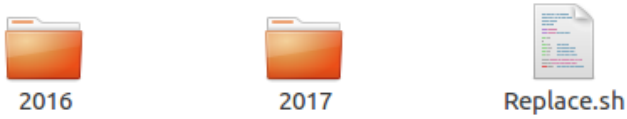
Documentação das Transformações das bases de Servidores

Alysson Gomes de Sousa

Pré-processamento dos arquivos

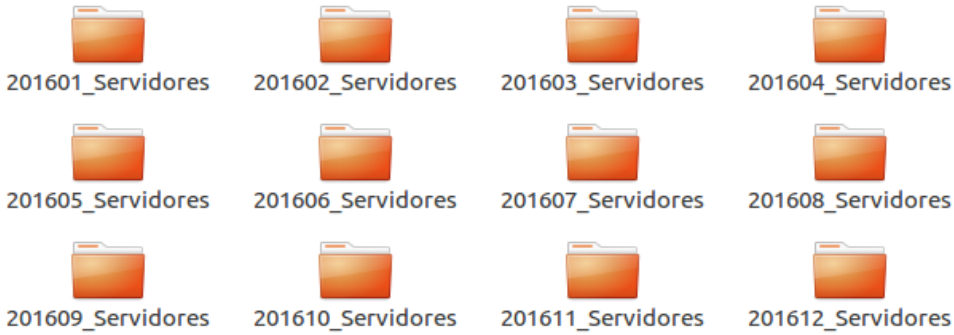
1. A primeira parte do pré-processamento é subtituir os caracteres de tabulação (\t) por cerquilha (#), para isso, usamos o arquivo Replace.sh (https://github.com/npi-ufc-qxd/dados-abertos/blob/master/Scripts/Replace.sh). Para utilizá-lo, devemos organizar os arquivos em diretórios separados por anos, como mostra a Figura 1.

Figura 1 – Organização dos anos



Os diretórios relativas aos meses estão organizadas na forma como mostra a Figura 2.

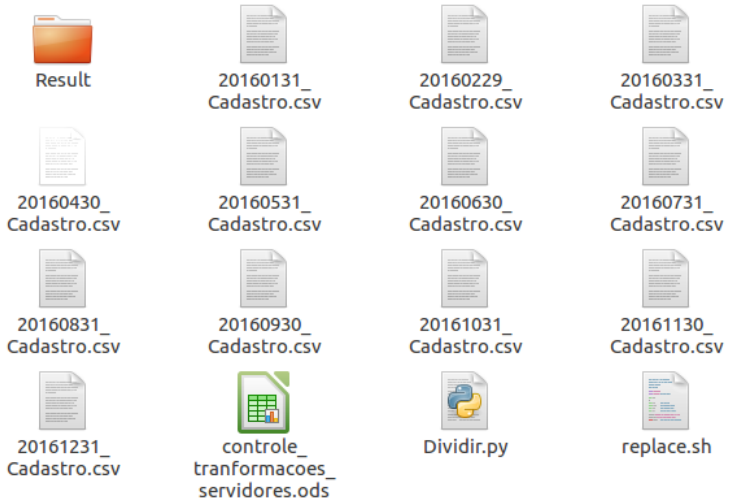
Figura 2 – Meses de 2016



Comando para executar o arquivo Replace.sh: `sh pasta_corrente/Replace.sh`

2. A segunda etapa do processo de pré-processamento é a quebra dos arquivos, porque o Pentaho não suporta o tamanho dos arquivos originais. Para isso, reunimos todos os arquivos que serão divididos em um único diretório, e criamos uma pasta com o nome Result (para guardar as partições dos arquivo), como mostra a Figura 3.

Figura 3 – Organização do diretório de partições

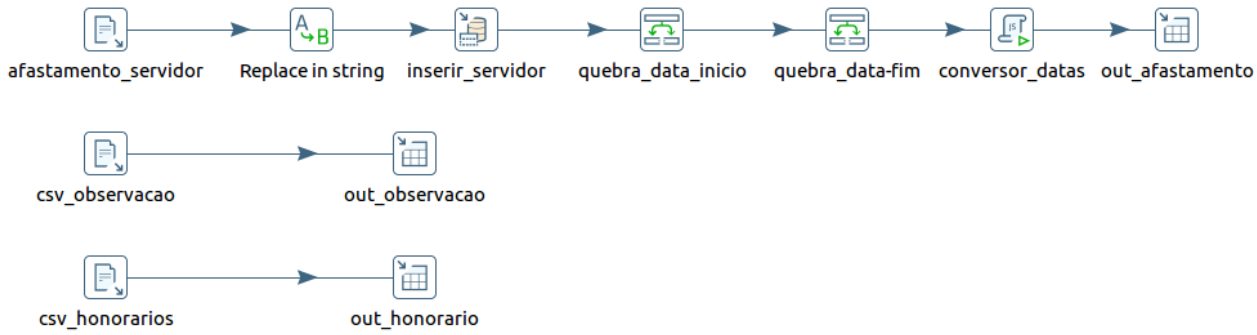


Em seguida utilizamos o arquivo Dividir.py (https://github.com/npi-ufc-qxd/dados-abertos/blob/master/Scripts/Dividir.py), que deve ser colocado no diretório junto com os arquivos serão divididos, como mostra a Figura 3, por fim, o arquivo Dividir.py deve ser executado (com o comando `python3 pasta_corrente/Dividir.py`) com o python na versão 3, logo após a execução, os arquivos particionados estarão no diretório Result.

TransformacaoObservacaoAfastamentoHonorarios.ktr

Esta transformação destina-se a migração dos dados dos arquivos de Observação, Afastamento e Honorários. A seguir descreveremos a função de cada passo das transformações, como mostra a figura a seguir.

Figura 4 – Transformação de Observação, Afastamento e Honorários



afastamento_servidor, csv_observacao, csv_honorarios. Estes três *steps* são responsáveis por carregar os arquivos de afastamento.csv, observação.csv e honorarios.csv, dividindo-os pelo limitador "#".

Replace in string. Em alguns casos, os campos "DATA_FIM_AFASTAMENTO" e "DATA_INICIO_AFASTAMENTO" possuem o valor "Não informada", porque não possuem as datas, por isso esse *step* substitui o valor "Não informada" pelo valor "00/00/0000".

inserir_servidor. Esse *step* insere na tabela "servidor" os dados do servidor contidos na linha do csv, caso ele ainda não esteja na tabela.

quebra_data_inicio, quebra_data-fim. Estes *steps* são responsáveis por quebrar a *string* dos campos "DATA_INICIO_AFASTAMENTO" e "DATA_FIM_AFASTAMENTO", respectivamente, em três colunas contendo o dia, o mês e o ano de cada data.

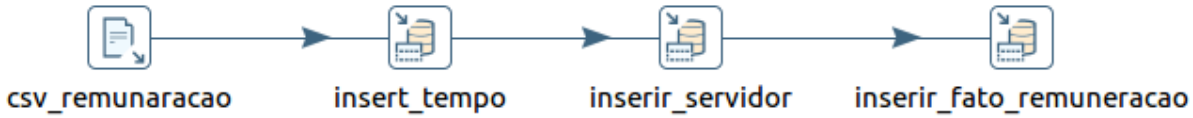
conversor_datas. Esse *step* do tipo *script* junta as colunas vindas dos *steps* anteriores em uma *string* no formato de uma data válida para ser inserido no banco de dados, a saber, mês/dia/ano.

out_afastamento, out_observacao, out_honorario. Estes *step* são responsáveis por receber os dados válidos dos *steps* anteriores, e inseri-los nas tabelas afastamento, observação e honorarios, respectivamente.

TransformacaoRemunaracao.ktr

Esta transformação destina-se a migração dos dados de remuneração dos servidores públicos, presentes no arquivo de Remuneração.

Figura 5 – Transformação Remuneração



csv_remuneracao. Este é responsável por carregar os arquivo o remuneracao.csv, dividindo-o pelo limitador “#”.

insert_tempo. Este *step* é responsável por inserir na tabela tempo o ano e mês do arquivo que está sendo processado.

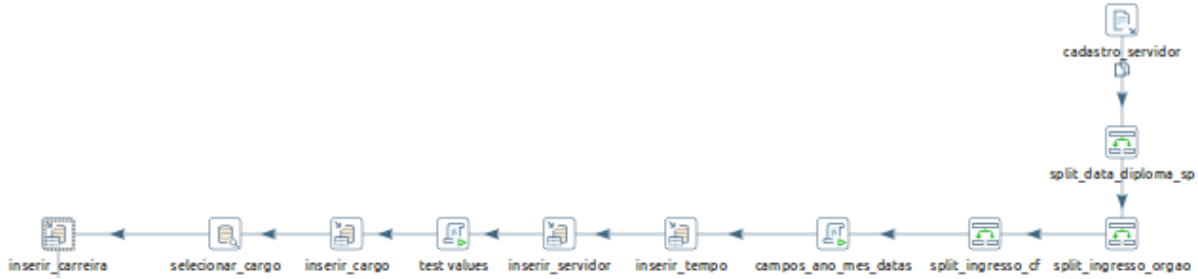
inserir_servidor. Esse *step* insere na tabela “servidor” os dados do servidor contidos na linha do csv, caso ele ainda não esteja na tabela.

inserir_fato_remuneracao. Esse *step* insere na tabela “fato_remuneracao” os dados válidos vindos dos *steps* anteriores, caso ele ainda não esteja na tabela.

TransServidor_modelo_hibrido.ktr

Esta transformação destina-se a migração dos dados relativo cadastro pessoal dos servidores públicos, como: nome, cpf, cargo, função, vinculo, órgão que ele está vinculado, etc. A primeira parte da transformação é formado pelos *steps* mostrados na Figura 6.

Figura 6 – Primeira parte da Transformação



cadastro_servidor. Este é responsável por carregar as partições do arquivo de cadastro, dividindo-o pelo limitador “#”.

split_data_diploma, **split_ingresso_orgao**, **split_ingresso_cf**. Este *steps* são responsáveis por quebrar a *string* dos campos “DATA_DIPLOMA_INGRESSO_SERVICOPUBLICO”,

“DATA_INGRESSO_ORGAO” e “DATA_INGRESSO_CARGOFUNCAO”, respectivamente, transformando cada campo em 3 colunas (dia, mês e ano).
campos_ano_mes_datas. Este *step* contém os atributos de ano e mês do arquivo csv, que são usado durante toda a transformação. Obs: os atributos ANO e MES devem ser atualizados conforme os meses que forem inseridos, por exemplo: ano=2017 e mes=10, para o arquivo csv correspondente ao mês de Outubro do ano de 2017.

inserir_tempo. *Step* responsável por inserir as tuplas de tempo, caso o mês e o ano corrente ainda não estejam na tabela tempo.

inserir_servidor. Esse *step* insere na tabela “servidor” os dados do servidor contidos na linha do csv, caso ele ainda não esteja na tabela.

test_values. *Step* que testa se os campos ***DESCRICAO_CARGO***, ***CLASSE_CARGO***, ***REFERENCIA_CARGO*** e ***NIVEL_CARGO*** não estão vazios, caso esteja, será colocado uma *string* vazia (‘’).

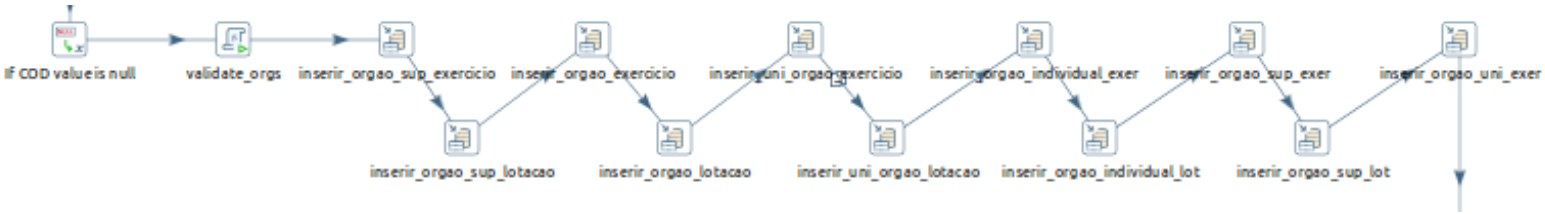
inserir_cargo. Esse *step* insere na tabela “cargo” os dados do cargo do servidor contidos na linha do csv, caso ele ainda não esteja na tabela.

selecionar_cargo. Este *step* realiza uma consulta no banco para retornar o código do cargo vindo do csv, pois o csv possui apenas o nome do cargo.

inserir_carreira. Este *step* inseri na tabela carreira os seguintes dados referentes ao *status* da carreira do servidor: ***COD_CARGO (***recebido do *step selecionar_cargo****),*** ***CLASSE_CARGO***, ***REFERENCIA_CARGO*** e ***NIVEL_CARGO.***

A segunda parte da transformação é composta pelos *steps* mostrados na Figura 7.

Figura 7 – Segunda parte da transformação



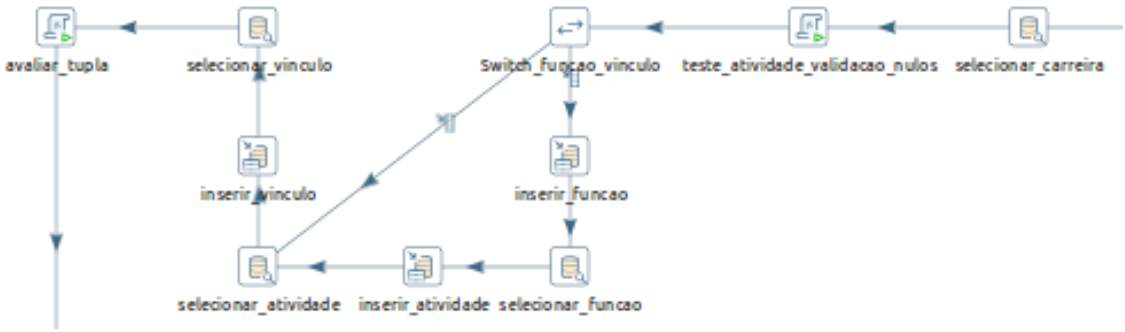
if COD value is null. Esse *step* verifica se os campos *COD_ORGSUP_EXERCICIO, COD_ORGSUP_LOTACAO, COD_ORG_LOTACAO, COD_ORG_EXERCICIO, COD_UORG_EXERCICIO, COD_UORG_LOTACAO estão vazios.*

validate_orgs. Esse *step* verifica se os campos *COD_ORGSUP_EXERCICIO, COD_ORGSUP_LOTACAO, COD_ORG_LOTACAO, COD_ORG_EXERCICIO, COD_UORG_EXERCICIO, COD_UORG_LOTACAO* estão vazios, se estiverem substitui os valores por 0. Além disso, se houver códigos com letras, ele substituirá o valor por um código totalmente numérico (esse processo é feito pela função ‘*hasing(string)’*).

inserir_orgao_sup_exercicio, inserir_orgao_sup_lotacao, inserir_orgao_exercicio, inserir_orgao_lotacao, inserir_uni_orgao_exercicio, inserir_uni_orgao_lotacao, inserir_orgao_individual_exer, inserir_orgao_individual_lot, inserir_orgao_sup_exer, inserir_orgao_sup_lot, inserir_orgao_uni_exerc, inserir_orgao_uni_lot. Todos estes *steps* são responsáveis por inserir os órgãos na tabela ORGAO, ORGAO_INDIVIDUAL, ORGAO_SUPERIOR, UNIDADE_ORGAO (existem 12 colunas no csv que estão relacionadas a órgão, no entanto, todas elas irão para as mesmas tabelas).

A terceira parte da transformação é formada pelos *steps* mostrados na Figura 8.

Figura 8 – Terceira parte da transformação



selecionar_carreira. Esse *step* consulta no banco um determinada carreiram, a partir dos seguintes campos: ***COD_CARGO,CLASSE_CARGO***, ***REFERENCIA_CARGO*** e ***NIVEL_CARGO.***

teste_atividade_validacao_nulos. Este *step* contém um *script* que verifica se o código e o nome da atividade do servidor estão vazios, caso estejam, ele irá substituir pelo código zero (0) e o nome receberá uma *string* vazia (‘’). Caso não esteja, o *script* verificará se o código possui letras, caso tenha, será substituído por um código totalmente numérico. Além disso, o *script* verificará se os campos ***sigla,funcao, nivel_funcao,opcao_parcial, REGIME_JURIDICO, UF_EXERCICIO, JORNADA_DE_TRABALHO e DIPLOMA_INGRESSO_ORGAO*** estão vazios, caso estejam, serão substituídos por uma *string* vazia.

Switch_funcao_vinculo. Este *step* é responsável por determinar a direção do fluxo da transformação a partir da função do servidor. Caso o servidor possua função, então deveremos inserir a função(inserir_funcao) e atividade(inserir_atividade) vinculada a esta função (selecionar_funcao), caso ele não tenha função, a transformação deve incaminhar-se para seleção da atividade(selecionar_atividade) e em seguinda o seu vinculo (inserir_vinculo).

inserir_funcao. Esse *step* insere na tabela “funcao” os dados da função do servidor contidos na linha do csv, caso ele ainda não esteja na tabela.

selecionar_funcao. Este *step* realiza uma consulta no banco para retornar o código da função vindo do csv, pois o csv possui apenas o nome da função.

inserir_atividade. Esse *step* insere na tabela “atividade” os dados da atividade do servidor contidos na linha do csv e que estão vinculados a função, caso ele ainda não esteja na tabela.

selecionar_atividade. Este *step* realiza uma consulta no banco para retornar o código da atividade vindo do csv, pois o csv possui apenas o nome da função.

inserir_vinculo. Esse *step* insere na tabela “vinculo” os dados do vinculo (tipo e situação) do servidor contidos na linha do csv, caso ele ainda não esteja na tabela.

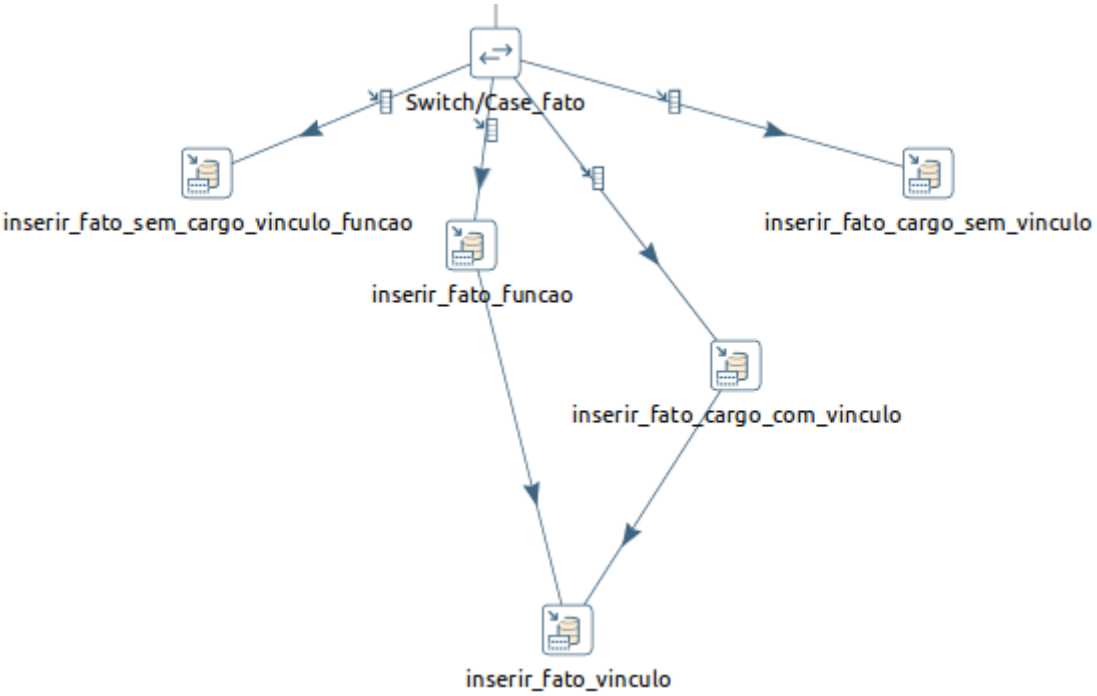
selecionar_vinculo. Este *step* realiza uma consulta no banco para retornar o código do vinculo vindo do csv, pois o csv possui apenas o nome do vinculo.

avaliar_tupla. Este *step* possui um *script* que avalia em qual das tabelas de fato (fato_funcao, fato_cargo, fato_vinculo, fato_sem_cargo_funcao_vinculo), uma determinada tupla (vinda do csv) deve ser registrada. A

script verifica se tupla do csv possui os dados requeridos pelas tabelas de fato, por exemplo: se a tupla possui informações de cargo, então a tupla deve ser registrada em fato_cargo, ou se a tupla possui dados de vinculo, então deve ser feito um registro na tabela fato_vinculo. O resultado da avalição é guardado na variável ***`result`***.

A quarta parte da transformação é formada pelos *steps* mostrados na Figura 9.

Figura 9 – Quarta parte da transformação



Switch/Case_fato. A partir da variável ***`result`*** do *step* **avaliar_tupla**, este *step* determinará qual o caminho que a tupla deverá trilhar, por exemplo: *SE* ***`result == 'F'`*** *ENTÃO* a tupla deve seguir para o *step* **inserir_fato_funcao**.

Obs[1]: As tuplas que possuem dados de função, necessariamente terão dados de vínculo.

Obs[2]: As tuplas que possuem dados de cargo, não necessariamente terão dados de vínculo, o que justifica os *steps* **inserir_fato_cargo_com_vinculo (*`result=='V'`*)** e **inserir_fato_cargo_sem_vinculo (*`result=='C'`*)**.

inserir_fato_sem_cargo_vinculo_funcao. Insere na tabela fato_sem_cargo_vinculo_funcao a tupla que não possui dados de cargo, nem de vínculo e nem de função.

inserir_fato_funcao. Insere na tabela fato_funcao a tupla que possui dados de função do servidor.

inserir_fato_vinculo. Insere na tabela fato_vinculo a tupla que possui dados de vínculo do servidor.

inserir_fato_cargo_com_vinculo. Insere na tabela fato_cargo a tupla que possui dados de cargo do servidor e que também possui dados de vínculos.

inserir_fato_cargo_sem_vinculo. Insere na tabela fato_cargo a tupla que possui dados de cargo do servidor, mas não possui dados de vínculos.