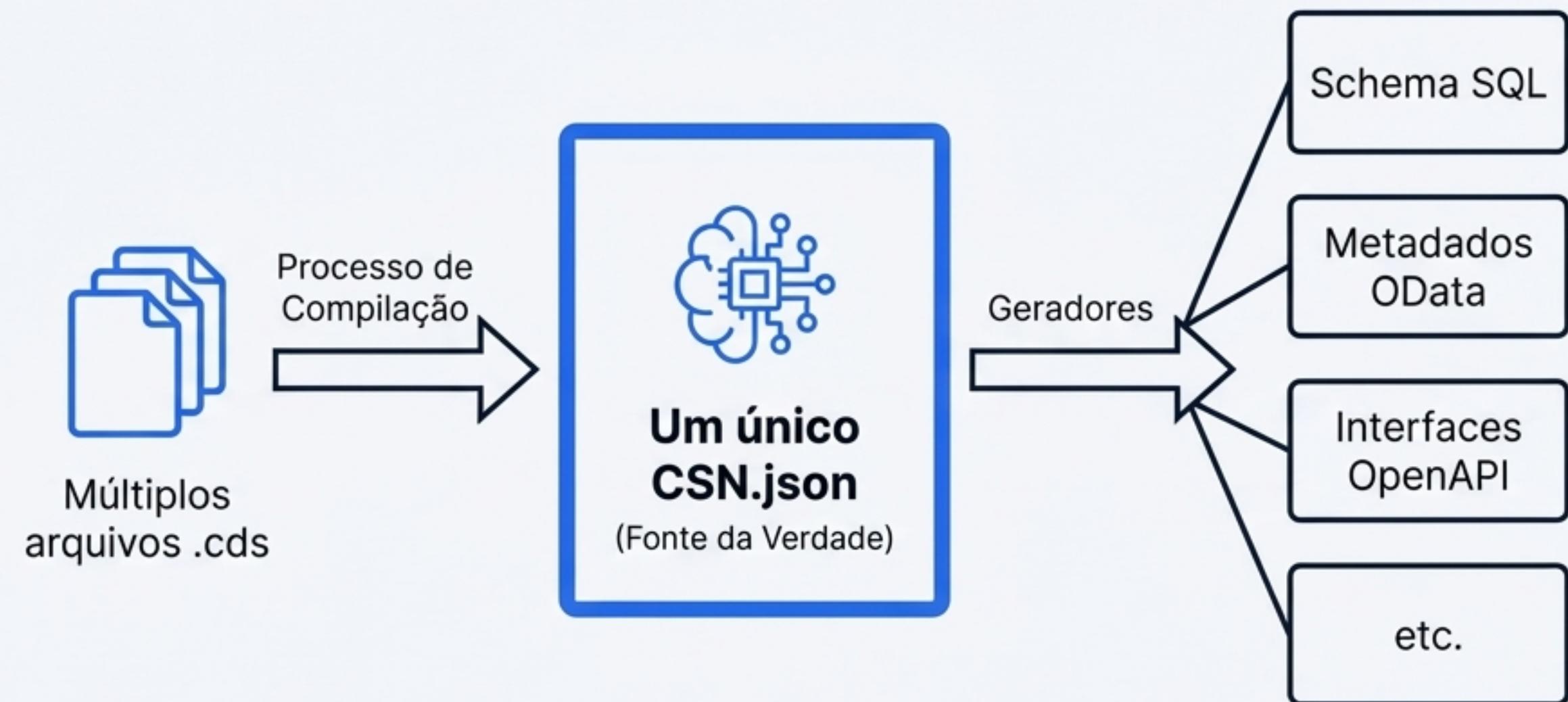
Escreve modelos de forma legível, concisa e focada no domínio de negócio.

Compilador (CSN):

Gera uma representação estruturada, detalhada e otimizada para ferramentas que captura a intenção completa do modelo. **É a fonte única da verdade.**

Runtimes (SQL, OData):

Consomem o CSN para gerar artefatos específicos da plataforma de destino.



Para Saber Mais

Explore a Documentação Oficial do SAP Capire

Core Schema Notation (CSN)

A especificação completa do formato.

cap.cloud.sap/docs/cds/csn



Definition Language (CDL)

A linguagem de autoria para modelos CDS.

cap.cloud.sap/docs/cds/cdl



Query Language (CQL)

A linguagem para consultar modelos CDS.

cap.cloud.sap/docs/cds/cql

