



## Guidelines de criação tabelas nas camadas

Criado por	@Marcello Otsuka
Status	DONE



### Contexto [↗](#)

É de suma importância mantermos uma padronização para a criação de tabelas no nosso ambiente de dados para garantir a organização e a consistência do ambiente, além de facilitar a manutenção futura por outros integrantes do time.

Neste guia, abordaremos os padrões de nomenclatura que devem ser seguidos para datasets e tabelas no BigQuery, bem como as melhores práticas para a criação de camadas staging.












### Escopo e não escopo [↗](#)

<b>Dentro do Escopo:</b>	<ul style="list-style-type: none"><li>• Definição e aplicação de regras para nomear datasets e tabelas na plataforma de dados da Arco.</li><li>• Regras aplicadas em cada camada</li><li>• Utilização de camadas staging</li></ul>
<b>Fora do Escopo:</b>	<ul style="list-style-type: none"><li>• Detalhamento de ferramentas utilizadas</li><li>• Governança de dados do ambiente</li></ul>



### Índice [↗](#)

- 1  Contexto
- 2  Escopo e não escopo
- 3  Índice
- 4  Guidelines
  - 4.1  Landing
  - 4.2  Raw
  - 4.3  Refined
  - 4.4  Business
- 5  Documentações auxiliares



### Guidelines [↗](#)

Vamos abordar aqui as diretrizes a serem seguidas em todas camadas do ambiente de dados para criação de tabelas, transformações e regras de negócios que podem ou não ser aplicadas.

## Landing [↗](#)

Camada que serve como ponto de entrada no ambiente de dados para informações provenientes de fontes variadas, como bancos de dados e APIs. Os dados são mantidos **exatamente iguais a fonte**.

### Regras e Transformações [↗](#)

✖ Nenhuma **transformação** ou **regra** deve ser aplicada aqui.

### Nomenclatura para criação das tabelas [↗](#)

Todos os dados ingeridos na Landing devem seguir o padrão de nomeação:

- **Projeto:** arco-data-lake
- **Dataset:** landing\_\_<origin>[\_instance]
  - origin: nome do database, API ou outra origem.
  - instance: nome da marca ou subdivisão relevante.
- **Tabelas:**
  - **Banco de dados:** db\_<origin>\_<schema>\_<table>
    - db: identifica tabelas de bancos de dados
    - schema: qual schema do banco de dados vem a informação.
  - **Outras fontes:** <origin>\_<domain>\_<table>
    - domain: pode ser o nome do endpoint
- **Exemplos:** Tabelas de Contas a Receber do ERP Protheus (SE1) das marcas SAS e SAE.
  - **SAS:** arco-data-lake.landing\_\_protheus\_prod.db\_PROTHEUS\_PROD\_dbo\_SE1010
  - **IS:** arco-data-lake.landing\_\_protheus\_prod.db\_PROTHEUS\_PROD\_dbo\_SE1020
  - **SAE:** arco-data-lake.landing\_\_protheus\_prod.db\_PROTHEUS\_PROD\_dbo\_SE1030

⚠ Os nomes das origens, schemas e tabelas devem ser mantidos como na fonte, não fazendo nenhum tipo de alteração para o inglês.

## Raw [↗](#)

Armazena os dados brutos vindos da camada Landing, já ajustados e otimizados para facilitar a modelagem de dados. Aqui são realizadas apenas pequenas padronizações e limpezas simples, como retirada de espaços em branco e tratamento de datas. Também teremos aqui as separações **current** e **event**.

### Regras [↗](#)

✖ Nenhuma **regra** deve ser aplicada nesta camada.


### Transformações [↗](#)

A ideia nesta camada é mantermos os dados e colunas o mais próximo de como visualizamos na origem, trazendo para essa camada apenas as colunas que realmente serão utilizadas, evitando processamento e armazenamento desnecessário.

Importante ressaltar que quando falamos de visualização como na origem aqui nos referimos a fonte original e não necessariamente como está na camada Landing. Por que? Nossos processos de ingestão normalmente adicionam algumas padronizações que mudam a forma de visualização das colunas, dificultando a leitura e validação ao compararmos com as fontes originais. Vamos apresentar alguns casos aqui exemplificando o que acontece em algumas ferramentas de ingestão e quais transformações devem ser aplicadas.

## Resumo das transformações por ingestão [↗](#)

Transformação/Ingestão	Kafka	Fivetran	Stitch
Remoção de espaços a mais com TRIM	✓	✓	✓
remover “before” e “after” das colunas	✓		
alteração de campos datetime para utc-3		✓	
cast de datas para date com origem sharepoint		✓	
aplicar rank para trazer registro mais atual para tabela “current”	✓	✓	✓
Substituir espaços vazios por null	✓	✓	✓

 Outras transformações podem ser aplicadas, desde que sejam com objetivo de manter os dados iguais a fonte original.

### Ingestão Kafka [↗](#)

O kafka adiciona nas colunas uma visão “before” e “after”, registrando os dados do evento antes e depois da alteração. Além disso, em alguns casos adiciona espaços a mais em campos de texto.

#### Tratativas [↗](#)

Para o nome das colunas, só utilizamos os registros de after, mas sem colocar o indicativo deste evento no nome da coluna. Além disso, usamos o **TRIM** para tirar espaços a mais e **NULLIF** para tratar strings vazias.


```
1 , NULLIF(TRIM(after.E5_LOJA), '')      as E5_LOJA
2 , NULLIF(TRIM(after.E5_PREFIXO), '')   as E5_PREFIXO
3 , NULLIF(TRIM(after.E5_TIPO), '')      as E5_TIPO
```

Para criarmos a visão **current**, também fazemos um rank ordenando pelo evento mais atual.

```
1 row_number () over (partition by coalesce(after.R_E_C_N_O_,before.R_E_C_N_O_)
2                      order by source.ts_ms desc, source.change_lsn desc) as rank
```

Também retiramos os registros deletados

```
1 select * from current_table where op <> 'd' or D_E_L_E_T_ <> '*'
```

 **Exemplo de script** com todas essas tratativas: [🔗raw\\_\\_protheus\\_prod.current\\_ct2010.sql](#)

### Ingestão Fivetran [↗](#)

Para ingestões do Fivetran, com origem do Sharepoint, as colunas de data, são inseridas no banco no formato timestamp numérico, o que exige um cast para data para conseguirmos visualizar exatamente como está no arquivo. Além disso, o horário de execução do evento é inserido com com horário em UTC, o que confunde o entendimento de quando o processo foi executado, dessa forma, também alteramos para UTC-3.

#### Alteração utc-3

```
1 FORMAT_TIMESTAMP("%F %T", _modified, 'America/Sao_Paulo') AS _modified
```


#### Cast campos data

```
1 DATE(TIMESTAMP_SECONDS(CAST(data_corte AS INT))) AS data_corte
```

 **Exemplo de script** que contém todos tratamentos: `raw__sharepoint_finance.current__status_cliente.sql`

## Nomenclatura para criação das tabela

Aqui ainda procuramos fazer referência aos nomes da origem, assim como na landing, porém de maneira mais enxuta. Por conta disso, o nome do dataset passa a ter o schema como opcional para as origens que trazemos apenas um schema. Já o nome das tabelas passa a apresentar também seu tipo de estrutura, sendo **current** ou **event**.

- **Projeto:** arco-data-lake
- **Dataset:** raw\_\_<origin>[\_schema]
  - origin: nome do database, API ou outra origem. (utilizar o mesmo da camada **Landing**)
  - schema: vale apenas para origens de banco de dados.
- **Tabela:** <table-type>\_\_<table-name>.
  - table-type: current ou event
  -  o nome das tabelas devem ficar em caixa baixa
- **Exemplos:** As mesmas tabelas presentes no exemplo da camada [landing](#), agora para camada raw.
  - arco-data-lake.raw\_\_protheus\_prod.current\_\_se1010
  - arco-data-lake.raw\_\_protheus\_prod.current\_\_se1020
  - arco-data-lake.raw\_\_protheus\_prod.event\_\_se1010
  - arco-data-lake.raw\_\_protheus\_prod.event\_\_se1020

## **Refined**



Transforma os dados da camada Raw, e também da staging refined (que será abordada adiante), **refinando-os** para formar uma **visão unificada** de processos específicos, **mantendo a granularidade** da informação encontrada na sua tabela fonte. Por exemplo, uma tabela de recebíveis de várias fontes como Protheus, Oracle, SAP é consolidada nesta camada em uma única tabela.

## **Regras**

A aplicação de regras nessa camada deve existir somente se forem amplamente difundidas dentro da entidade que essa tabela trata, sendo útil para diversos processos que poderão utiliza-la. Como exemplo, temos uma regra aplicada na tabela `current__receivables` em uma coluna que indica se aquele recebível é de uma operação B2B ou B2C. Essa regra trata de um conceito amplamente difundido dentro da empresa e de suma importância quando se trata de recebíveis.

## **Transformações**




Nessa camada alteramos o tipo dos campos, fazendo cast para int, string, date, etc. Alteramos o nome das colunas para ficarem mais condizentes com seu assunto, tendo como padrão a língua inglesa em caixa baixa. O relacionamento com outras tabelas para adição de campos é permitido. Não há nenhuma regra obrigatória para o tratamento de strings, como retirada de acentos ou deixar tudo em caixa alta/baixa.

 Para nomenclatura de colunas confira esta documentação de boas práticas:  [Guidelines para descrições de coluna](#)

S

## Nomenclatura para criação das tabela

Os datasets são separados por **entidades**, como exemplo temos a entidade “receivables”. Aqui entendemos entidade como os objetos que possuem uma identidade única e podem sofrer alterações ao longo do tempo, não sendo necessariamente o nome da tabela.

 Consulte o levantamento dos principais **domínios** e **entidades** da arco:  [https://arcoeducacao.sharepoint.com/:w:/r/sites/CV-DataEngineering/\\_layouts/15/doc2.aspx?sourcedoc=%7BF3D70AB9-FE3C-4B3C-AC35-0B91B17DFBD5%7D&file=Mapeamento%20Dom%C3%ADnios%20e%20Entidades%20Arco.docx&action=default&mobileredirect=true](https://arcoeducacao.sharepoint.com/:w:/r/sites/CV-DataEngineering/_layouts/15/doc2.aspx?sourcedoc=%7BF3D70AB9-FE3C-4B3C-AC35-0B91B17DFBD5%7D&file=Mapeamento%20Dom%C3%ADnios%20e%20Entidades%20Arco.docx&action=default&mobileredirect=true)  Conectar a conta do OneDrive

Usaremos então o seguinte padrão:

- **Projeto:** arco-data-warehouse
- **Dataset:** refined\_\_<entity>
- **Tabela:** <table-type>\_\_<table-name>.
  - table-type: current ou timeline
- **Exemplo:** As mesmas tabelas presentes no exemplo das camadas [landing](#) e [raw](#), agora unificadas em uma só na camada refined.
  - arco-data-warehouse.refined\_\_receivables.current\_\_receivables
  - arco-data-warehouse.refined\_\_receivables.timeline\_\_receivables

## Staging Refined

Essa subcamada tem intuito de auxiliar no processo de construção dos dados. Esta área conterá tabelas intermediárias destinadas exclusivamente a esse propósito, facilitando a compreensão e manutenção do código.

## Regras e Transformações

Segue os mesmos padrões de regras e transformações da sua camada alvo: refined.

## Nomenclatura para criação das tabela

Mantemos aqui para os datasets a mesma **entidade** que irá constar na tabela refined, seguindo com o mesmo exemplo da entidade “receivables”.

- **Projeto:** arco-data-warehouse
- **Dataset:** staging\_\_refined\_\_<entity>
- **Tabela:** <table-type>\_\_<table-name>\_\_<origin>[\_subject].
  - **table-type:** current ou timeline
  - **origin:** a fonte da onde vem a informação, em casos dos erps poderia ser o Protheus, SAP, Oracle ou nos CRMs Salesforce, Pipedrive, Zoho, etc.
  - **subject:** Entra como opcional aqui, podendo ser qualquer segmentação que faça sentido para auxiliar no desenvolvimento. No exemplo que vamos apresentar a seguir utilizamos uma separação por marca.
- **Exemplo:** As mesmas tabelas presentes no exemplo das camadas [landing](#) e [raw](#), agora tratadas para serem unificadas na camada refined.
  - arco-data-warehouse.refined\_\_receivables.current\_\_receivables\_\_protheus\_\_sas
  - arco-data-warehouse.refined\_\_receivables.current\_\_receivables\_\_protheus\_\_is
  - arco-data-warehouse.refined\_\_receivables.current\_\_receivables\_\_protheus\_\_sae

## Business



Utiliza os dados refinados da camada Refined para transformá-los em informações estratégicas, desenvolvendo KPIs e OKRs que apoiam decisões e melhoram processos operacionais. Como exemplo temos a tabela de contas a receber contendo as regras de negócio aplicadas.

## Regras

As regras aplicadas aqui não precisam ter um conceito amplo como abordadas na camada refined. Os filtros e agregações necessários para criação da tabela que o negócio precisa podem ser aplicados aqui. Como exemplo temos a tabela `timeline__accounts_receivable` que contém apenas os títulos considerados para o contas a receber que serão utilizados nos fechamentos mensais da companhia.




## Transformações

Como a business consumirá os dados da camada refined que já é tratada, não haverá transformações de tipo de coluna. Manteremos o mesmo padrão de colunas em inglês, caixa baixa.

 Para nomenclatura de colunas confira essa documentação de boas práticas:  [Guidelines para descrições de coluna](#)

## Nomenclatura para criação das tabela

A princípio os datasets serão separados por **domínios**, como exemplo temos o domínio “finance”. Aqui entendemos domínio como a esfera de influência que agrupa as entidades, não sendo necessariamente o nome do departamento em si.

 Consulte o levantamento dos principais domínios e entidades da arco:  [https://arcoeducacao.sharepoint.com/:w:/r/sites/CV-DataEngineering/\\_layouts/15/doc2.aspx?sourcedoc=%7BF3D70AB9-FE3C-4B3C-AC35-0B91B17DFBD5%7D&file=Mapeamento%20Dom%C3%ADnios%20e%20Entidades%20Arco.docx&action=default&mobileredirect=true](https://arcoeducacao.sharepoint.com/:w:/r/sites/CV-DataEngineering/_layouts/15/doc2.aspx?sourcedoc=%7BF3D70AB9-FE3C-4B3C-AC35-0B91B17DFBD5%7D&file=Mapeamento%20Dom%C3%ADnios%20e%20Entidades%20Arco.docx&action=default&mobileredirect=true)  Conectar a conta do OneDrive

- **Projeto:** arco-data-warehouse
- **Dataset:** business\_\_<domain>
- **Tabela:** <table-type>\_\_<table-name>.
- **Exemplo:** Aqui utilizaremos as tabelas da camada [refined](#) para criação de visões mais estratégicas.
  - arco-data-warehouse.business\_\_finance.current\_\_accounts\_receivable
  - arco-data-warehouse.business\_\_finance.timeline\_\_accounts\_receivable

## Staging Business

Assim como nada camada Refined, aqui temos uma subcamada opcional que também tem intuito de auxiliar no processo de desenvolvimento. Esta área conterá tabelas intermediárias destinadas exclusivamente a esse propósito, facilitando a compreensão e manutenção do código.

## Regras e Transformações

Segue os mesmos padrões de regras e transformações da sua camada alvo: business.

## Nomenclatura para criação das tabela

- **Projeto:** arco-data-warehouse
- **Dataset:** staging\_\_refined\_\_<domain>
  - domain: Mantemos aqui o mesmo **domínio** que irá constar na tabela business, seguindo com o mesmo exemplo do domínio “finance”.
- **Tabela:** <table-type>\_\_<table-name>\_\_<origin>[\_subject].
  - table-type: current ou timeline
  - origin: a fonte da onde vem a informação, em casos dos erps poderia ser o Protheus, SAP, Oracle ou nos CRMs Salesforce, Pipedrive, Zoho, etc.

- subject: Entra como opcional aqui, podendo ser qualquer segmentação que faça sentido para auxiliar no desenvolvimento. No exemplo que vamos apresentar a seguir utilizamos uma separação por marca.

- **Exemplos**

- arco-data-arehouse.staging\_\_refined\_\_receivables.current\_\_accounts\_receivable\_\_protheus\_\_b2b
- arco-data-warehouse.staging\_\_refined\_\_receivables.current\_\_accounts\_receivable\_\_protheus\_\_b2c

## Documentações auxiliares

As documentações abaixo lhe ajuda a entender com mais profundidade as camadas e outros temas abordados aqui.

- [Camadas do Ambiente de Dados](#): PPT contendo uma breve apresentação das camadas do nosso ambiente de dados.
- [Camadas ambiente de dados](#): Documentação completa para se aprofundar, contendo as camadas e projetos que utilizaremos no nosso ambiente, explicando estrutura, governança e organização.
- [Entidades e Domínios Arco](#): Mapeamento dos principais domínios e entidades que temos na empresa. Bom para ter uma versão macro do que temos aqui.