**4. DESCRIÇÃO DA MONTAGEM DO AMBIENTE**

Nessa seção serão descritos todos os passos, técnicas e dados utilizados para a aplicação da metodologia de *Business Intelligence* no contexto do presente trabalho.

4.1 Introdução

Segundo Braghittoni (p.1, 2017): “O BI é um conjunto de conceitos e métodos para melhorar o processo de tomada de decisão, utilizando-se de sistemas fundamentados em fatos e dimensões”. Nesse caso pode-se perceber que o BI é uma metodologia, que possui regras, ordem e práticas para sua aplicação. Sendo assim, é necessário descrever cada uma das partes que vão compor o ambiente de inteligência, utilizando como base os autores Braghittoni (2017), Carvalhaes e Alves (2015), Inmon (2005) e Kimball e Ross (2013).

Esse ambiente divide-se em (Figura x):

* Parte 1: Fontes de Dados (*Data Source*), em que é feita a definição da localização dos dados, seu formato e quais deles serão aproveitados para a análise.
* Parte 2: Área de *Staging* (*Staging Area*), passo intermediário e opcional onde os dados são gravados, na sua forma original, em tabelas para posterior transformação e inserção.
* Parte 3: *Data Warehouse* (DW), que adquire os dados do banco *Staging*, após serem feitas as transformações para que eles possam ser utilizados pela ferramenta de BI. Sua criação pode ser antes ou depois de um *Data Mart*.
* Parte 4: *Data Marts*, que são pequenos DW relacionados a um assunto específico. Sua criação pode ser antes ou depois de um *Data Warehouse* dependendo a abordagem escolhida.
* Parte 5*:* Análise dos resultados, em que a ferramenta de BI escolhida acessa os dados de um *Data Warehouse* ou de um *Data Mart* para que sejam feitas as gerações das análises.

Suas definições serão explicadas a frente.

Figura x – Arquitetura do ambiente de BI.



Fonte: Panoly (2019).

4.2 Montagem do ambiente – Fontes de Dados (*Data Source*).

O primeiro passo na aplicação dos processos de Business Intelligence é definir quais serão as bases de dados utilizadas para o processo e quais dados serão extraídos delas. No caso do presente trabalho, foram utilizadas as bases de micro dados do censo escolar do INEP, disponíveis no Portal Brasileiro de Dados Abertos no link: <http://dados.gov.br/dataset/microdados-do-censo-escolar> e no próprio site do INEP no link: <http://inep.gov.br/web/guest/microdados>. Para a melhor delimitação do trabalho, foram utilizados os censos dos anos de 2015 a 2018.

Os arquivos estão em formato CSV (*Comma-separated Values*) que é um tipo de arquivo onde seus dados estão separados por algum delimitador, no caso das bases do INEP é utilizado o delimitador *Pipe* (|). Eles são divididos em Turmas, Escolas, Matriculas (Centro-Oeste, Nordeste, Norte, Sudeste e Sul), e Docentes (Centro-Oeste, Nordeste, Norte, Sudeste e Sul), onde se encontra as informações das turmas, das escolas, dos alunos e dos docentes envolvidos nos censos de cada ano, respectivamente.

Além dos dados principais, faz-se necessário o uso de tabelas auxiliares para auxiliar na definição dos dados do INEP, já que são utilizados campos com os códigos dos Países, Unidades da Federação (UF), Municípios, Distritos, Mesorregiões e Microrregiões. Para o primeiro, o INEP disponibiliza em sua base, ao fazer download, uma tabela (Figura x) que contêm os códigos dos países descritos no censo, já que alunos estrangeiros também são envolvidos no censo escolar.

Figura x – Tabela de códigos dos países.



Fonte: Adaptado de INEP (2019).

Para as UF, Municípios, Distritos, Mesorregiões e Microrregiões, foram utilizadas as bases de códigos *Geodata* disponíveis no site *GitHub* no link*:* https://github.com/paulofreitas/geodata-br/tree/master/data/pt. O *GitHub* é um site para a criação de repositórios públicos e privados com o intuito de compartilhar informações e códigos e o repositório *Geodata* tem como propósito prover informações precisas e atualizadas acerca dos dados geográficos do Brasil. Essas informações são um compilado das informações disponíveis na SIDRA (Sistema IBGE de Recuperação Automática) formados pelo IBGE (Instituto Brasileiro de Geografia e Estatística).

4.3 Montagem do ambiente – Área de *Staging*

Inmon (p. 29, 2005) define em um dos seus postulados sobre *Data Warehouse* a não volatilidade, ou seja, os dados dentro do mesmo não podem sofrer alterações. Isso significa que se faz necessária uma fase intermediária antes de carregar os dados no DW, para isso tem-se a *Staging Area* ou *Data Stage*. Com todos os dados já na máquina é iniciada a montagem dos processos de ETL para fazer a carga no Banco de Dados de *Staging*.

Será utilizado o *Pentaho Data Integrator* (PDI) versão 5.0.1 para iniciar os processos de ETL, separando as cargas por assunto. O PDI utiliza duas nomenclaturas como *Job* e *Transformation*, o primeiro é a menor ação possível que o programa possa fazer como ler o arquivo ou fazer inserção, e o segundo é um conjunto de outros *Jobs* para fazer uma execução única e contínua. Como na imagem abaixo:

Figura x – Exemplo de *Transformation* e *Job*.



Fonte: Pentaho (20-?).

A carga dos arquivos no BD dos arquivos principais (turmas, matrícula, escolas, docentes) e das bases de códigos das UF, Municípios, Distritos, Mesorregiões e Microrregiões são compostas por três passos, em que o PDI encontra os arquivos, prepara-os para a inserção e grava-os no BD, como pode ser visto na imagem abaixo:

Figura x – Visão da ETL das bases principais.

Fonte: Autores (2019).



Os passos são descritos abaixo:

*Get* *File Names*: Esse *step* procura nomes de arquivos ou pastas. Ele é recomendado para quando se tem uma grande massa de dados em que todos precisam ser gravados. Os padrões dos nomes são adquiridos conforme uma expressão regular.

*Text File Input*: Aqui o Pentaho prepara um ou mais arquivos de textos para a inserção, nele são configuradas diversas opções como os delimitadores do texto, linha de título, formato e colunas adicionais para serem adicionadas no momento da carga.

*Table Output*: Realiza a carga dos dados estruturados em uma tabela no banco de dados. A tabela não precisa ser criada com antecedência, já que o PDI prepara um comando SQL automaticamente para a mesma ser criada.

Já para a carga das tabelas contendo o código dos países, foi utilizado um padrão de carga diferente, já que o arquivo que possui esse dado está em um formato diferente das outras bases, como pode ser visto na imagem abaixo:

Figura x – Visão geral da ETL de auxiliares.



Fonte: Autores (2019).

Os passos são descritos abaixo:

*Microsoft Excel Input*: Esse *step* procura nomes de arquivos do tipo XLS (formato utilizado nas versões de 97 até 2003) e/ou XLSX (utilizado na versão de 2007 em diante). Nele podem-se configurar opções como, especificar de qual linha e/ou coluna deve-se iniciar a análise, se os títulos das colunas estão na primeira linha (*Header*), além de especificar campos adicionais no momento da carga.

*Table Output*: Como descrito nas cargas principais, esse passo realiza a carga dos dados estruturados em uma tabela no banco de dados. A tabela não precisa ser criada com antecedência, já que o PDI prepara um comando SQL automaticamente para a mesma ser criada.

Após definir cada uma das ETLs, será usado uma *Transformation* para unir todos os outros *Jobs*, como pode ser visto na imagem abaixo:

Figura x – Visão geral da ETL *Staging*.



Fonte: Autores (2019).

Com todo o fluxo executado, o banco de dados *Staging* foi finalizado na forma da imagem abaixo:

Figura x – Visão do Banco *Staging*.



Fonte: Autores (2019).

4.4 Montagem do ambiente – *Data Warehouse*

Após o banco *Staging* estar completamente carregado, será iniciado os processos para a formação do armazém de dados.

4.4.1 Fato e Dimensão

Em uma modelagem multidimensional temos dois tipos de tabelas principais: Fato e Dimensão.

Pela definição de Kimball (2013) dimensão é uma coleção de atributos semelhantes ao texto que estão altamente correlacionados entre si. Isso quer dizer que ela possui característica descritiva. Para se criar uma dimensão podem ser feitas algumas perguntas como “quando”, “onde”, “quem”, e “o que”. Já na tabela fato, normalmente os dados são apenas números, categorizando-a em quantitativa, mas também se podem ter textos que estão classificando o fato em análise.

4.4.2 Abordagem Inmon x Kimball

Antes de começar o desenvolvimento das ETLs, deve-se pensar como será a estrutura e modelo do *Data Warehouse*, tendo como base a abordagem Inmon ou Kimball. Vale ressaltar que não há uma escolha certa ou errada, mas aquela que atende melhor os requisitos e necessidades da organização.

Inmon utiliza a abordagem *top-down* em que o DW é um repositório de dados centralizado, sendo assim o componente mais importante da organização (PANOLY, 2019). Ele é o primeiro modelo criado logo após a extração de dados, e após sua finalização são criados todos os *Data Marts* necessários. Seu diagrama é mostrado abaixo:

Figura x – Modelo Inmon.



Fonte: Panoly (2019).

Em contrapartida, Kimball utiliza a abordagem *bottom-up* em que é feita primeiramente a criação de *Data Marts* em cada área de interesse para depois se criar um grande *Data Warehouse* que é unicamente uma junção de todos esses *Marts* (PANOLY, 2019). Tal como Kimball (1997) afirma: “O *Data Warehouse* não é nada mais do que uma junção de diversos *Data Marts*”. Seu diagrama é mostrado abaixo:

Figura x – Modelo Kimball.



Fonte: Panoly (2019).

Para a realização desse trabalho foi escolhida a abordagem Inmon porque o projeto não terá uso de *Data Marts*, assim sendo, será criado unicamente o *Data Warehouse* para armazenar os dados.

4.4.3 Modelos Estrela e Floco de Neve (*Star Schema and* *Snow-Flake* *Schema*)

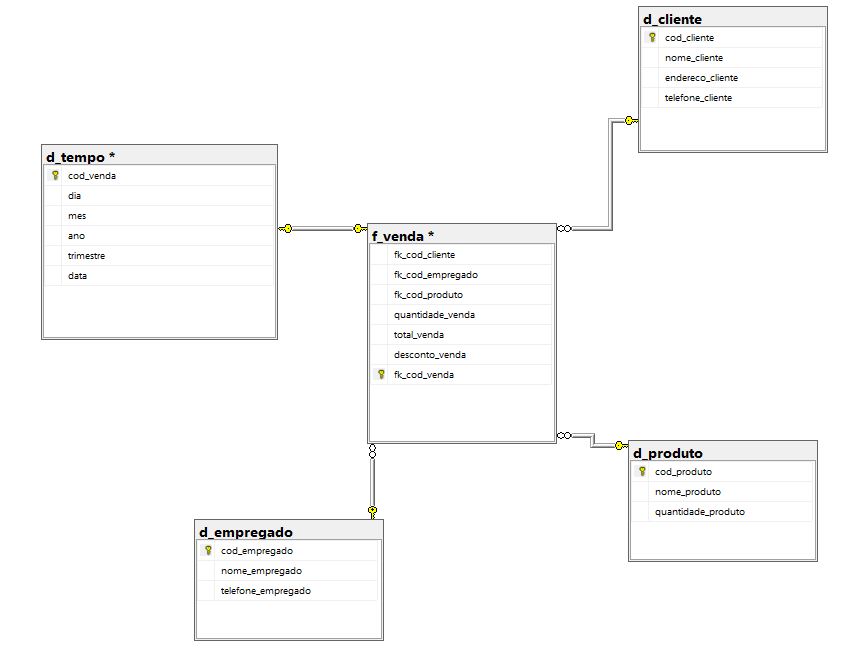
Tendo definida a estrutura, inicia-se o desenvolvimento do modelo do DW.

Em um modelo de dados multidimensional pode ser utilizado dois tipos de modelos, que são: tipo Estrela (*Star Schema*) ou tipo Floco de Neve (*Snow-Flake* *Schema*).

O modelo Estrela é o mais básico e mais comum para a arquitetura do *Data Warehouse*. No seu desenho, a tabela fato (F\_VENDA, Figura x) assume o centro da arquitetura seguido pelas tabelas de dimensões, que em volta dela, definem a quantidade de pontas da Estrela (CARVALHAES e ALVES, 2015). Possui como vantagem uma visualização simplificada dos dados, além de mais agilidade nas análises.

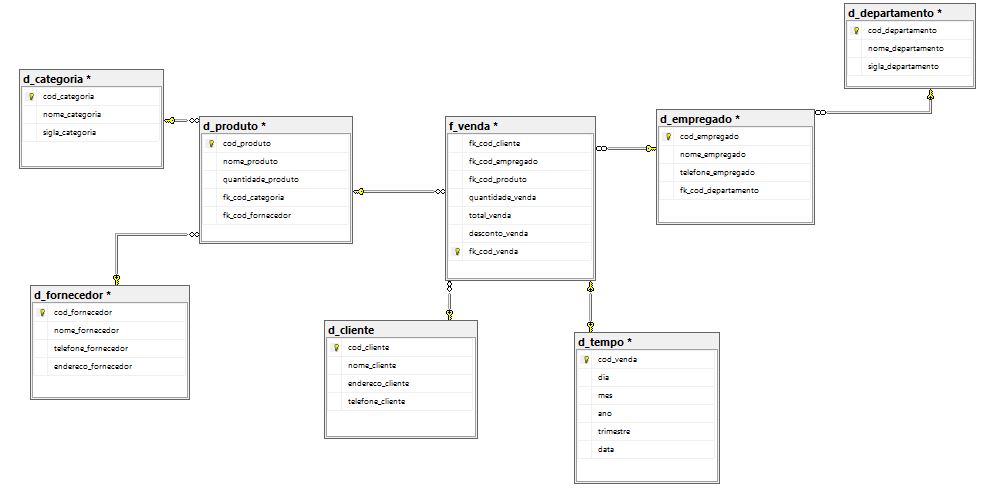
Figura x – Exemplo de modelo Estrela.

Fonte: Autores (2019).



O modelo Floco de Neve é um modelo específico que, partindo do modelo Estrela, as dimensões que possuem hierarquia são decompostas em outras tabelas (CARVALHAES e ALVES, 2015). Nesse modelo tem-se uma redução de redundância nas tabelas de dimensões e uma menor quantidade de memória utilizada. Um exemplo seria a dimensão chamada Data, em que ela poderá ser decomposta em outras tabelas como dia, mês, ano, trimestre, etc. Assim, essas “sub-dimensões” vão compor a dimensão principal. Seu diagrama é mostrado abaixo:

Figura x – Exemplo de modelo Floco de Neve.



Fonte: Autores (2019).

Para o presente trabalho, será utilizado o modelo Floco de Neve. Devido algumas dimensões apresentarem hierarquia nelas, houve-se a necessidade de criar uma tabela adicional.

4.4.4 Indicadores levantados para as análises

Ao desenvolver uma plataforma de BI, o objetivo é sempre responder a perguntas utilizando dados, que por sua vez se transformam em informação e auxílio na tomada de decisão. Para isso é necessário levantar perguntas que serão os indicadores da análise, com isso, serão definidas as fatos e dimensões. As perguntas levantadas pelo grupo são as seguintes:

1. Qual o total de alunos por cada Cor/Raça definida pelo Censo Escolar entre os anos da análise?
2. Qual o total de alunos que se declararam negros entre os anos da análise?
3. Qual o total de alunos estrangeiros que se declararam negros entre os anos da análise?
4. Qual o país que possui a maior quantidade de alunos estrangeiros negros no Brasil entre os anos da análise?
5. Qual o total de alunos negros por região, UF e município entre os anos da análise?
6. Qual é a diferença de alunos negros entre as regiões Nordeste e Sudeste nos anos da análise?
7. Qual é a quantidade de alunos negros no Distrito Federal entre os anos da análise?
8. Qual a quantidade de alunos negros que estudam em escolas sem água, energia, esgoto e alimentação entre os anos da análise?
9. Qual a quantidade de alunos negros por sexo entre os anos da análise?
10. Qual a quantidade de alunos negros nos módulos de ensino Presencial, Semipresencial e a Distância entre os anos da análise?
11. Qual a quantidade de alunos negros que moram em zona Urbana ou Rural entre os anos da análise?
12. Qual a quantidade de alunos negros que estudam em escolas Públicas e Privadas?
13. Qual a quantidade de alunos negros que estudam em escolas Urbanas e Rurais?
14. Qual a quantidade de alunos negros em cada etapa de ensino definida no censo entre os anos da análise?

Com as perguntas concluídas, pode-se agora levantar os fatos e dimensões da análise. Será utilizada apenas uma tabela fato, que é a tabela de matrículas (alunos) e as seguintes dimensões: Tempo (Ano), Localidade Município (Município, UF, País), Localidade Distrito (Microrregião, Município, UF, Região, Mesorregião, Distrito) e Escola.

Com a fato e as dimensões já definidas, será criada as ETLs para a carga das informações no *Data Warehouse*.

4.4.5 Processo ETL para carga do *Data Warehouse*

Nessa parte de explicação das ETLs, será separado por dimensões que possuem padrões de carga semelhantes, explicando os dados envolvidos e o processo.

4.4.5.1 Definição dos indicadores nulos

Segundo Braghittoni (p. 94, 2017) nenhuma coluna que esteja inserida no *Data Warehouse* pode aceitar valores nulos (*null*). O site *W3Schools* (20-?) define valores *null* como um campo que não possui valor, deixado em branco no momento da gravação. Sendo assim, há a necessidade de criar valores genéricos para definir um valor que veio nulo. Em cada uma das dimensões explicadas adiante, será descrito os seus respectivos indicadores nulos.

4.4.5.2 Dimensão Tempo (Ano)

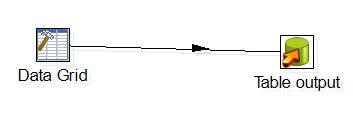
Inmon (p. 29, 2005) define em um dos seus postulados sobre *Data Warehouse* a variabilidade com o tempo, ou seja, um DW e suas informações vivem com base no tempo. Com base dessa informação, Braghittoni (p. 31, 2017) atesta que “Por mais que não exista nenhuma outra dimensão no seu DW, a dimensão temporal deve estar lá” e também (p. 73, 2017) “Como postulado por Inmon, o DW é sempre variável com o tempo, ou seja, a dimensão DATA deve invariavelmente existir”.

Conforme definido pelos dois autores, todo projeto necessita obrigatoriamente de uma dimensão de tempo, em contrapartida, esse ‘tempo’ pode ser descrito de formas diferentes por cada projeto, com base nas necessidades das análises. Ele pode ser definido tanto como informações separadas (ano ou mês ou dia), uma data, formada por ano, mês, e dia, ou até uma informação mais complexa inserindo trimestre, dia da semana, hora, etc.

Para o presente trabalho, a dimensão de tempo será identificada pelos anos referentes a cada análise (2015 até 2018), um identificador para cada uma delas e seu indicador nulo que será explicado adiante.

Seu diagrama de carga é mostrado na imagem abaixo:

Figura x – Visão geral da ETL Ano para o DW.



Fonte: Autores (2019).

Os seguintes passos foram utilizados:

*Data Grid*: Neste *step* será criada uma tabela com um conjunto constante de dados, informando os nomes dos campos, seus tipos, e seus respectivos dados. Aqui está sendo carregado cada um dos anos da análise e um indicador para cada um deles.

*Table Output*: Conforme explicado na parte do *Staging*,esse passo realiza a carga dos dados estruturados em uma tabela no banco de dados. A tabela não precisa ser criada com antecedência, já que o PDI prepara um comando SQL automaticamente para a mesma ser criada. Aqui os dados estão sendo gravados na tabela D\_TEMPO do banco de dados.

Indicador nulo da dimensão:

Como explicado no início da seção, serão atribuídos valores para serem inseridos para caso o campo da informação, no momento da carga, for nulo. Para essa dimensão será usado o indicador ‘-1’ para caso em algum momento da carga essa informação estiver nula.

4.4.5.3 Dimensões Localidade

Como descrito na seção 4.4.4 acerca dos indicadores e das dimensões, será usado duas tabelas chamadas de Aluno e Matrícula, elas por sua vez utilizam informações geográficas na sua estrutura.

Em base dos microdados do INEP, as tabelas sobre Aluno utilizam as informações de município, UF, e país, por outro lado, a dimensão Escola faz uso das informações de distrito, município, UF, microrregião, mesorregião e região.

Tendo em vista que cada uma das tabelas apresenta uma combinação de informações diferentes, fez-se necessário dividir as informações de localidade, com cada uma sendo chamada pelo seu menor grão de informação. No caso das combinações de Aluno, a dimensão com as suas combinações será chamada de Localidade Município, e de Escola, chamada de Localidade Distrito, por este ser o menor nível de informação. Essas informações geográficas seguem a seguinte ordem (do maior para o menor):

1. País;
2. Região;
3. UF;
4. Mesorregião;
5. Microrregião;
6. Município;
7. Distrito;

As definições de cada uma das dimensões Localidade serão explicadas adiante.

4.4.5.3.1 Dimensão Localidade Distrito

Como explicado anteriormente, uma das tabelas será a Localidade Distrito que irá apoiar as combinações da tabela Escola. Essa tabela de Localidade é formada pela combinação das informações de distrito, município, microrregião, mesorregião, UF e região, junto de um identificador único para cada uma dessas combinações, além dos identificadores nulos.

Seu diagrama de carga é mostrado na imagem abaixo:

[INSERIR IMAGEM]

Os seguintes passos foram utilizados:

*Table Input* (na imagem acima com os nomes: distrito, municipio, microrregiao, mesorregiao, uf): Este *step* permite utilizar os dados já existentes em alguma tabela para fazer outras operações, como a inserção em outro banco, por exemplo. Os dados nesse passo são adquiridos por meio de um comando SQL, mas o *Pentaho* possui uma interface gráfica para selecionar esses dados, sem necessidade do comando, se assim o usuário preferir.

*Sort rows* (na imagem acima com os nomes: Sort distrito, Sort municipio, Sort microrregiao, Sort mesorregião, Sort regiao, Sort distrito\_municipio, Sort municipio\_microrregiao, Sort microrregiao\_mesorregiao, Sort mesorregiao\_uf): Esse passo possibilita a ordenação de um conjunto de dados com base em uma coluna informada. Seu uso é semelhante ao comando *order by* do SQL. Nele podem ser configuradas outras opções como ordenação ascendente ou descendente e diferenciação de maiúsculas e minúsculas.

*Merge Join* (na imagem acima com os nomes: Merge distrito\_municipio, Merge municipio\_microrregiao, Merge microrregiao\_mesorregiao, Merge mesorregiao\_uf, Merge uf\_regiao): Com um funcionamento semelhante ao comando *JOIN* do SQL, esse *step* une dois fluxos de informação com base em uma coluna compartilhada (*key*), além de ser possível a configuração da forma de união (retornar apenas os dados que se relacionam ou também aqueles que não se relacionam). Requer o uso do *step* *Sort rows* antes deste para ordenação da coluna escolhida.

*Select values*: Este passo é utilizado para remover colunas, alterar o nome delas bem como seus tipos.

*Execute SQL Script*: Aqui nesse *step* o *Pentaho* permite executar um ou mais comandos SQL para fazer alguma operação no BD, seja uma consulta (*SELECT*) ou uma inserção (*INSERT*). Além disso, é possível utilizar variáveis criadas no próprio PDI no código.

*Table Output*: Conforme explicado na parte do *Staging*,esse passo realiza a carga dos dados estruturados em uma tabela no banco de dados. A tabela não precisa ser criada com antecedência, já que o PDI prepara um comando SQL automaticamente para a mesma ser criada.

O uso de todos os passos descritos anteriormente serão explicados adiante.

Descrição do processo de carga:

Este processo de carga inicia-se com a aquisição dos dados da tabela (*step Table input* com o nome ‘distritos’) que contêm as informações dos códigos de distritos (menor nível de informação, daí o nome da dimensão), dados estes inseridos previamente na seção 4.3 de montagem do *Staging*. Além desses códigos, são lidos também os códigos referentes aos municípios associados a cada distrito, na própria tabela de distritos e na tabela de municípios (*step Table input* com o nome ‘municipio’).

No momento que todas as informações são adquiridas, o próximo passo é ordená-las (*step Sort rows* com os nomes ‘Sort distrito’ e ‘Sort municipio’) de modo ascendente escolhendo a coluna que contêm os códigos dos municípios, esse passo de ordenação é necessário para o funcionamento correto do próximo passo.

Após as informações ordenadas, é feita a ‘união’ desses dois fluxos de informação (*step Merge Join* com o nome ‘Merge distrito\_municipio’) utilizando como coluna de união os códigos de município em cada um dos fluxos. Por exemplo: na tabela de distritos, um distrito de nome Lua Nova (cód. 521295610) possui nas suas informações o código de município 5212956, fazendo a união desse código na tabela de municípios, é encontrado esse código associado ao nome do município Matrinchã. Esse processo é feito para todos os distritos na tabela distritos no banco *Staging*.

A próxima tabela a ser acessada é a que contêm as informações dos códigos das Microrregiões (*step Table input* com o nome ‘microrregiao’). Como descrito no processo da tabela distrito, essa tabela foi carregada na seção 4.3 do banco de *Staging*. Em conjunto desses, no momento da carga da tabela anterior de municípios também foi adquirida as informações referentes aos códigos Microrregiões associadas a cada uma.

Tal como no processo anterior, após as informações serem adquiridas, elas são ordenadas (*step Sort rows* com os nomes ‘Sort microrregiao’ e ‘Sort distrito\_municipio’) de modo ascendente, agora utilizando a coluna com os códigos das Microrregiões como referência.

Após isso é feita a ‘união’ (*step* *Merge Rows* com o nome ‘Merge município\_microrregiao’) desses fluxos tal como o processo anterior, mas utilizando a coluna com o código das Microrregiões como forma de união. Por exemplo: continuando com o município anteriormente especificado (Matrinchã, cód. 5212956), ele possui em sua base o código de Microrregião 52002, que na tabela de Microrregião o código está associado ao nome Rio Vermelho.

O mesmo processo é aplicado a seguir, com as informações de Mesorregião sendo adquiridas (*step Table input* com o nome ‘mesorregiao’) e ordenadas (*step Sort rows* com os nomes ‘Sort mesorregiao’ e ‘Sort municipio\_microrregiao’) pelo seu respectivo código junto com as informações de mesorregião adquiridas na tabela de microrregião, sendo feita a sua união (*step* *Merge Rows* com o nome ‘Merge microrregiao\_ mesorregiao’) no final do processo.

Repetindo os processos anteriores, adquirem-se as informações dos códigos das UFs brasileiras (*step Table input* com o nome ‘uf’) e é feita a sua ordenação junto com o resultado da união anterior (*step Sort rows* com os nomes ‘Sort uf’ e ‘Sort microrregiao\_mesorregiao’) e posteriormente a sua união (*step* *Merge Rows* com o nome ‘Merge mesorregião\_uf’) com base nas informações dos códigos das UFs na ordenação anterior.

Por último, são adquiridas as informações sobre as regiões brasileiras (*step Data Grid* com o nome ‘regiao’), feita sua ordenação e da união anterior (*step Sort rows* com os nomes ‘Sort regiao’ e ‘Sort mesorregiao\_uf’) e a união desses resultados com base na coluna de código das regiões (*step* *Merge Rows* com o nome ‘Merge uf\_regiao’).

Finalmente, após todos os resultados serem retornados é usado o *step Select values* para remover as colunas redundantes geradas no momento das uniões, mantendo uma coluna para cada código e seus respectivos nomes, sendo ordenado logo após (*step Sort rows*) e inserido na sua dimensão de localidade (*step Table output*).

Indicadores nulos da dimensão:

Como explicado no início da seção, serão atribuídos valores para serem inseridos para caso o campo da informação, no momento da carga, for nulo. Para essa dimensão será usado o indicador ‘-1’ para caso em algum momento da carga essa informação estiver nula.

4.4.5.3.2 Dimensão Localidade Município

Para a segunda tabela, tem-se a Localidade Município. A tabela em questão vai apoiar as combinações da fato Aluno, composta por município, UF e país. Além do identificador único para cada combinação e indicadores nulos.

Seu diagrama de carga é mostrado na imagem abaixo:

[INSERIR IMAGEM]

Os seguintes passos foram utilizados:

*Table Input* (na imagem acima com os nomes: municipio, uf, pais): Como descrito na carga anterior, este *step* permite utilizar os dados já existentes em alguma tabela para fazer outras operações, como a inserção em outro banco, por exemplo. Os dados nesse passo são adquiridos por meio de um comando SQL, mas o *Pentaho* possui uma interface gráfica para selecionar esses dados, sem necessidade do comando, se assim o usuário preferir.

*Sort rows* (na imagem acima com os nomes: Sort municipio, Sort uf, Sort pais, Sort municipio\_uf, Sort pais\_uf, Sort cartesian): Como dito na carga anterior, esse passo possibilita a ordenação de um conjunto de dados com base em uma coluna informada. Seu uso é semelhante ao comando *order by* do SQL. Nele podem ser configuradas outras opções como ordenação ascendente ou descendente e diferenciação de maiúsculas e minúsculas.

*Merge Join* (na imagem acima com os nomes: Merge município\_uf, Merge pais\_uf): Explicado na carga anterior, possui um funcionamento semelhante ao comando *JOIN* do SQL. Esse *step* une dois fluxos de informação com base em uma coluna compartilhada (*key*), além de ser possível a configuração da forma de união (retornar apenas os dados que se relacionam ou também aqueles que não se relacionam). Requer o uso do *step* *Sort rows* antes deste para ordenação da coluna escolhida.

*Select values*: Como explicado na carga anterior, este passo é utilizado para remover colunas, alterar o nome delas bem como seus tipos.

*Execute SQL Script*: Conforme explicado anteriormente, nesse *step* o *Pentaho* permite executar um ou mais comandos SQL para fazer alguma operação no BD, seja uma consulta (*SELECT*) ou uma inserção (*INSERT*). Além disso, é possível utilizar variáveis criadas no próprio PDI no código.

*Add constants*: Neste passo é criado um fluxo constante de dados para serem inseridos junto com outro fluxo. Nele é possível configurar o nome da coluna que vai gerar esses dados, bem como os próprios dados a serem gerados.

*Join rows (cartesian product)*: Tem o funcionamento parecido com o *Merge Join*, mas no seu caso, ele é usado para multiplicar dois fluxos de informações criando todas as combinações possíveis entre eles, fazendo o chamado ‘produto cartesiano’.

*If Field Value is Null*: Aqui nesse passo o *Pentaho* permite inserir valores em campos que estão nulos. É possível escolher um valor padrão ou especificar valores para cada uma das colunas que chegam ao fluxo de dados.

*Table Output*: Conforme explicado na parte do *Staging*,esse *step* realiza a carga dos dados estruturados em uma tabela no banco de dados. A tabela não precisa ser criada com antecedência, já que o PDI prepara um comando SQL automaticamente para a mesma ser criada.

O uso de todos os passos descritos anteriormente serão explicados adiante.

Descrição do processo de carga:

Este processo de carga inicia-se com a aquisição dos dados dos códigos e nomes da tabela de Municípios (*step Table Input* com o nome ‘municipio’) carregadas na seção 4.3 de *Staging* junto com os códigos das Ufs associadas a eles, além dos dados dos códigos e nomes das UFs brasileiras (*step Table Input* com o nome ‘uf’). Junto com essa carga, é adicionado o código ‘76’ que é referente ao Brasil para ser carregado junto (*step Add constants*).

Ao final dessas duas cargas, são feitas suas respectivas ordenações (*step Sort rows* com os nomes ‘Sort municipio’ e ‘Sort uf’). Após suas ordenações concluídas, é feita a união dos dois fluxos de informações (*step Merge Join* com o nome ‘Merge município\_uf’) utilizando como coluna de união os códigos das UFs.

Junto da carga anterior, é feita a aquisição dos dados referentes aos códigos dos países (*step Table Input* com o nome ‘pais’) e sua ordenação ascendente pelo mesmo (*step Sort rows* com o nome ‘Sort pais’). Com os dois *steps* de ordenação prontos (‘Sort pais’ e ‘Sort município\_uf’) é feita a cópia de seus dados para cada um dos passos seguintes: o primeiro (*step Merge Join* com o nome ‘Merge pais\_uf’), faz a união dos dados dos municípios e UFs com o código ‘76’ que é referente ao país Brasil. O segundo (*step Join Rows (cartesian product)*), faz todas as combinações possíveis dos dados provenientes de municípios e UFs com os outros países, isso foi feito para ter as combinações dos alunos nascidos no exterior/naturalizados, que nasceram em outro país, mas que residem no Brasil. Ao final, os dois fluxos são mais uma vez ordenados (*step Sort rows* com o nome ‘Sort pais\_uf’ e ‘Sort cartesian’).

Após a finalização das ordenações anteriores, são removidas as colunas redundantes resultantes das uniões (*step Select values*) e checados os seus valores nulos (*step If field is null*), que nessa situação, serão os alunos estrangeiros que, na base do INEP, não possuem registros de UF/Município de nascimento/endereço, diferente dos alunos nascidos no exterior/naturalizados que possuem um país estrangeiro e registro de UF/Município de nascimento. Ao final é feita sua ordenação ascendente pelo código do país (*step Sort rows*) e sua inserção na respectiva dimensão no *Data Warehouse* (*step Table output*).

Como passo independente, ou seja, que pode ser executado antes ou depois da inserção dos dados no *DW*, tem-se a inserção dos indicadores nulos (*step Execute SQL script*) na dimensão de combinações de municípios. Esses indicadores serão detalhados adiante.

Indicadores nulos da dimensão:

Como citado anteriormente, essa dimensão possuirá os seguintes indicadores de informação nula:

-1: Caso todas as informações estiverem nulas. Esse indicador foi criado em duas combinações: Quando a informação de município, UF e país de origem for nula (-1, -1, -1); Quando o aluno tem como país de origem o Brasil, mas não possui informações referentes ao seu município e sua UF (-1, -1, 76).

-2: Caso o aluno for estrangeiro. Esse indicador foi criado em uma combinação: Quando o aluno tiver como país de origem qualquer valor diferente de 76 (que faz referência ao Brasil) e suas informações de UF e município não estiverem disponíveis, por exemplo, se o aluno for natural dos Estados Unidos: (-2, -2, 840).

-3: Caso o aluno for naturalizado/nascido no exterior. Esse indicador foi criado em uma combinação: Quando o aluno for naturalizado/nascido no exterior e suas informações de município e UF não estiverem disponíveis (-3, -3, 76).

4.4.5.4 Dimensão Escola

Para a carga da dimensão das escolas, a ordem de ações consiste na extração dos dados dos dois tipos de escolas (públicas e privadas), transformação dos dados inserindo os indicadores de nulo dos dois tipos de escolas, remover as informações duplicadas e posterior inserção delas no *Data Warehouse*, como segue:

[INSERIR IMAGEM]

Os passos estão descritos a seguir:

*Table Input*: Este *step* permite utilizar os dados já existentes em alguma tabela para fazer outras operações, como a inserção em outro banco, por exemplo. Os dados nesse passo são adquiridos por meio de um comando SQL, mas o *Pentaho* possui uma interface gráfica para selecionar esses dados, sem necessidade do comando, se assim o usuário preferir. A utilização de dois passos será explicada adiante.

*If Field Value is Null*: Aqui nesse passo o *Pentaho* permite inserir valores em campos que estão nulos. É possível escolher um valor padrão ou especificar valores para cada uma das colunas que chegam ao fluxo de dados. Seus dois usos nessa ETL serão explicados adiante.

*Dummy (do nothing)*: Esse *step* não realiza ações, mas pode ser utilizado para unir diferentes fluxos de dados ou analisar os dados que estão sendo recebidos. Nessa ETL, ele está juntando os dois fluxos de dados para centralizar a inserção.

*Sort rows*: Como dito na carga anterior, esse passo possibilita a ordenação de um conjunto de dados com base em uma coluna informada. Seu uso é semelhante ao comando *order by* do SQL. Nele podem ser configuradas outras opções como ordenação ascendente ou descendente e diferenciação de maiúsculas e minúsculas.

*Unique rows*: Esse *step* é utilizado para eliminar dados que porventura venham duplicados no fluxo de carga, além disso, podem ser configurados contadores para a quantidade de duplicatas encontradas e redirecionamento delas para outro passo. Requer o uso do *step* *Sort rows* antes deste para ordenação da coluna escolhida.

*Database lookup*: Esse passo permite a comparação de um ou mais valores vindos de um fluxo de dados com uma ou mais colunas inseridas em uma tabela no banco de dados. Essa comparação, quando verdadeira, retorna uma coluna específica no banco, que pode ser renomeado. Além disso, podem ser feitas configurações para o uso de cache, que em dadas situações podem melhorar a performance da carga. Seu uso nessa ETL será explicado adiante.

*Select values*: Como explicado na carga anterior, este passo é utilizado para remover colunas, alterar o nome delas bem como seus tipos.

*Replace In String*: Esse passo permite que o PDI possa substituir um valor de algum campo por outro valor especificado. Esse campo pode ser especificado diretamente ou por meio de uma expressão regular, após, é informado o valor de procura e depois o novo valor, que também pode estar em outro campo.

*Table Output*: Conforme explicado na parte do *Staging*,esse passo realiza a carga dos dados estruturados em uma tabela no banco de dados. A tabela não precisa ser criada com antecedência, já que o PDI prepara um comando SQL automaticamente para a mesma ser criada.

Descrição do processo de carga:

Primeiramente, são feitas as aquisições dos dados referentes às escolas em duas partes, o primeiro (*step Table input*) procura as escolas públicas e o segundo (*step Table input 2*) procura as escolas privadas conforme a coluna TP\_DEPENDENCIA de cada um deles. Caso a escola tiver os códigos 1, 2, 3 ela é considerada uma escola pública por ter dependência nas esferas federal, estadual ou municipal, respectivamente, ou ela possui o código 4 referente a uma escola privada.

Após a pesquisa dos dados tem-se a parte de transformação, em que os próximos passos substituem os valores nulos por dois tipos de indicadores. Na pesquisa de escolas públicas, é feita a substituição dos dados (*step* *If Field Value is Null*) acerca dos mantedores de escolas privadas e da categoria privada da mesma, já que por ser uma escola pública, esses indicadores não se aplicam a ela e sempre estarão nulos, além da substituição quando essa informação estiver vazia. No outro passo onde é feita a pesquisa de escolas privadas, é feita a substituição de todos os valores nulos (*step* *If Field Value is Null 2*). Os indicadores nulos dessa dimensão serão explicados adiante.

Com as substituições concluídas, o *Dummy* é utilizado como o *step* que recebe todo esse fluxo de dados para centralizá-los e enviar para o próximo *step* em que é feita a sua ordenação (*step Sort rows*), e após ser ordenado, é feita remoção das escolas duplicadas (*step Unique rows*) para manter uma lista única com todas as escolas envolvidas na análise.

Tendo os dados duplicados removidos, é feita a procura (*step Database lookup*) do código referente a cada combinação de região, distrito, microrregião, mesorregião, uf e município na tabela D\_LOCALIDADE\_DISTRITO, carregada anteriormente para apoiar essas combinações. Quando uma combinação é encontrada com sucesso, é retornado para o fluxo o código referente a essa combinação.

Com todas as combinações encontradas na dimensão de localidade distrito, é feita a substituição dos dados (*step Replace in string*) de algumas colunas com informações referentes às escolas pelo o seu significado segundo o dicionário de dados do INEP. Por exemplo, a coluna IN\_AGUA\_INEXISTENTE indica se a escola possui ou não abastecimento de água, no fluxo, os dados existentes nessa coluna são 0 para ‘Não’ e 1 para ‘Sim’, assim, esse *step* faz essa substituição do valor numérico pelo seu valor de significado. Ao finalizar, os dados são mais uma vez ordenados e inseridos na tabela da dimensão escolar.

Indicadores nulos da dimensão:

Como citado anteriormente, essa dimensão possuirá os seguintes indicadores de informação nula:

-1: Caso alguma informação estiver nula.

-2: Inserido nas colunas referentes aos mantedores privados e na categoria de escola privada, quando a escola for pública, já que essas duas informações não se aplicam a uma escola que é pública.

4.4.5.5 Fato Aluno

Agora na última tabela do *Data Warehouse*, tem-se a fato aluno que é gerada após todas as dimensões estiverem prontas no BD.

Na sua carga, o processo é semelhante à carga da dimensão escolas, com o diferencial da substituição dos anos da análise por seus respectivos códigos definidos na dimensão ano no momento da carga.

[INSERIR IMAGEM]

Os passos estão descritos a seguir:

*Table Input* (na imagem acima com os nomes: Table input brasileiros, Table input naturalizados, Table input estrangeiros): Este *step* permite utilizar os dados já existentes em alguma tabela para fazer outras operações, como a inserção em outro banco, por exemplo. Os dados nesse passo são adquiridos por meio de um comando SQL, mas o *Pentaho* possui uma interface gráfica para selecionar esses dados, sem necessidade do comando, se assim o usuário preferir. A utilização de três passos será explicada adiante.

*If Field Value is Null*: Aqui nesse passo o *Pentaho* permite inserir valores em campos que estão nulos. É possível escolher um valor padrão ou especificar valores para cada uma das colunas que chegam ao fluxo de dados. Seus três usos nessa ETL serão explicados adiante.

*Dummy (do nothing)*: Esse *step* não realiza ações, mas pode ser utilizado para unir diferentes fluxos de dados ou analisar os dados que estão sendo recebidos. Nessa ETL, ele está juntando os três fluxos de dados para centralizar a inserção.

*Replace In String*: Esse passo permite que o PDI possa substituir um valor de algum campo por outro valor especificado. Esse campo pode ser especificado diretamente ou por meio de uma expressão regular, após, é informado o valor de procura e depois o novo valor, que também pode estar em outro campo.

*Database lookup*: Esse passo permite a comparação de um ou mais valores vindos de um fluxo de dados com uma ou mais colunas inseridas em uma tabela no banco de dados. Essa comparação, quando verdadeira, retorna uma coluna específica no banco, que pode ser renomeado. Além disso, podem ser feitas configurações para o uso de cache, que em dadas situações podem melhorar a performance da carga. Seu uso nessa ETL será explicado adiante.

*Select values*: Como explicado na carga anterior, este passo é utilizado para remover colunas, alterar o nome delas bem como seus tipos.

*Sort rows*: Como dito na carga anterior, esse passo possibilita a ordenação de um conjunto de dados com base em uma coluna informada. Seu uso é semelhante ao comando *order by* do SQL. Nele podem ser configuradas outras opções como ordenação ascendente ou descendente e diferenciação de maiúsculas e minúsculas.

*Table Output*: Conforme explicado na parte do *Staging*,esse passo realiza a carga dos dados estruturados em uma tabela no banco de dados. A tabela não precisa ser criada com antecedência, já que o PDI prepara um comando SQL automaticamente para a mesma ser criada.

Descrição do processo de carga:

Primeiramente, são feitas três pesquisas de dados. A primeira (*step Table input* com o nome ‘Table input brasileiros’) procura os estudantes brasileiros, a segunda (*step Table input* com o nome ‘Table input naturalizados’) procura os estudantes naturalizados, a terceira (*step Table input* com o nome ‘Table input estrangeiros’) procura os estudantes estrangeiros, conforme a coluna CO\_PAIS\_ORIGEM de cada um deles. Caso o aluno possuir o código 76 nessa coluna, significa que ele tem nacionalidade brasileira/naturalizado (esse é o código do Brasil na tabela de países do INEP). Caso contrário, ele é definido como estrangeiro.

Após a pesquisa dos dados tem-se a parte de transformação, em que os próximos passos (*step If field value is null*) substituem os valores nulos por até três tipos de indicadores, esses indicadores serão explicados adiante.

Com as substituições concluídas, é feita a junção dos três fluxos de dados (*step Dummy (do nothing)*) e envio para o próximo passo (*step Replace in string*) que faz a substituição dos dados relativos aos anos da análise (2015, 2016, 2017 e 2018) pelos seus códigos definidos na tabela dimensão ano (1, 2, 3 e 4, respectivamente), além de indicadores referentes ao sexo do aluno, cor/raça, nacionalidade, entre outros.

Ao ser feitas as substituições, são feitas as comparações das combinações de município, UF e país de origem (tanto como nascimento e endereço), com o seu equivalente na dimensão D\_LOCALIDADE\_MUNICIPIO (*step Database lookup e Database lookup 2*), essa combinação ao ser verdadeira, retorna o código associado a ela existente na dimensão.

Concluindo esse passo, é feita a remoção das colunas com os códigos das UFs, municípios e país de origem (tanto como nascimento e endereço) para manter apenas o código referente à combinação dessas três informações para que possa ser ligada a dimensão D\_LOCALIDADE\_MUNICIPIO, após é feita a sua ordenação e finalmente a carga dessas informações na tabela fato dos alunos (*step Table output*).

Indicadores nulos da dimensão:

Como citado anteriormente no *step If field value is null* essa dimensão possuirá os seguintes indicadores de informação nula:

-1: Caso alguma informação estiver nula.

-2: Indicador usado para representar uma informação nula nas colunas de UF e município de um aluno que é estrangeiro.

-3: Indicador usado para representar uma informação nula nas colunas de UF e município de um aluno que é naturalizado.

Nessa última carga, são finalizadas todas as dimensões e a fato referente ao ambiente de BI do presente trabalho, com o banco de dados se apresentando da forma segundo a imagem abaixo:

[INSERIR IMAGEM]

Sua diagramação se apresenta da forma segundo a imagem abaixo:

[INSERIR IMAGEM]