**ETL - EXTRAIR, TRANSFORMAÇÃO, CARREGAR**

4. instalando o pentaho data integration (PDI)

**AULA 13**

-> STEPS(SÃO AS FUNÇÕES) E HOPE(AS LIGAÇÕES)

**AULA 14**

-> DEFINIÇÕES DO INPUT (CSV FILE INPUT)

-Step name (nome do step)

-Filename (local do arquivo que quero extrair)

-Delimiter (demilitador, especifica o caractere separador do campo)

-Enclosure (especifica um caractere delimitador, geralmente são as aspas)

-NIO buffer size (define o tamanho do buffer de leitura)

-Lazy conversion (mantém a conversão dos dados para um momento posterior)

-Header row present (identifica os cabeçalhos dos arquivos)

-Add filename to result (add o nome dos arquivos aos arquivos usados na transformação)

-The row number field name (inclui o número de linha no campo interger no fluxo de dados)

-Running in parallel (ler partes do mesmo arquivo em paralelo)

-File encoding (tipo de caractere, mais utilizado UTF-8)

-> STEP DUMMY (DO NOTHING)

--Funções Dummy

-saída de dados

-pré-visualização de dados

**AULA 15 INPUT**

-> Definições do input (txt file input)

--Aba File

-Nome do Step

-File or directory (selecionar o arquivo e depois adicionar)

--Aba Content

-Filetype (CSV, por conta que csv é um arquivo texto)

-Separetor (;)

-Enclousure (")

-Header (marcar, porque tem a primeira linha como cabeçalho)

-Compression (para comprimir o arquivo, deixar none se não for)

-No empty rows (não trazer linhas vazias)

-The date format Locale (pt\_BR)

--Aba Fields

-obter campos

-preview

--obs: Quando a aba da tranformação esta em negrito é por conta que o arquivo não esta salvo.

**AULA 16 - INPUT EXCEL**

--Aba Files

-Nome do step

-Spread sheet type (engine) - tipo de excel

-file or directory

--Aba !Sheets

-Get sheetname(s)... (ele abre um pop-up, selecione a pasta clientes e adicionar, ok)

-- Aba !Fields

-Get Fields from header row...

-trim type (os dois lados) limpar dados

**AULA 17 - INPUT MULTIPLE FILES EXCEL**

-Pegar caminho onde estão os arquivos ( neste caso C:\Material\_PDI\Inputs\Arquivos\_excel\Files)

-Spread Sheet type (Apache POI Streaming)

-Exclude Regular Expression (expressão regular - ^count.+)

-- Aba !Fields

-Get Fields from header row...

-deletar as colunas repetidas

**AULA 18 - INPUT ACCESS**

--Aba File

-Nome do step (Produto)

-File or directory (browse - local)

--Aba Content

-Table (selecionar a tabela que irei utilizar, no caso produtos)

--Aba Fields

-Get Fields

-Preview

**AULA 19 - INPUT XML**

-Escolher Get data from XML

--Aba File

-file or directory (escolher caminho e arquivo xml)

--Aba Content

-Loop XPath (Get XPath nodes, selecionar Deputados/Deputado para me trazer todos as colunas)

--Aba Fields

-get fieds

**AULA 20 - INPUT JSON**

salvar transformação

-JSON input, Dummy e ligar o step com o shift

abrir o input

--Aba File

-nome step (MoedaMundial)

-browse (achar o arquivo pelo caminho)

--Aba Fields

temos que saber que dados tem no arquivo

abrir o arquivo JSON (abrir com bloco de notas)

-name (por o nome da coluna)

-Path ($..nome\_da\_coluna\_name, mesmo nome)

**AULA 21 - DATA GRID**

-step data grid e dummy

-abrir data grid

para que serve o

-criar dados estaticos, criar dentro do PDI os dados estaticos

-aba meta são as colunas da tabela

-data são os dados da tabela

--Aba Meta

-Name -type

Cliente String

EstadoCivil String

Sexo String

**AULA 22 - TABLE INPUT**

-STEP Table input, dummy

antes de confiturar, abrir pgadmin

no pgadmin, criar o banco de dados estudos

dentro de estudos com o botão direito escolher a opção restore

->obs: dentro da aula existe uma solução alternativa caso dê erro

abrir o table input

--new...

-Connection name (conn\_estudo)

escolher o banco "PostgreSQL"

-Host Name (localhost)

-Database Name (nome do banco de dados que estamo utilizado, no caso, "estudos"

-Port Number (vem por padrão, 5432)

-Username (postgres)

-Password (123456)

--Get SQL select statement (verificar as tabelas que tenho no meu banco)

**AULA 23 - OUTPUT EXCEL**

usar table input e criar a conexão com estudos

usar o step output excel

--Aba File

-filename (colocar o nome de saida e procurar o destino em navegar e quando selecionar, apagar a extensão, porque ja tem na opção de baixo)

--Aba Fields

-obter campos

excutar transformação

**AULA 24 - OUTPUT EXCEL WRITER**

escrita com arquivo excel

-step csv file input, excel writer

input csv orçamento

--Step Excel Writer

--Aba File & Sheet

-filename (escolher o caminho do arquivo na pasta)

-extensão (escolher a extenção do arquivo)

-If output file exists (se ja existe o arquivo, selecionar "use existing file for writing")

-Sheet name (nome da aba na planilha, no caso, Plan1)

-If sheet exists in output file (escrito no arquivo existente, write to existing sheet)

--Aba Content

-Auto Size Column (marcar)

-Write Header (deixar marcado)

-Start writing at cell A1 (começar a escrever a partir da celula A1)

**AULA 25 - OUTPUT CSV E ACCESS**

usar 2 Table input, text file output, access output

pegar a conexão ja feita no da aula table input

ir em File -> Open -> abrir input Table

ir em View -> Conexões ->botão direito na conexão conn\_estudo e selecionar "share" (compartilhar)

agora pegar a conexão na primeira table input e pegar a tabela categoria

abrir o step text file output

--Aba File

-nome step (saida categoria)

-filename (local onde vai ser a saida do arquivo e no final digitar \nome\_arquivo)

-Extensão (csv)

agora pegar a conexão da segunda table input e pegar a tabela subcategoria

abrir o step microsoft access output

--Aba File

-nome step (saida subcategoria)

-filename (local onde vai ser a saida do arquivo)

-target table (nome da tabela, subcategoria)

**AULA 26 - OUTPUT TEXT FILE**

-step table input, text file output

-pegar a tabela produto na table input

--Text File Output

-selecionar onde vai ser a saida do arquivo (\nome\_arquivo)

-Extensão ja vem como padrão txt

**AULA 27 - OUTPUT XML E JSON**

abrir pgadmin

criar um novo banco de dados chamados "dados\_publicos" e restaurar o banco com as tabelas

criar uma nova transformação

-step table input, json output, xml output

usar 1 input para dois caminhos de saida

-step table input

--Criar uma nova conexão

-new...

-Connection name (daddos\_publicos)

-Host Name (localhost)

-Databases Name (dados\_publicos)

-Port Number (5432)

-Username (postgres)

-Password (123456)

-get SQL

-trazer a tabela "cgu"

usar apelidos para as clunas

-as id\_orgao

-as nome\_orgao

-as ano

-as mes

-as favorecido

-as data\_transacao

-as valor\_transacao

ligar as duas saidas

--step JSON OUTPUT

--Aba General

-Nr rows in a bloc (mudar para 0, para não criar um arquivo a cada linha populada da tabela)

-filename (local onde o arquivo vai ser salvo e escolher o nome do arquivo)

--Aba Fields

-get fields

--step XML OUTPUT

--Aba File

-filename (local que vai ser armazenado com o nome do arquivo)

**AULA 28 - TABLE OUTPUT**

-Step CSV file input, table output

step CSV file input

empregados (pegar o arquivo CSV empregados, usar o delimitador ;)

step table output

-connection (conn\_estudo)

-target table (empregados)

-SQL (criar SQL para popular a tabela, exucutar a consulta, ja cria automaticamente o sql da consulta)

**AULA 29 - OUTPUT UPDATE X INSERT/UPDATE**

criar nova transformação

steps excel input, table output, update, insert/update

quando ligar os step escolher a opção de copia para que todos os dados sejam copiados, se escolher distribuir o input vai distrinuir os dados entre as ligações

-step excel

--aba files

-nome step (vendedores)

-spread (excel 2007)

-navegar (procurar vendedores e add)

--aba sheets

-get sheetnames

--aba fields

-get fields

--step update

-target table (vendedores)

-get fields (comparar com um interger, deixar o id e apagar os outros campos)

-update fields ( deixar somente os campos que vão ser alterados)

--step inset/update

-connection (conn\_estudo)

-target (vendedores)

-get fields (deixar somente o id de comparação)

-get updates fields (pode marcar se quer o update ou não)

**-AULA 30 - TRANSFORMAÇÃO DE DADOS**

**AULA 31 - ALTERANDO E REMOVENDO COLUNAS -** SELECT VALUES (alterar memoria pentaho)

se tiver 8gb de memoria ram e o pentaho estiver lento.

vai no local do arquivo bat e botão direito, editar.

procurar um quadrado onde tem um monte de asterisco e logo abaixo set java runtime.

abaixo dele tem um if.

if "%PENTAHO\_DI\_JAVA\_OPTIONS%"=="" set PENTAHO\_DI\_JAVA\_OPTIONS="-Xms1024m" "-Xmx2048m"

neste if podemos mudar o minimo e o máximo de memória, o padrão é o que esta acima.

-steps - table input, table output

conectar os steps

salva como Select Values

-step - usar dois select values

dentro do step select values temos 3 abas

--Aba Select & Alter

-Rename (podemos alterar o nome, tamanho e a presição)

--Aba Remove

tudo que deixar em fieldname irá apagar

--Aba Meta-Data

(pode alterar o nome, tipo entre outras formatações)

**AULA 32 - SPLIT FIELDS E CONCAT FIELDS**

-steps - csv file input, 2 split fields, 3 concat fiels, 2 select values, 1 table output-s

-step csv file input

-name (GastosViagens)

-step split fields (sepação de campos)

-field to split (id\_pais)

-delimiter (\)

-new field (id, pais)

-remove id (n, n)

-type (integer, String)

-step split fields 2 (sepação de campos)

-step name (separa meio de transporte)

-field to split (meio\_de\_transporte)

-delimiter (-)

-new field (uf\_destino, transporte)

-remove id (n, n)

-type (String, String)

podemos previsualizar a execução sem executar a transformação:

botão direito no utimo step -> preview... -> quick launch

-step concat fields (unir campos)

-target field name (Origem)

-separator (-)

quais campos vai unir

name (pais, origem)

type (String, String)

-step concat fields (unir campos)

-nome step (destino)

-target field name (Destino)

-separator (-)

quais campos vai unir

name (destino\_pais, uf\_destino)

type (String, String)

-step select values

-aba remove

-get fields (deixar os campos para apagar -> pais, uf\_origem, pais\_destino, uf\_destino)

-step select values

-aba meta-data

-get fields (formatar tipo campo)

-step table output

-nome step (GastosViagens)

-connection (dados\_publicos)

-target table (GastosViagens)

-SQL (criar e executar SQL)

**AULA 33 - ORDENANDO E AGRUPANDO - SORT ROWS / GROUP BY**

-criar uma nova transformação

-usar step table input, sort rows, group by, memory group by

-step table input

-connection (dados\_publicos)

-get sql (tabela pagamento - deixar somente o valor e orgão pagador)

-step sort rows

-fieldname (qual coluna irá ser ordenado, no caso, orgao pagador)

-step group by

-group field (selecionar o campo que irá agrupar, no caso orgao pagado)

-Name (campo que vai ser agregado, dar um nome a ele, no caso, total)

-subject (valor irá ser o campo de agregação)

-type (tipo de agregação, no caso vai ser uma soma - SUM)

neste caso ele usou o sort para ordenar e o group para agrupar.

ja o memory group by faz as duas coisas, mais ele aloca os dados em memoria e se tiver usando muitos dados pode estourar a memória do computaodr.

-step memoru group by

-get fields

-group field (apagar valor)

-Name (total)

-subject (valor)

-type (Sum)

**AULA 34 - DIRECIONAMENTOS DE SAIDA - VALUE MAPPER / SWITCH CASE.**

-steps - table input, value mapper, switch / case, excel output, 2 text file output

-step excel output

-nome step (Produção)

-filename (local de salvar)

--aba fields - obter os campos

-step txt file output

-nome step (Design)

-extensão (csv)

-filename (local de salvar)

--aba fields - obter os campos

-step txt file output

-nome step (DemaisCargos)

-extensão (txt)

-filename (local de salvar)

--aba fields - obter os campos

-step table input

-connection (conn\_estudo)

-get sql (empregados)

-step value mapper

-step name (Altera sexo)

-fieldname (sexo)

-source value ( M, F)

-target value (masculino, feminino)

-step switch / case

-step name (Direcionamento de saídas)

-field name (departamento)

-use string contains (marcar)

-case value data type (String)

-value (Production, Tool Design)

-target step (Produção, Design)

-default target step (DemaisCargos)

**AULA 35 - FILTRANDO REGISTROS E TRATANDO NULOS**

-steps 1 table input, 3 filter rows, 4 dummy, 1 if field value is null

table input

-connection (dados\_publicos)

-get sql (pagamentos)

filter rows 1

-step name (Maior 300 e Menor 500)

field (campo valor, condição >, value 300)

add condition (and)

field (campo valor, condição <, value 500)

filter rows 2

-step name ( )

-field (campo orgao\_pagador, condição is null)

if field value is null

-step name (tratando nulos)

-select fields (selecionar)

-field (orgao\_pagador)

-replace (Sigilosa)

-set empty string (N)

filter rows 3

-step name (Maior 500)

-field (campo valor, condição >, value 500)

**AULA 36 - Filtros Aprimorados - Java Filter**

steps - 1 excel input, 1 java filter, 2 dummy

step dummy 1

-name (Verdadeiro)

step dummy 2

-name (Falso)

Excel input

-files -> navegar (arquivo)

-sheets -> get sheetname

-field -> mudar os campos devidos

java filter

-condition (nome.startsWith("C") -> começar com...)

-condition (nome.contains("M" -> contém... )

-condition (nome.equals("Eliz Lima" -> contém... )

-condition (idade > 30)

-condition (idade > 30 && sexo.contains("F") )

**AULA 37 - MANIPULANDO STRINGS - REPLACE IN STRING E STRING CUT**

-steps - 1 csv file input, 1 replace in string, 1 string cut, 2 dummy

csv file input

-filename (arquivo nome indeferidos)

-obter campos

replace string

-in stream field (sexo, sexo)

-search (Masculino, Feminino)

-Replace (M, F)

-out stream field (criar novo campo - Homens, Mulheres)

string cut (trás o intervalo do corte)

-in stream field (sexo)

-out stream field (sexo2)

-cut from (0)

-cut to (4)

**AULA 38 - REALIZANDO CALCULOS - CALCULATOR**

steps - 1 excel input, 1 calculator, 1 dummy

excel input

filename (arquivo lucros)

calculator

novo campo (total\_compra)

calculo (A \* B)

campo A (qtd\_vendida)

campo B (valor\_compra)

tipo do valor (number)

novo campo (total\_venda)

calculo (A \* B)

campo A (qtd\_vendida)

campo B (valor\_venda)

tipo do valor (number)

novo campo (Lucro\_Total)

calculo (A - B)

campo A (total\_venda)

campo B (total\_compra)

tipo do valor (number)

**AULA 39 - GENERALIDADES - FORMULA**

steps - 1 table input, 1 formula, 1 dummy

table input

-get sql (cgu)

formula

- new field (MES\_ANO)

- formula ( coluna tem que por entre [])

"0"&[MES]&"/"&[ANO]

-value type (string)

- new field (LEFT)

- formula ( coluna tem que por entre [])

LEFT([nome];4)

-value type (string)

- new field (UPPER)

- formula ( coluna tem que por entre [])

UPPER([orgao])

-value type (string)

**AULA 40 - TRABALHANDO COM VARIÁVEIS E SUB-TRANSFORMAÇÕES**

steps - 1 Data Grid, 1 table input, 1 dummy

data grid

->meta

-Name (data)

-type (date)

-format (dd/MM/yyyy)

->data

por os dados

(28/05/1956)

table input

-nome step (Parametros)

-connection (conn\_estudo)

-get sql(empregados)

adicionar clausula where dentro da consulta

WHERE datanascimento > ?

\*->(o ? serve como variável pegando o dado da data do data grid mas para isso temos que marcar a caixa replace variables in script)

\*->(em insert data from step temos que pegar o nome do step anterior no caso data grid)

copiar steps e colar e adicionar um novo dummy e liga a primeira execução na primeira linha para que o step não execute

além de desativar os hopes (as linhas)

muda o nome da copia de parametro para variavel

dar dois click da tela do pdi que vai aparecer uma caixa

escolher a aba Parâmetros

-Parâmetro (Data\_Teste)

-Valor padrão (28/05/1956)

fecha essa janela e vai no input table variáveis

apagar insert data from step e apagar o ?

no local da ? apertar ctrl + space e escolher a variável data\_teste e por entre aspas simples

obs: por padrão ele fica ${data\_teste}

copiar somente o table input variavel e o dummy

pegar step dummy, transformation Executor

step transformation Executor

-step name

-browse (caminho de uma transformação - executa uma transformação que ja fiz)

criar uma nova transformação como Sub transformação

na sub transformação copiar o data grid da outra transformação para essa nova.

data grid

-data (28/05/1976)

-step set variables

-field name (campo que vou pegar - data)

-variable name (DATA\_NASCIMENTO)

-variable scope type (valid in the virtual machine)

volta na primeira transformação

step transformation Executor

-browse (procura a transformação - sub transformação)

step talbe input

mudar a variável da where para DATA\_NASCIMENTO (se caso não aparecer, salvar, fechar e depois abrir novamente a transformação e verificar que vai ter aparecido)

**AULA 41 - TRANSFORMANDO COLUNAS EM LINHAS - ROW NORMALIZAR**

step que transforma colunas em linhas

-steps data grid, row normalize

step data grid

-meta

name (id, nome, jan, fev, mar, abri)

type (interger, string, number nos outros dados dos meses)

-data

adicionar os dados

id (1, 2, 3, 4)

nome (Maria, João, Pedro, Marcos)

jan (100, 200, 300, 400)

fev (150, 200, 250, 350)

mar (125, 150, 250, 300)

abri (200, 250, 300, 350)

-step row normaliser

step name ()

type field (nome da minha coluna - Mes)

Fieldname (nome das colunas dos meses - por de jan a abr)

type (nao é o tipo de dado, é o mesmo nome do field name - repetir o nome)

new field (novo campo - Valores em todos)

**AULA 42 - TRANSFORMANDO LINHAS EM COLUNAS - ROW DENORMALIZAR**

steps data grid, sort rows, group by, denomaliser, if field value is null, dummy

ligar todos os steps nessa mesma ordem

step data grid

-meta

name (produto, modelo)

type (string, string)

set empty string (n, n)

-data

produto (1 produto1, 4 produto2, 3 produto3)

modelo (modeloA, ModeloB, modeloC, modeloA, ModeloB, modeloC, ModeloB, modeloC)

step sort row

-fieldname ( produto, modelo)

-Ascending (s, s)

step group by

group field (aqui neste exemplo vou agrupar pela duas colunas - produto, modelo)

aggregates (criar novo campo)

name (QTD)

subject (modelo)

type (contagem - number of values(N))

step denomaliser

the key field (modelo)

group field (produto)

-target field

targer fieldname (Qtd modeloA, Qtd modeloB, Qtd modeloC)

Value fieldname (QTD, QTD, QTD)

key value (modeloA, modeloB, modeloC)

type (interger, interger, interger)

aggregation (sum, sum, sum)

step if field value is null

select field (selecionar)

-fields

field (Qtd modeloA, Qtd modeloB, Qtd modeloC)

replace by value (0, 0, 0)

**AULA 43 - UTILIZANDO O MERGE JOIN**

STEPS - 2 excel input, merge join, 1 dummy

excel input

spread sheet (apache poi)

navegar (Vendedores)

Sheets

add

fields

obter campos e limpar dos dois lados (trim type)

format (#, tira casa decimal)

excel input 2

spread sheet (apache poi)

navegar (Vendas)

Sheets

add

fields

obter campos e limpar dos dois lados (trim type)

format (#, tira casa decimal)

merge join

step name (merge Join)

first step (excel input)

second steps (excel input 2)

join type (inner)

**AULA 44 - UTILIZANDO O MULTWAY MERGE JOIN.**

STEPS - 3 excel input, multiway merge join, dummy

botão direito na transformação, por os nomes nas nota

Vendas, Vendedores, Vendas\_clientes

configurar os 3 inputs com os respectivos arquivos

multiway merge join (posso fazer um join com 3 tabelas)

input 1 (vendas)

input 2 (vendedores)

input 3 (vendas\_clientes)

Select Keys (a chave que as tabelas se interligão, no caso, id\_vendedor)

**AULA 45 - UNION ALL COM DUMMY.**

STEPS - excel input, get data from xml, text file input, table output, dummy

excel input (Filial\_01)

text file input (Filial\_02)

get data from xml (Filial\_03)

-file (procurar o arquivo e adicionar)

-content (loop XPath, selecionar Rows/Row para selecionar todas as colunas abaixo dele)

se o dummy tiver os arquivos todas as mesmas colunas e mesmo tipo de dados ele faz a união dessas tabelas.

table output

nome step (Filial Matriz)

connection (dados\_publicos)

target table (Filial\_Matriz)

SQL (mudar os tipos dos números para int)

**AULA 46 - UTILIZANDO APPENDS STREAMS E UNIQUE ROWS**

STEPS - 2 data grid, 1 append streams, 1 dummy, 1 unique rows

data grid

-meta

name (cargo)

type (string)

-data

vendedor

analista

assistente

data grid

-meta

name (cargo)

type (string)

-data

Coordenador

Gerente

Vendedor

ligar os dois steps ao append

Append streams (semelhante ao dummy com relação ao union, aqui a diferença é que podemos ordenador a tabela que fica em cima ou em baixo)

step name

Head hop (data grid)

tail hop (data grid 2)

Unique rows

field name (cargo, nome do campo que quero comparar os campos unicos)

ignore case (Y)

**AULA 47 - UTILIZANDO O MERGE ROWS (DIFF)**

STEPS - excel input, table output, table input, 2 sort rows, merge rows (diff), switch case, 2 select values,2 dummy

excel input

navegar para matriculados e fazer a importação do arquivo

table output

nome do step (Matriculados)

connections (dados\_publicos)

target (matriculados)

sql (trocar pra int)

table input

name (table matriculados)

connection (dados\_publicos)

get sql (matriculados)

2 sort

fielname (cod)

ascedenting (S)

Merge Rows (diff)

references rows (sort rows)

compare rows origin (sort rows 2)

flag fieldname (flagfield)

key field (cod)

value field (cod)

swich case

field (flagfield)

case value (String

value ( new, identical)

target (select values, selct values 2)

select values

get fields e retirar o flag

select values 2

get fields

**AULA 48 - ENTENDO O LOOKUP**

busca em uma base de dados em uma outra base de dados ou entre arquivos

-lookup funciona de N X 1 (muitos para Um) e 1 X 1 (Um para Um)

**AULA 49 - UTILIZANDO O STREAM LOOKUP**

STEPS - 2 EXCEL INPUT, CSV INPUT, 2 STREAM LOOKUP, 1 DUMMY

CSV INPUT

filename (buscar arquivo "Alunos")

delimiter (;)

obtem campos

preview

EXCEL INPUT 1

-files

spread sheet ( Excel 2007 XLSX (Apache POI))

directory ( arquivo "endereço")

add

-!sheets

get sheetnames (obter a pasta do arquivo)

-fields

get fields

trim type (os dois)

STREMA LOOKUP 1

step name (busca endereço)

lookup step (buscar excel input)

-the key(s) to look

field (id - campo de verificação)

lookupfield (id - campo de verificação)

-specify the fields

field (Estado)

type (string)

EXCEL INPUT 2

-files

spread sheet ( Excel 2007 XLSX (Apache POI))

directory ( arquivo "nascimento")

add

-!sheets

get sheetnames (obter a pasta do arquivo)

-fields

get fields

trim type (os dois)

obs: na idade por em format o #

STREMA LOOKUP 2

step name (busca idade)

lookup step (buscar excel input 2)

-the key(s) to look

field (id - campo de verificação)

lookupfield (id - campo de verificação)

-specify the fields

field (idade)

type (integer)

**AULA 50 - UTILIZANDO DATABASE LOOKUP**

STEPS - 2 TABLE OUTPUT, 2 EXCEL INPUT, 2 DUMMY, 1 CSV INPUT, 2 DATABASE LOOKUP,

procurar sql server configuration manager

ou C:\Windows\SysWOW64\SQLServerManager14.msc

-serviços do SQL Server

verificar se o serviço esta em execução (iniciar serviços)

-configuração do SQL native client

--protocolos de cliente

verificar se o TCP/IP esta habilitado (habilitar se não estiver)

-configuração de rede do SQL Servers

--protocolo para MSSQLSERVER

verificar se o TCP/IP habilitado

endereço ip - configuração da porta 1433

microsoft sql tools

management studio

senha 123456

autenticação do sql server

clicar em cima de banco de dados

nova consulta

CREATE DATABASE VARIADOS; (seleciona e executa)

USE VARIADOS;

EXCEL INPUT 1

-files

spread sheet ( Excel 2007 XLSX (Apache POI))

directory ( arquivo "endereço")

add

-!sheets

get sheetnames (obter a pasta do arquivo)

-fields

get fields

trim type (os dois)

EXCEL INPUT 2

-files

spread sheet ( Excel 2007 XLSX (Apache POI))

directory ( arquivo "nascimento")

add

-!sheets

get sheetnames (obter a pasta do arquivo)

-fields

get fields

trim type (os dois)

obs: na idade por em format o #

TABLE OUTPUT 1

nome step (Endereço)

connection (new...

->connections type: MS SQL Server (Native)

->Access:Native (JDBC)

->connections name: conn\_variados\_sqlserver

->Host Name: nome que esta no sqlserver [no meu caso LUCAS-NOTE]

->Database Name: VARIADOS

->Instance Name: pode deixar em branco

->Port Number: 1433

->Username: sa

->Password: senha cadastrada (no meu caso 123456)

target table (ENDERECO)

SQL (tira ; do final da consulta)

TABLE OUTPUT 2

nome step (NASCIMENTO)

target table (NASCIMENTO)

SQL (tirar ; para não da erro ou se caso precisar(der erro sem o ;) deixar o ;)

CSV INPUT

step name (alunos)

navegar (procurar alunos)

delimiter (;)

DATABASE LOOKUP

nome step (Busca Endereço)

connection (conn\_variados\_sqlserver)

table lookup (ENDERECO)

-chave para examinar

campo da tabela (id)

comparar (=)

campo 1 (id)

-valores a sairem

campo (cidade)

tipo (string)

DATABASE LOOKUP 2

nome step (Busca Nascimento)

connection (conn\_variados\_sqlserver)

table lookup (NASCIMENTO)

-chave para examinar

campo da tabela (id)

comparar (=)

campo 1 (id)

-valores a sairem

campo (idade)

tipo (integer)

**AULA 51 - UTILIZANDO O REST CLIENT - CONSUMINDO API JSON**

STEPS - GENERATE ROWS, REST CLIENT, JSON INPUT, SELECT VALUES, EXCEL OUTPUT

GENERATE ROWS

nome (URL)

limit (1 - gerar apenas uma linha com essa url)

tipo (String)

valor (http://compras.dados.gov.br/fornecedores/v1/fornecedores.json?uf=DF)

REST CLIENT

-general

accept url from field (marcar)

url field name (url)

http method (GET)

application type (JSON)

result field name (resultado)

JSON INPUT

-file

source is from a previous step (marcar)

select field (resultado)

-fields (ter que digitar todos os campos da nossa url

Name

id

cnpj

nome

ativo

id\_municipio

uf

id\_porte\_empresa

habilitado\_licitado

Path

$..id

$..cnpj

$..nome

$..ativo

$..id\_municipio

$..uf

$..id\_porte\_empresa

$..habilitado\_licitado

tipo

Integer

string

string

boolean

integer

string

integer

boolean

SELECT VALUES (CAPTURAR AS COLUNAS)

apagar url, resultado

EXCEL OUTPUT

navegar (salvar em outputs, arquivos excel, salva como fornecedores)

**AULA 52 - UTILIZANDO O REST CLIENT - CONSUMINDO** **API XML**

STEPS - GENERATE ROWS, REST CLIENT, GET DATA FROM XML, SELECT VALUES

GERENATE ROWS

Fields

limit (1)

nome (url)

tipo (string)

valor (https://sistemas.tce.pe.gov.br/DadosAbertos/Municipios!xml)

procurar o api para colar em valor (tce ce api)

REST CLIENT

URL field (url)

HTTP method (GET)

application type (XML - versões nova do programa ja identifica o typo)

GET DATA FROM XML

->file

XML Source is defined (marcar)

get XML source (result)

->content

loop xpath (capturar caminho - /resposta/conteudo/Municipios)

->Fields

tem que por os nomes dos campos no xpath (os mesmo que tem no XML)

no name pode dar outros nomes

SELECT VALUES

->remove

get fields (deixar o que quero remover)

**AULA 53 - UTILIZANDO WEB SERVICE LOOKUP**

STEPS - WEB SERVICES LOOKUP, GET DATA FROM XML, SELECT VALUES

WEB SERVICES LOOKUP

https://www2.camara.leg.br/transparencia/dados-abertos/dados-abertos-legislativo/webservices/deputados

-usar a url acima para conseguir o endpoint (url abaixo)

aba web service

url (https://www.camara.gov.br/SitCamaraWS/Deputados.asmx?wsdl)

clicar em load para poder disponibilizar os metodos para operation

operation (obterDeputados)

quando clicar na operation, vai aparecer uma nova aba para poder obter campos

aba obterDeputadosResult

obtem campo

GET DATA FROM XML

->file

XML Source is defined (marcar)

get XML source (ObterDeputadosresult)

->content

loop xpath (capturar caminho - /ObterDeputadosResult/deputados/deputado)

->Fields

tem que por os nomes dos campos no xpath utilizando (os mesmo que tem no XML)

no name pode dar outros nomes

SELECT VALUES

remove

deixa o campo que pretendo remover

TABLE OUTPUT

nome (output)

connection (conn\_variados\_sqlserver)

target table (Web\_Service)

SQL (executar o script)

INSERT / UPDATE

connection (conn\_variados\_sqlserver)

target table (Web\_Service)

the key

get fields

update fields

get update fields

campo update deixar como "N" no ideCadastro

**AULA 54 - UTILIZANDI O REST CLIENT - CONSUMINDO API COM TOKEN (AUTHORIZATION)**

procurar api atlas noticia (primeiro link)

recursos e api

utilizando api

criar cadastrado

acesso via postman

STEPS - GENERATE ROWS, REST CLIENT

GENERATE ROWS

limit (1)

fields:

nome (urlToken)

tipo (string)

valor (https://api.atlas.jor.br/api/v1/auth/login)

nome (body)

tipo (string)

valor ({"email": "lucasbragadasilva@gmail.com", "password": "sua senha"})

REST CLIENT

accept url from field (marcar)

url field name (urlTokien)

http method (post)

body field (body)

application type (json)

SELECT VALUES

remove todos menos result

FORMULA

new field (Authorization)

formula ("Bearer"&""&[result])

value type (string)

JSON INPUT

-File

souce is from a previous step (marcar)

select field (result)

-Fields

name (access\_token)

Path ($..access\_token)

tipo (string)

alterar a formula e por access\_token

REST CLIENT 2

analytic?estado=RO

url (

http method (GET)

result field (dados)

-headers

get fields

deixar Authorization

SELECT VALUES 2

remover dados da aba remover

JSON INPUT 2

-File

souce is from a previous step (marcar)

select field (dados)

-Fields

campos o retorno do api

name (nome campo)

Path ($..campo)

tipo (tipo dado)

**AULA 55 - COMBINATION LOOKUP/UPDATE - SCD TIPO\_02**

STEPS - EXCEL INPUT, COMBINATION LOOKUP/UPDATE

EXCEL INPUT

adicionar o arquivo fornecedor e adicionar os campos.

COMBINATION LOOKUP/UPDATE

nome step (Fornecedores)

connection (conn\_variados\_sqlserver)

tabela de destino ( Fornecedor)

obtem campos

campo chave técnica (sk\_fornecedor)

usa tabela máxima +1 (marcar)

SQL

**AULA 56 - DIMENSION LOOKUP/UPDATE -SCD TIPO\_02**

STEPS - EXCEL INPUT, DIMENSION LOOKUP/UPDATE

EXCEL INPUT

adicionar arquivo fornecedor 2

DIMENSION LOOKUP/UPDATE

step name (Fornecedor02)

connection (conn\_variados\_sqlserver)

target table (Fornecedor02)

commit size (100, a cada 100 registro insere no banco)

get fields

-keys

deixar somente a chave (idfornecedor)

-fields

get fields

dimension field (NOME)

stream field (NOME)

type of dimension (insert)

technical key (sk\_fornecedor)

use table maximum +1 (marcar)

version field (version)

min.year (2010)

max year (2023)

**AULA 57-58 STEP MODIFIELD JAVASCRIPT VALUE PART-01\_02**

STEPS - EXCEL INPUT, MODIFIED JAVA SCRIPT VALUE, DUMMY

EXCEL INPUT

adicionar arquivo javascript

MODIFIED JAVA SCRIPT VALUE

// script here

var nome01 = upper(nome);

var nome02 = lower(nome01);

var nome03 = initCap(nome02);

var nome04 = replace(nome, "a", "x");

var nome05 = nome.replace(/M/g, "V"); (/M vai procurar todos os M, /g server para procurar linha a linha, "V" e substituir pelo V)

if(sexo == "M"){

sexo = "Masculino";

}else{

sexo = "Feminino";

}

var bonus;

if(realizado >= meta){

bonus = realizado \* 0.20;

}else{

bonus = 0

}

var novoCargo;

if(idade <= 30){

novoCargo = "Vendedor Junior";

}else if(idade <= 40){

novoCargo = "Vendedor Pleno";

}else{

novoCargo = "Vendedor Senior";

}

var data = new Date();

var hoje = data.getDate() + "/" + data.getMonth() + "/" + data.getFullYear();

fieldname (nome variavel, nome01)

type (string)

replace value (N)

fieldname (nome variavel, nome02)

type (string)

replace value (N)

fieldname (nome variavel, sexo)

type (string)

replace value (S) (para poder substituir o sexo utilizando a clausula if)

fieldname (bonus)

type (number)

replace value (N)

fieldname (novoCargo)

type (String)

replace value (N)

fieldname (hoje)

type (String)

replace value (N)

**AULA 59 - INTRODUÇÃO AOS JOBS**

o que são jobs?

recurso de execução

podem executar tanto a transformações quanto outros jobs

STEPS - GENERATE ROWS, DUMMY,

GENERATE ROWS

nome (Gerar linhas)

limit (10000)

-fields

nome (Numeros)

tipo (Interger)

valor (1)

Criar um novo Job

file -> Novo -> Job

STEPS - START, TRANFORMATION, SUCESS

START

onde vai iniciar

ele tem um recurso de agendamento de execução

geralmente não se usa muito esse recurso por conta que tem que deixar oPDI aberto

vamos criar um repositorio e criar um arquivo .bat para automatizar a execução sem a necessidade de o PDI ficar aberto 24h por dia

(não mexer em nenhuma opção do start)

nos hops (linhas)

verde, cadeado, x

verde = ele só vai para o sucess se não ocorrer nenhum erro

x(vermelho) = se não ocorrer nenhum erro ele não da sucess

cadeado (amarelo) = independete de dar erro ou não ele vai para o sucess

TRANSFORMATION

entry name (Generate Rows)

browse... (localizar o arquivo no caso, gerenate rows)

**AULA 60-61 - FOREACH LOOP COM JOB E VARIÁVEIS NO CONSUMO DE API - PART01\_02**

https://servicodados.ibge.gov.br/api/docs/localidades#api-UFs-estadosUFGet

https://servicodados.ibge.gov.br/api/v1/localidades/estados/{UF}

CRIAR 2 JOBS

CRIAR 2 TRANFORMAÇÃO

TRANSFORMAÇÃO 1 - SETA UF

STEPS - DATA GRID, COPY ROWS TO RESULT

DATA GRID

-meta

name (ESTADOS)

type (String)

-data

SP

RJ

RO

CE

SE

COPY ROWS TO RESULT

TRANSFORMAÇÃO 2 - API Loop

STEPS - GET ROWS FROM RESULT, SET VARIABLES, GET VARIABLES, 2 WRITE TO LOG, REST CLIENT, JSON INPUT, TABLE OUTPUT, SELECT VALUES

GET ROWS FROM RESULT

field name (ESTADOS)

type (String)

SET VARIABLES

field name (ESTADOS)

variable name (VAR\_UFS)

variable scope type (valid in the root job)

GET VARIABLES

name (UFS)

variable(

ctrl + space (atalho para aparecer os dados)

se não

${VAR\_UFS}

)

tipo (string)

WRITE TO LOG

field (UFS)

REST CLIENT

URL(https://servicodados.ibge.gov.br/api/v1/localidades/estados/{UF})

setar a variavel no lugar da {UF}

URL(https://servicodados.ibge.gov.br/api/v1/localidades/estados/${VAR\_UFS})

http method (GET)

application type (JSON)

result field name (Dados)

WRITE TO LOG 2 - teste, depois apagar

field (Dados)

JSON INPUT

-file

source is from a previous step (marcar)

select fields (Dados)

-fields

name (nomes)

path ($.. nomes)

SELECT VALUES

removar dados, estados, ufs

TABLE OUTPUT

nome step ()

connection (conn\_variados\_sqlserver)

target table (API\_LOOP)

SQL

JOB 1 - SALVER COMO JOB LOOP

STEPS - START, TRANSFORMATION, JOB, SUCESS

TRANSFORMATION

entry name (SETA UF)

browse (SETA UF)

JOB

execute every input rows (marcar) (para cada linha que entrar do copy rows to result ele vai executar linha po linha)

JOB 2 - Job Exec API Loop

STEPS START, TRANSFORMATION, SUCESS

TRANSFORMATION

execute every input rows (marcar)

browse (API Loop)