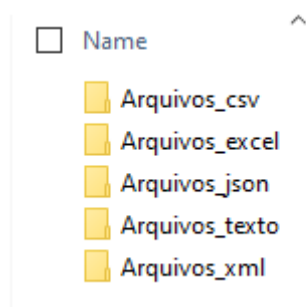


Exercício 00 – Preparação do Ambiente

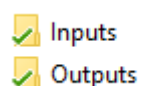
Baixe os arquivos de Input a partir do repositório GitHub do curso:

<https://github.com/hgsilva/posbi-etl>

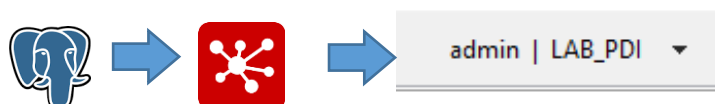
Navegue pela pasta para se ambientar com o seu conteúdo:



Estes arquivos e pastas serão utilizados em diversos exercícios, para armazenar dados de entrada como também os de saída das Transformações.

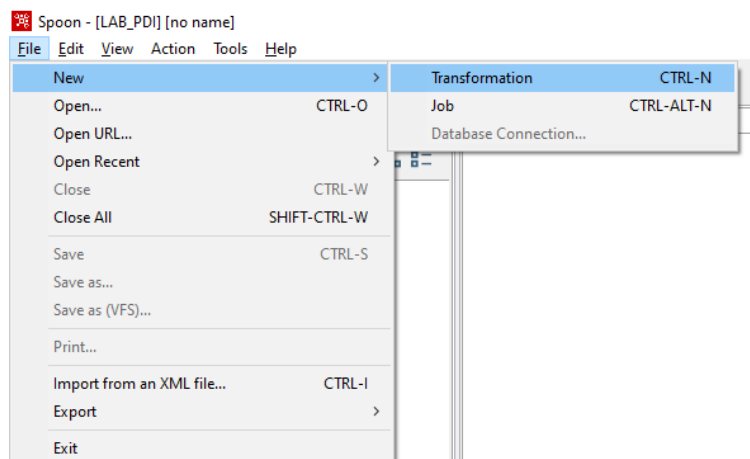


Inicie o serviço do PostgreSQL, o serviço Pentaho Data Integrator e em seguida, conecte-se ao repositório do curso:

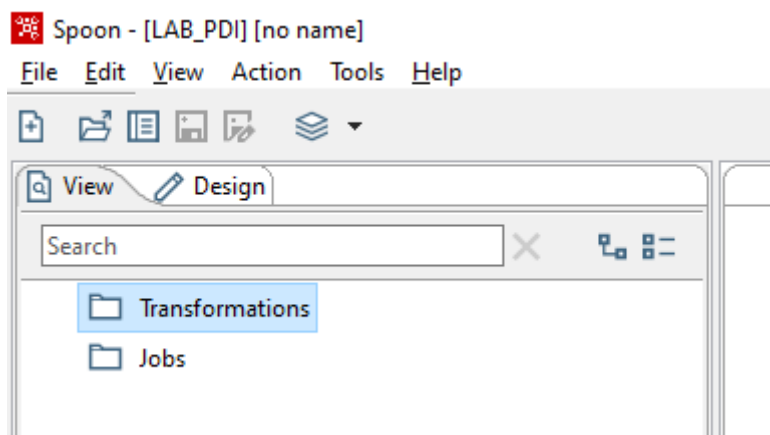


Exercício 01 – Input Data Grid

1.1 Crie uma transformação nova:

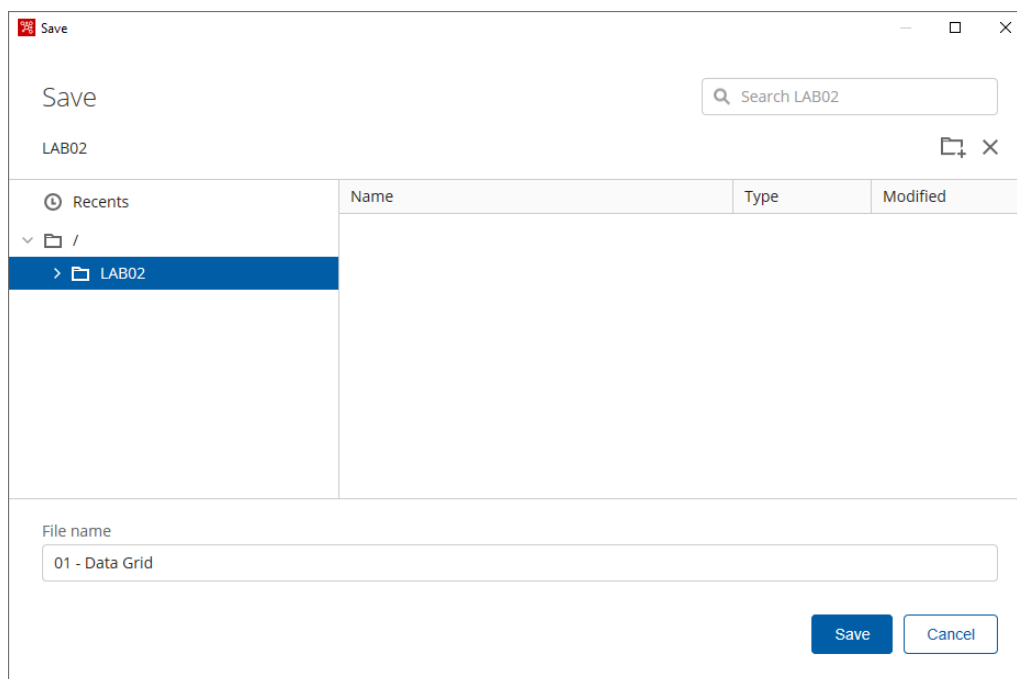


ou

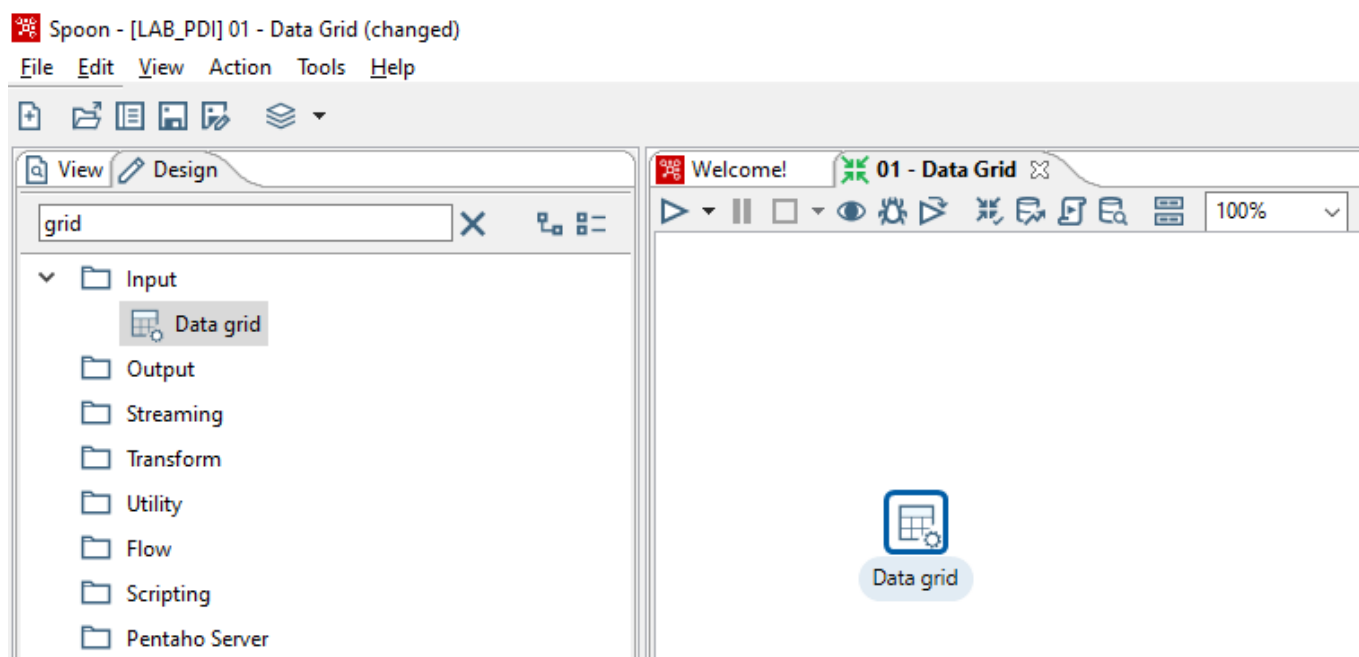


Ou CTRL + N

1.2 Salve a transformação em seu repositório, dentro da pasta LAB02:



1.3 Na aba “Desgin”, localize o step “Data Grid” e o arraste para o ambiente gráfico da transformação “01 – Data Grid”:



1.4 Com um clique duplo no step, configure suas propriedades:

Data grid

Step name: Data grid

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Set empty string?
1	ID	Integer							
2	DESC	String							

Help OK Preview Cancel

Data grid

Step name: Data grid

#	ID	DESC
1	-1	Não Informado
2	-2	Não Localizado
3	-3	Não Aplicável

Help OK Preview Cancel

Enter the preview size

Enter the number of rows to preview:

1000

OK Cancel

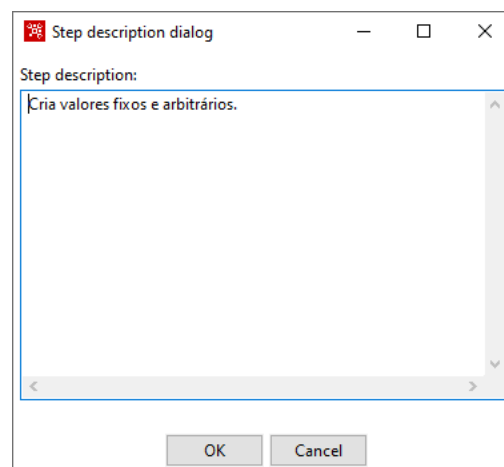
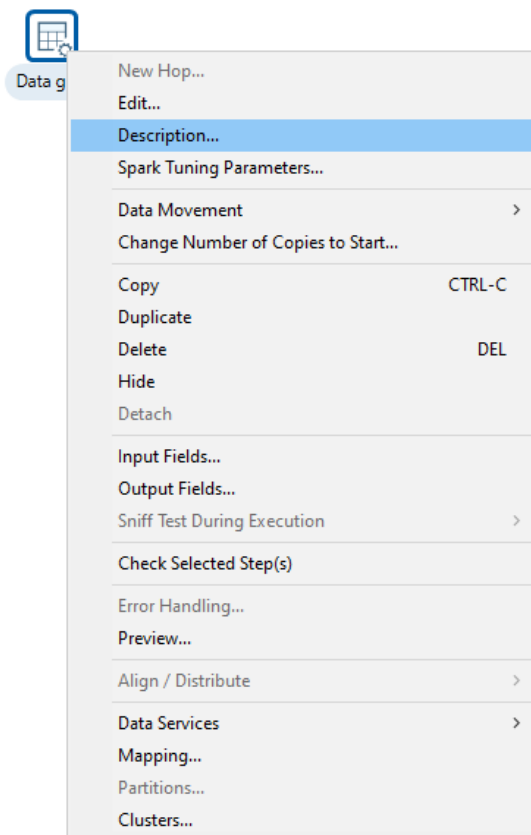
Examine preview data

Rows of step: Data grid (3 rows)

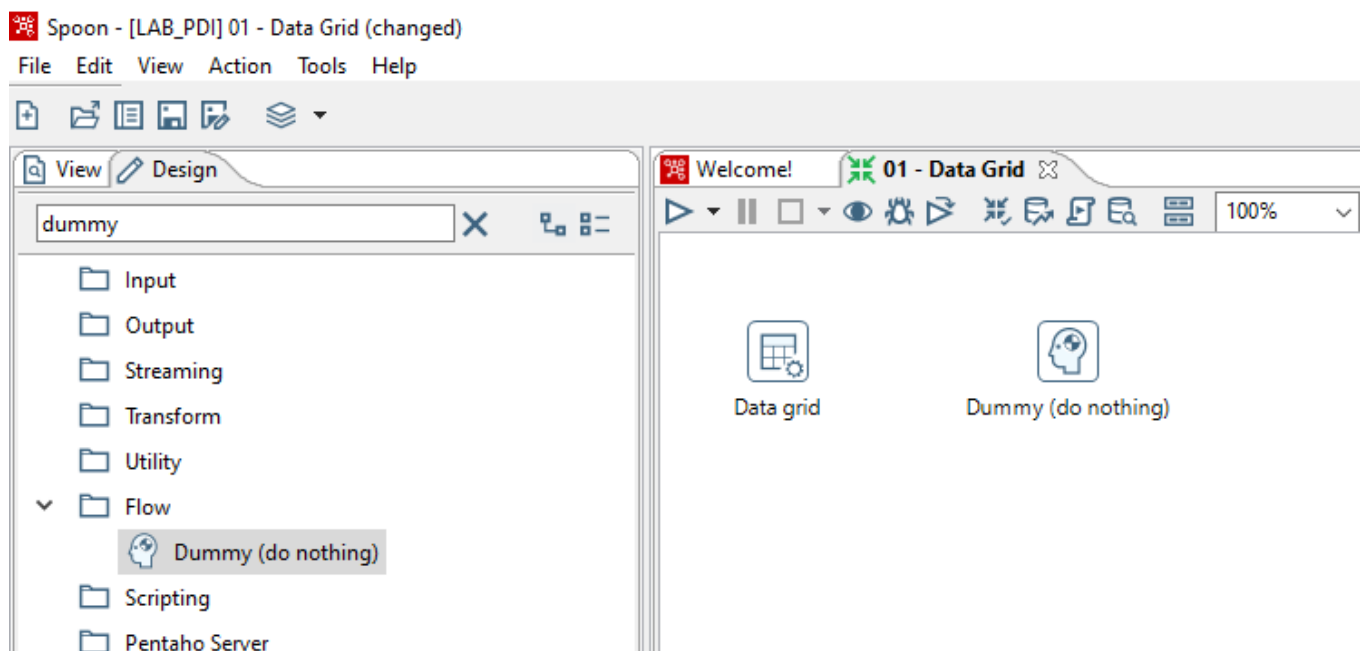
#	ID	DESC
1	-1	Não Informado
2	-2	Não Localizado
3	-3	Não Aplicável

Close Show Log

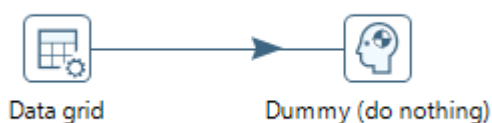
1.5 Com o botão direito, abra o menu de contexto do step e adicione um comentário à descrição deste step:



- 1.6** Adicione o step de fluxo “Dummy” (ainda não serão tratados os outputs, mas este passo irá representar de forma inócua, um step de output):



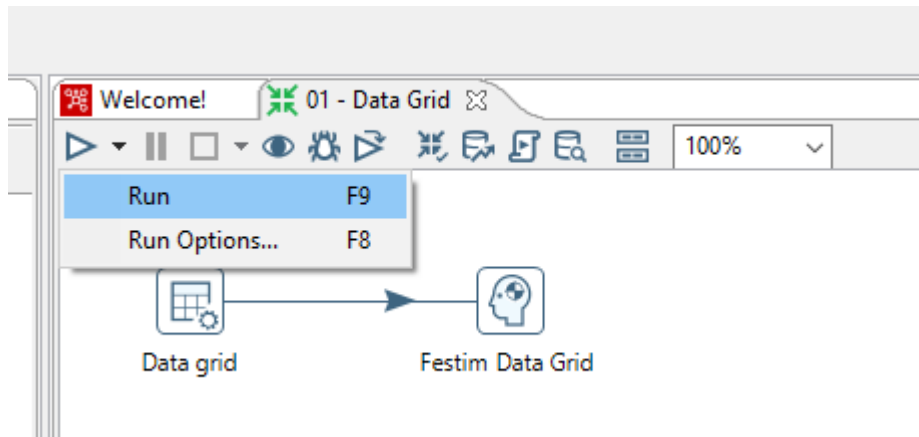
- 1.7** Adicione um “HOP” (conexão) entre os dois steps (pressione SHIFT, clique no step de entrada, arraste e solte até o step de saída, libere SHIFT):



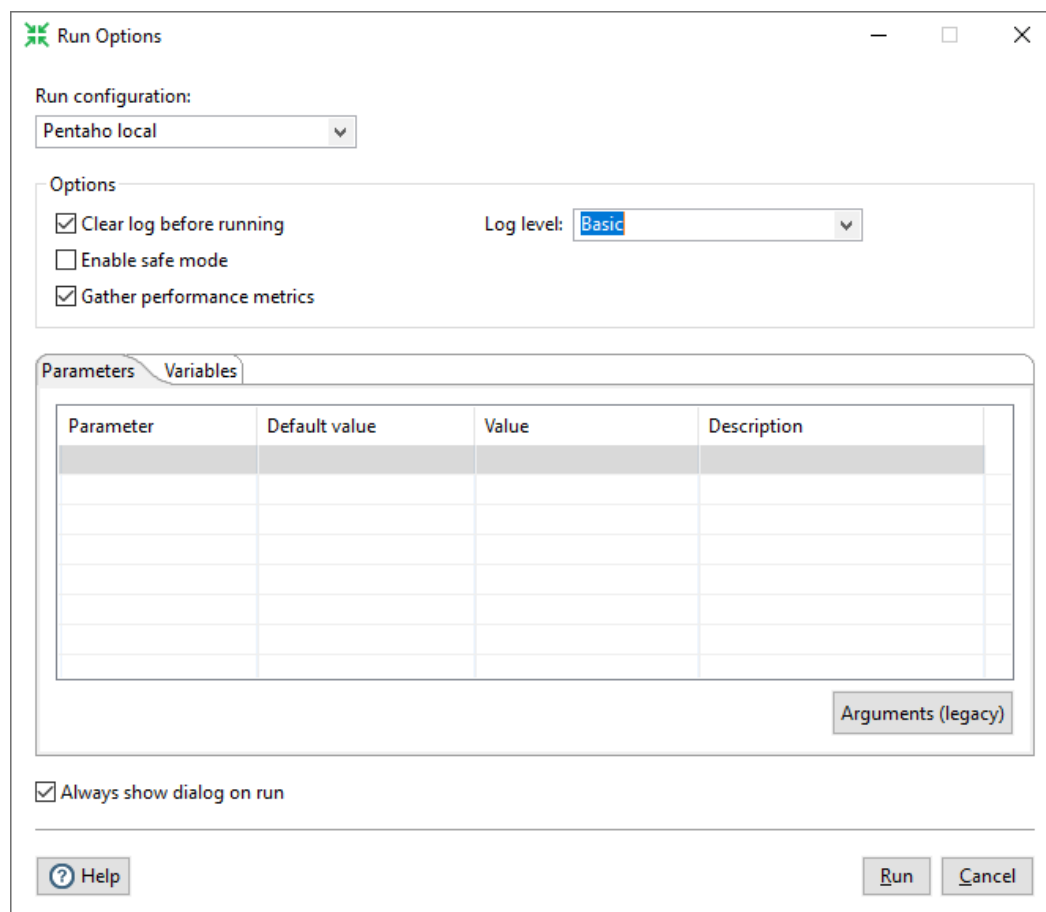
- 1.8** Com um clique duplo, modifique o nome do step “Dummy” e salve a transformação:



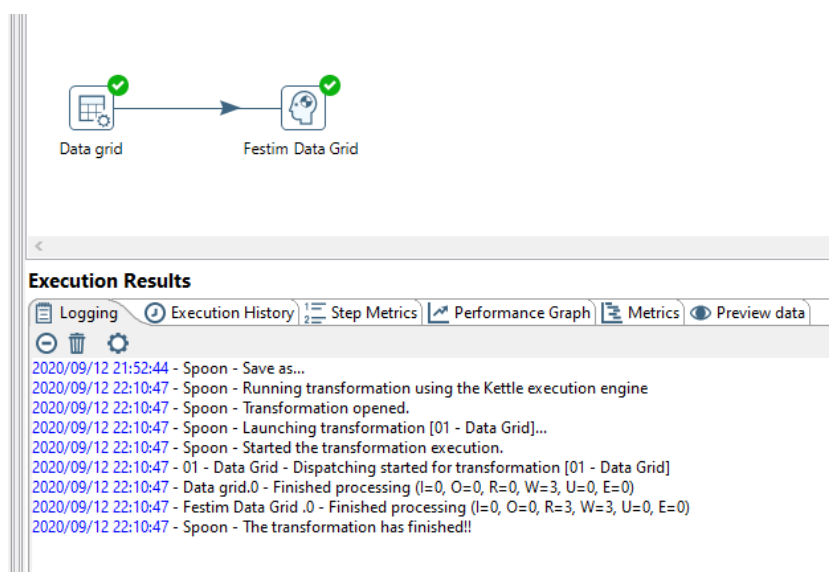
1.9 Execute a transformação:



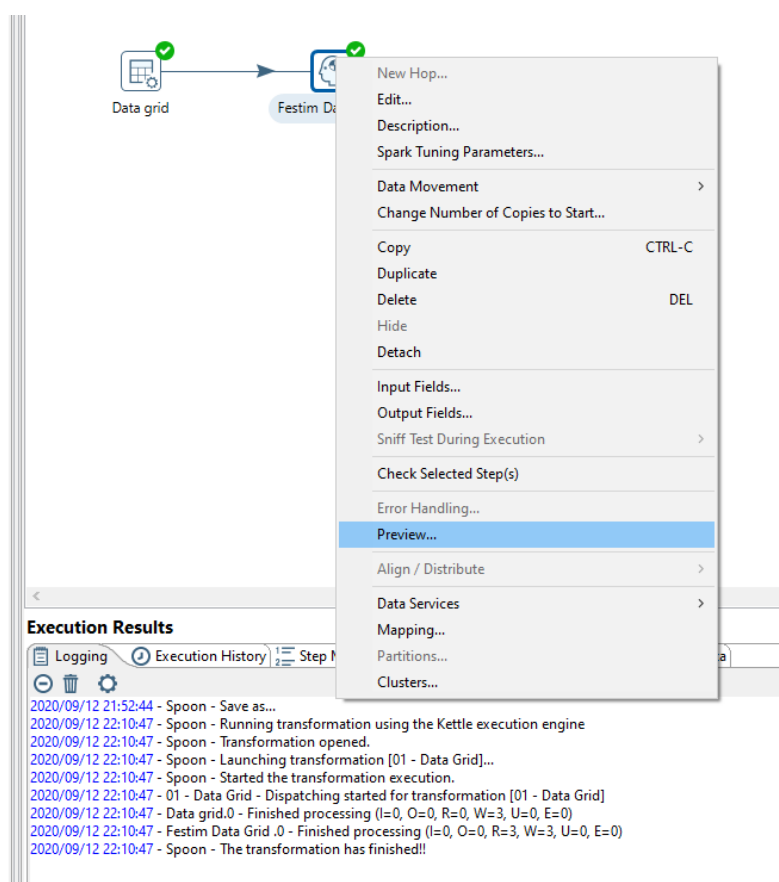
1.10 Verifique o nível de detalhamento do LOG, e garanta que o parâmetro “Log level” esteja marcado como “Basic”:



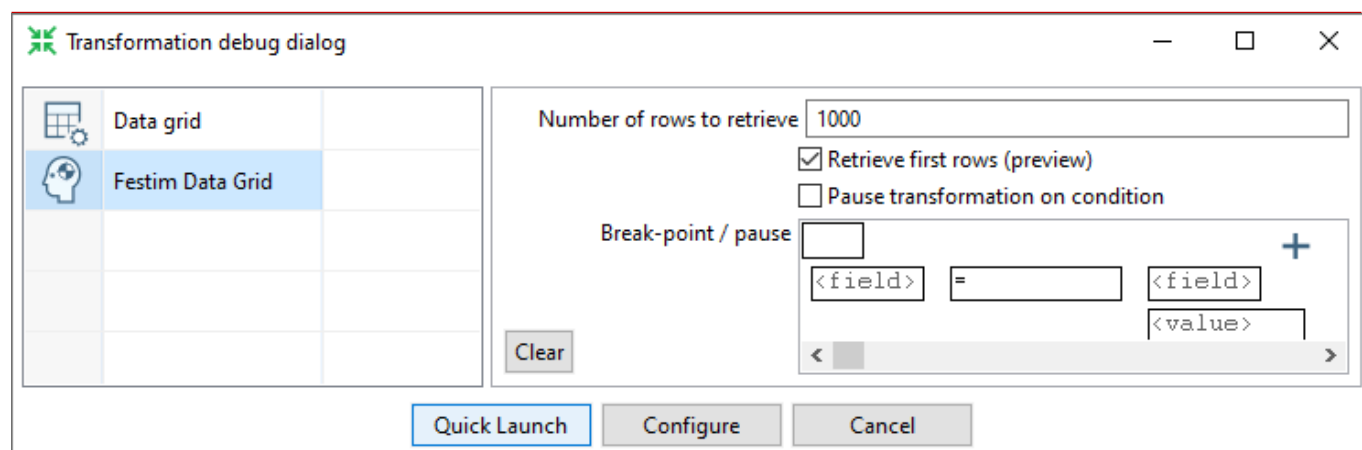
1.11 Confirme se a execução foi realizada com sucesso:



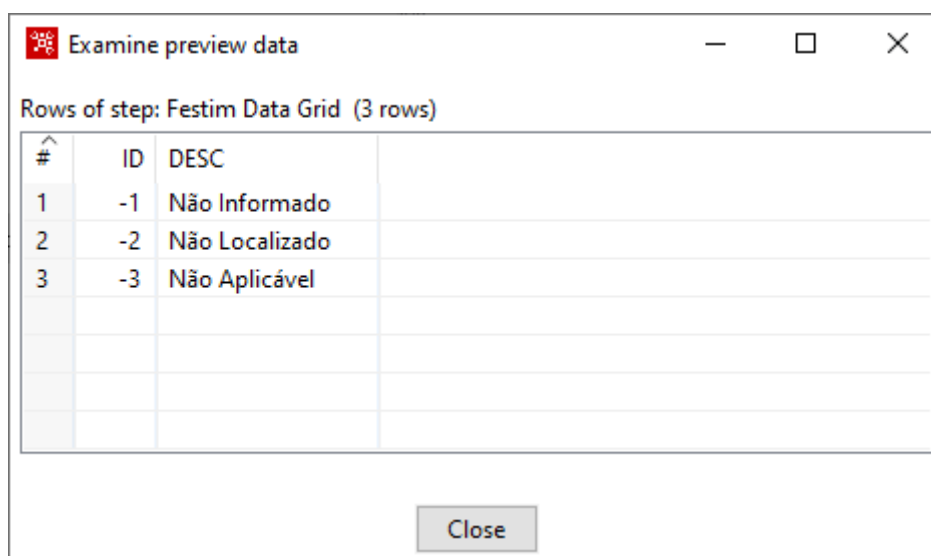
1.12 Valide o transporte dos dados para o step “Dummy” com uma visualização prévia dos dados. Em seguida, salve e feche a transformação:



Com o botão direito, selecione “Preview”



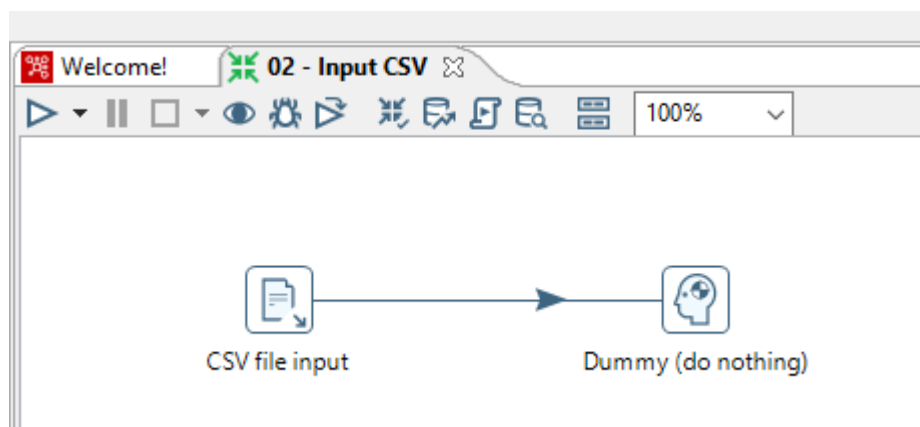
Selecione o step "Dummy" e clique em "Quick Launch"



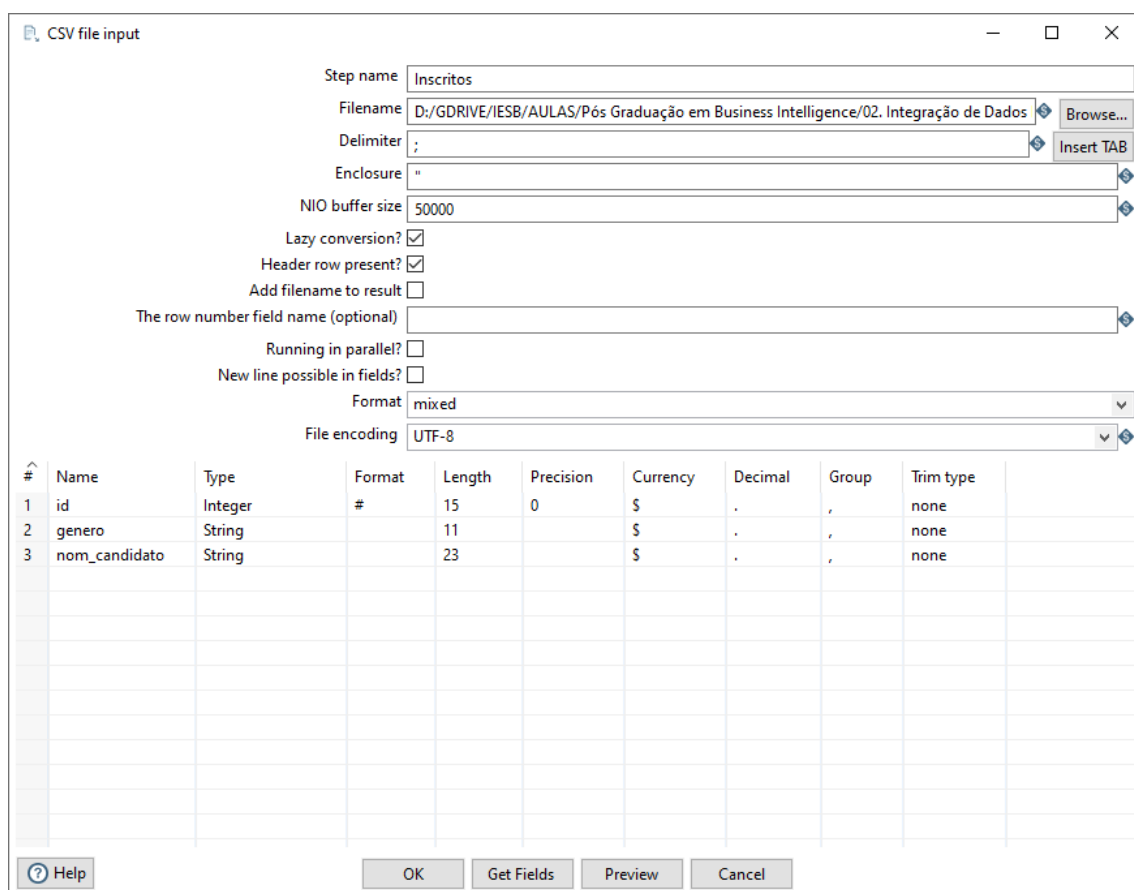
Valide o conteúdo e feche a janela

Exercício 02 – Input CSV

2.1 Crie uma transformação nova chamada “02 – Input CSV” e a salve no repositório, dentro da pasta LAB02. Em seguida, insira os steps “CSV file input” e “Dummy”, criando um HOP entre estes dois steps:



2.2 Configure o step “CSV file input” para receber os dados do arquivo CSV de entrada, e clique em “Get Fields” para obter a estrutura do arquivo:



#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	id	Integer	#	15	0	\$.	,	none
2	genero	String		11		\$.	,	none
3	nom_candidato	String		23		\$.	,	none

2.3 Clique em “Preview” para obter uma prévia dos dados, e confirme que o arquivo está sendo lido de forma correta:

Examine preview data

Rows of step: Inscritos (1000 rows)

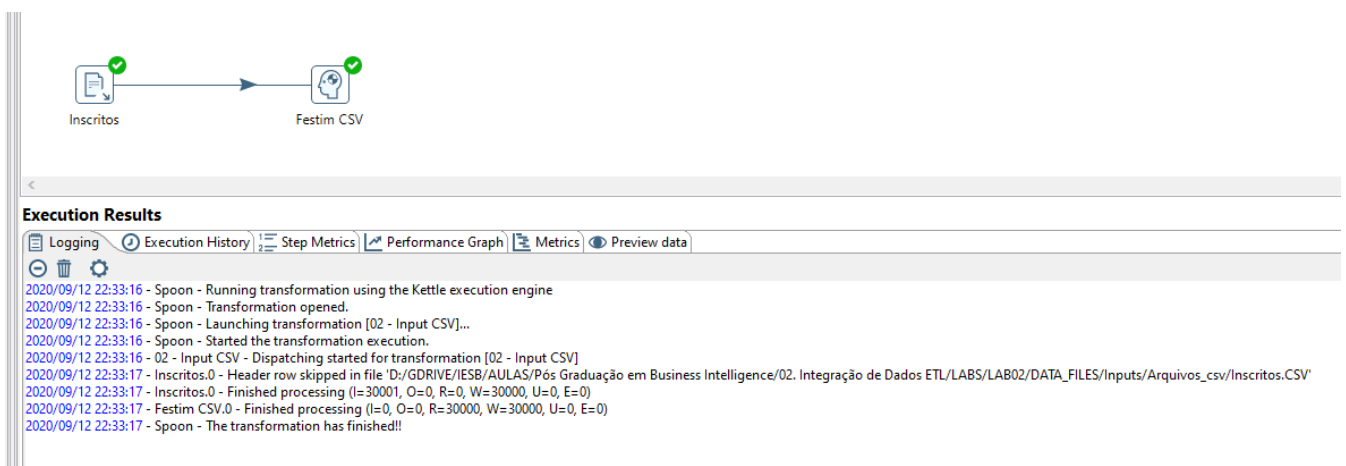
#	id	genero	nom_candidato
1	1	masculino	Martins
2	2	masculino	Carvalho
3	3	feminino	Rocha
4	4	masculino	Cunha
5	5	feminino	Costa
6	6	masculino	Cardoso
7	7	masculino	Almeida

Close Show Log

2.4 Altere o nome do step “Dummy”:



2.5 Salve e execute a transformação:



Execution Results

Logging Execution History Step Metrics Performance Graph Metrics Preview data

2020/09/12 22:33:16 - Spoon - Running transformation using the Kettle execution engine
 2020/09/12 22:33:16 - Spoon - Transformation opened.
 2020/09/12 22:33:16 - Spoon - Launching transformation [02 - Input CSV]...
 2020/09/12 22:33:16 - Spoon - Started the transformation execution.
 2020/09/12 22:33:16 - 02 - Input CSV - Dispatching started for transformation [02 - Input CSV]
 2020/09/12 22:33:17 - Inscritos.0 - Header row skipped in file 'D:/GDRIVE/IESB/AULAS/Pós Graduação em Business Intelligence/02. Integração de Dados ETL/LAB02/DATA_FILES/Inputs/Arquivos_csv/Inscritos.CSV'
 2020/09/12 22:33:17 - Inscritos.0 - Finished processing (I=30001, O=0, R=0, W=30000, U=0, E=0)
 2020/09/12 22:33:17 - Festim CSV.0 - Finished processing (I=0, O=0, R=30000, W=30000, U=0, E=0)
 2020/09/12 22:33:17 - Spoon - The transformation has finished!!

2.6 Nos dados enviado ao step “Dummy” (renomeado para ‘Festim CSV’), repare que nas linhas do campo “nom_candidato” há espaços indesejados ao final do texto:

Examine preview data

Rows of step: Festim CSV (1000 rows)

#	id	genero	nom_candidato
1	1	masculino	Martins
2	2	masculino	Carvalho
3	3	feminino	Rocha
4	4	masculino	Cunha
5	5	feminino	Costa
6	6	masculino	Cardoso
7	7	masculino	Almeida

Close Stop Get more rows

2.7 Para corrigir isto, edite o step de input, nas propriedades da estrutura do arquivo, altere a “Trim type” para “both” nos campos “genero” e “nom_candidato”:

CSV file input

Step name: Inscritos

Filename: D:/GDRIVE/IESB/AULAS/Pós Graduação em Business Intelligence/02. Integração de Dados Browse...

Delimiter: ; Insert TAB

Enclosure: "

NIO buffer size: 50000

Lazy conversion? ☒

Header row present? ☒

Add filename to result? ☐

The row number field name (optional):

Running in parallel? ☐

New line possible in fields? ☐

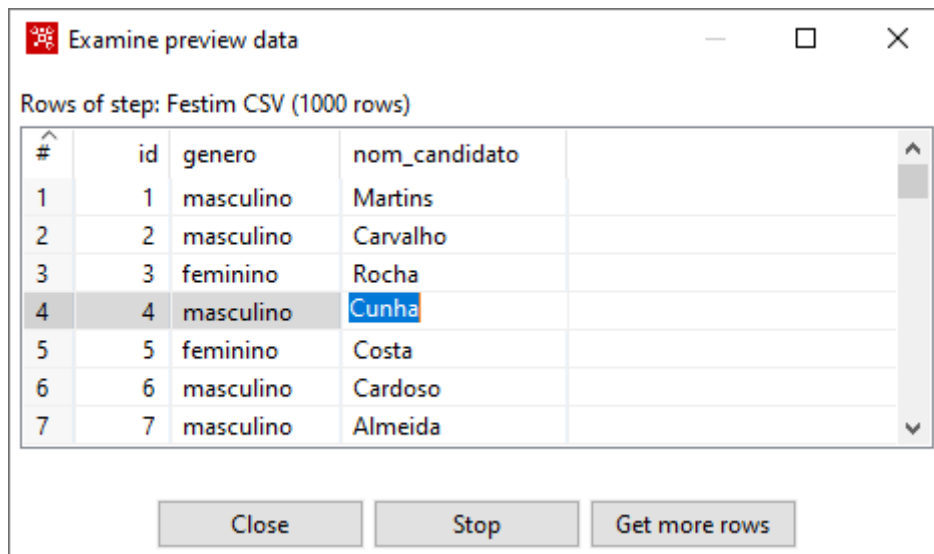
Format: mixed

File encoding: UTF-8

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	id	Integer	#	15	0				none
2	genero	String		11					both
3	nom_candidato	String		23					both
4									
5									
6									
7									

Help OK Get Fields Preview Cancel

2.8 Salve a transformação, execute novamente o processo e verifique na prévia que a situação foi contornada.



#	id	genero	nom_candidato
1	1	masculino	Martins
2	2	masculino	Carvalho
3	3	feminino	Rocha
4	4	masculino	Cunha
5	5	feminino	Costa
6	6	masculino	Cardoso
7	7	masculino	Almeida

Exercício 03 – Input Table

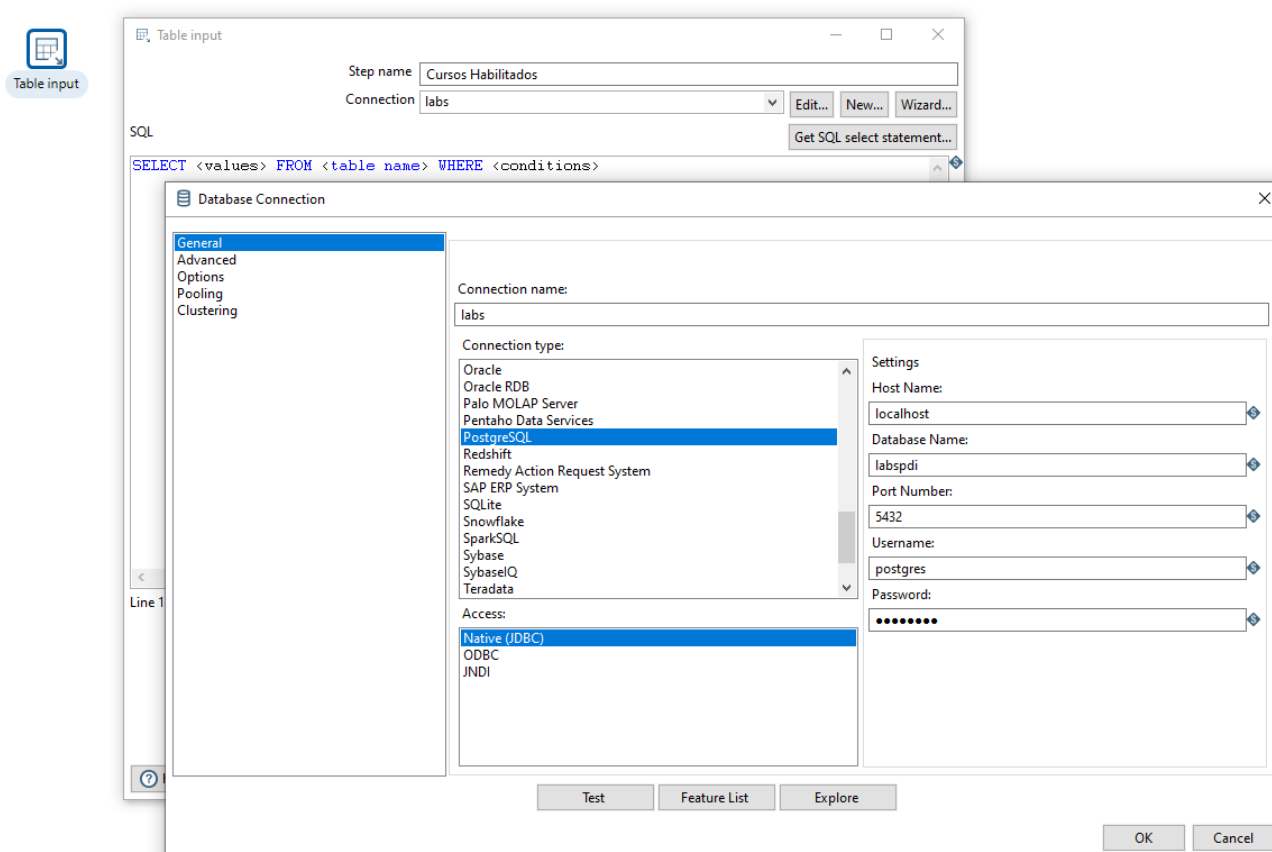
3.1 No PostgreSQL, dentro do banco de dados “labspdi”, crie um novo schema chamado “labs”. Neste esquema, crie uma tabela executando o código a seguir:

```
CREATE TABLE labs."TABLE_INPUT"
(
    "ID_CURSO" integer NOT NULL,
    "NOM_CURSO" character varying(200) COLLATE pg_catalog."default" NOT NULL,
    "CAMPUS" character varying COLLATE pg_catalog."default"
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

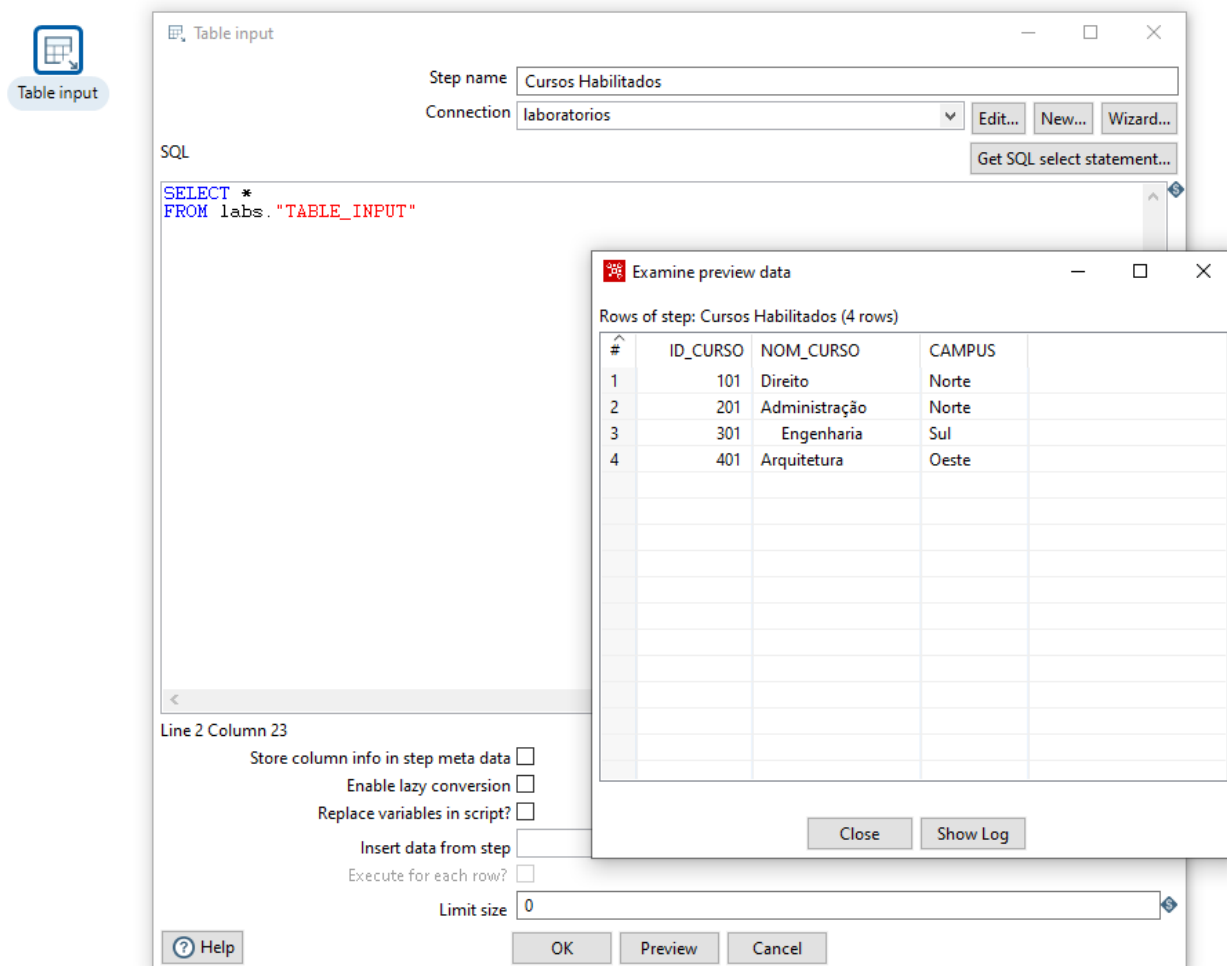
ALTER TABLE labs."TABLE_INPUT"
    OWNER to postgres;
```

```
INSERT INTO labs."TABLE_INPUT" VALUES (101,'Direito','Norte');
INSERT INTO labs."TABLE_INPUT" VALUES (201,'Administração','Norte');
INSERT INTO labs."TABLE_INPUT" VALUES (301,'Engenharia','Sul');
INSERT INTO labs."TABLE_INPUT" VALUES (401,'Arquitetura','Oeste');
```

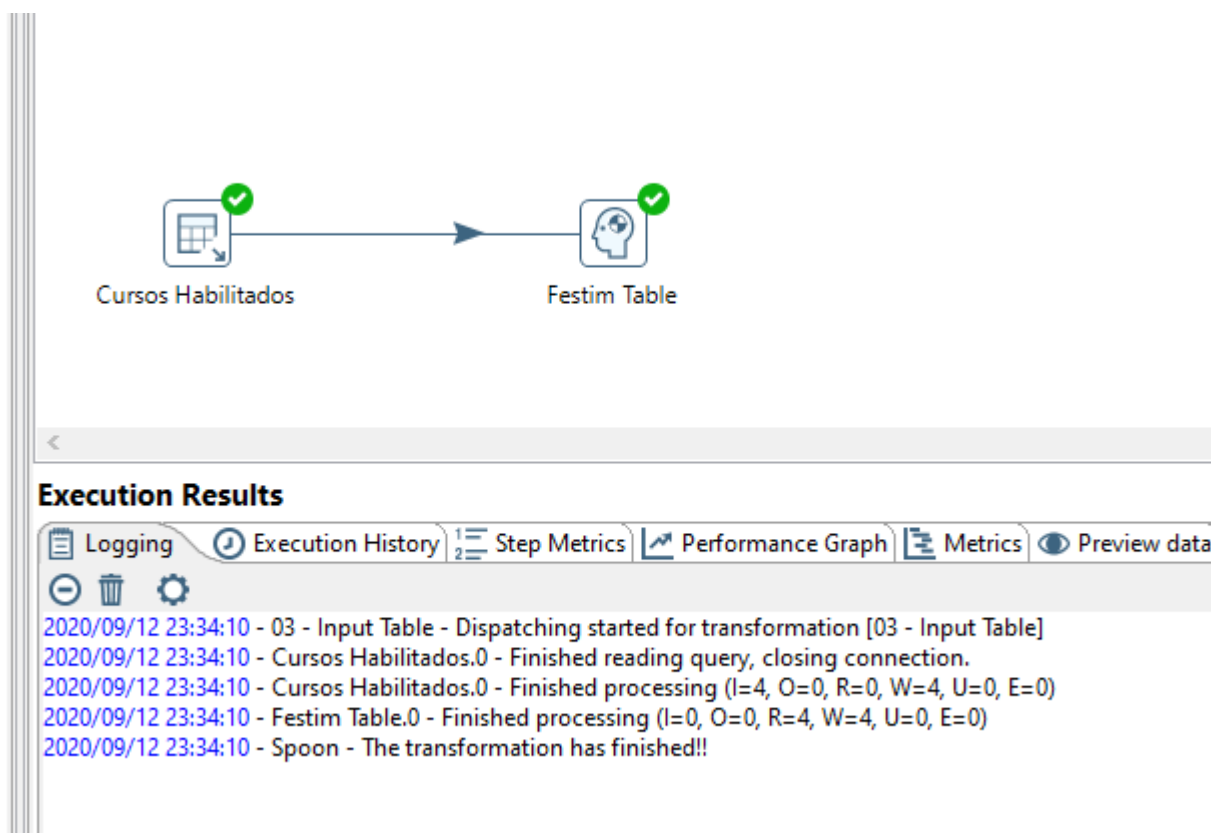
3.2 No PentahoDI, crie uma nova transformação e salve como “03 – Input Table”. Em seguida, traga o step “Table Input” para o ambiente gráfico da transformação e realize sua configuração:



- 3.3** Após a criação da conexão, clique em “Get SQL select statement” para buscar os dados.
IMPORTANTE: Ao selecionar a tabela de origem, colocar aspas duplas no nome da tabela, conforme imagem a seguir:



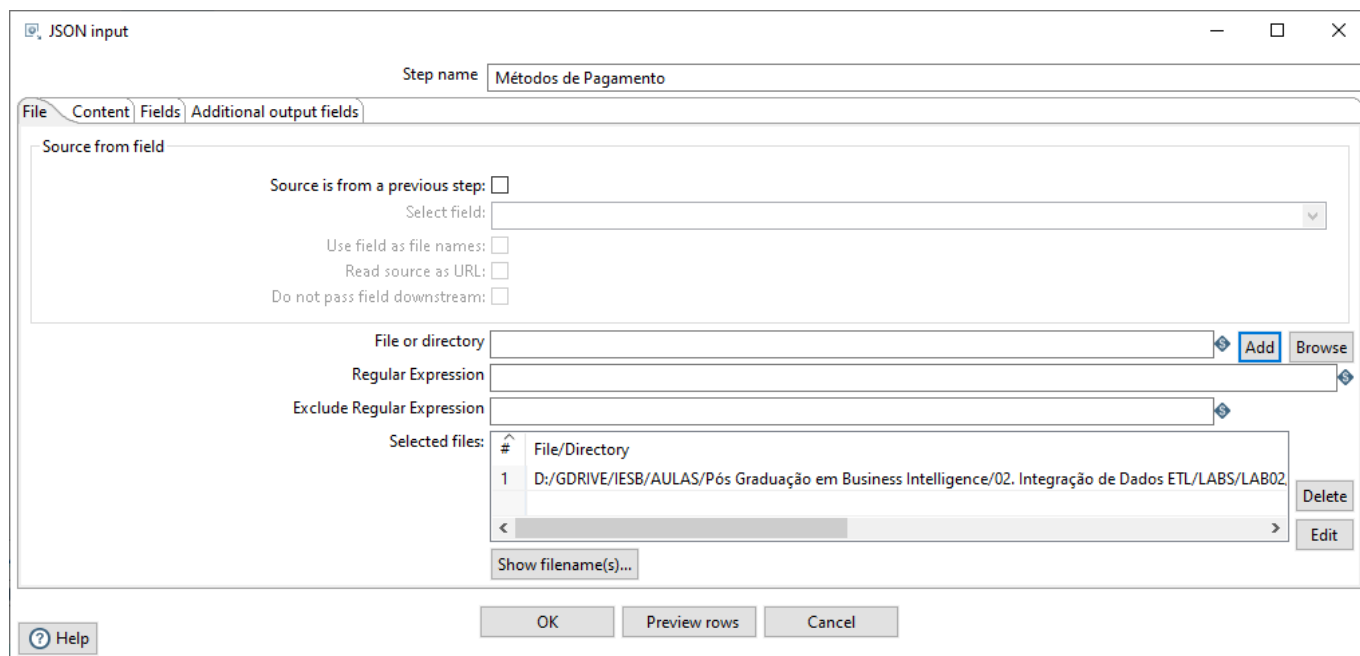
3.4 Adicione o step “Dummy”, altere seu nome, crie o HOP entre os steps, salve e execute a transformação:



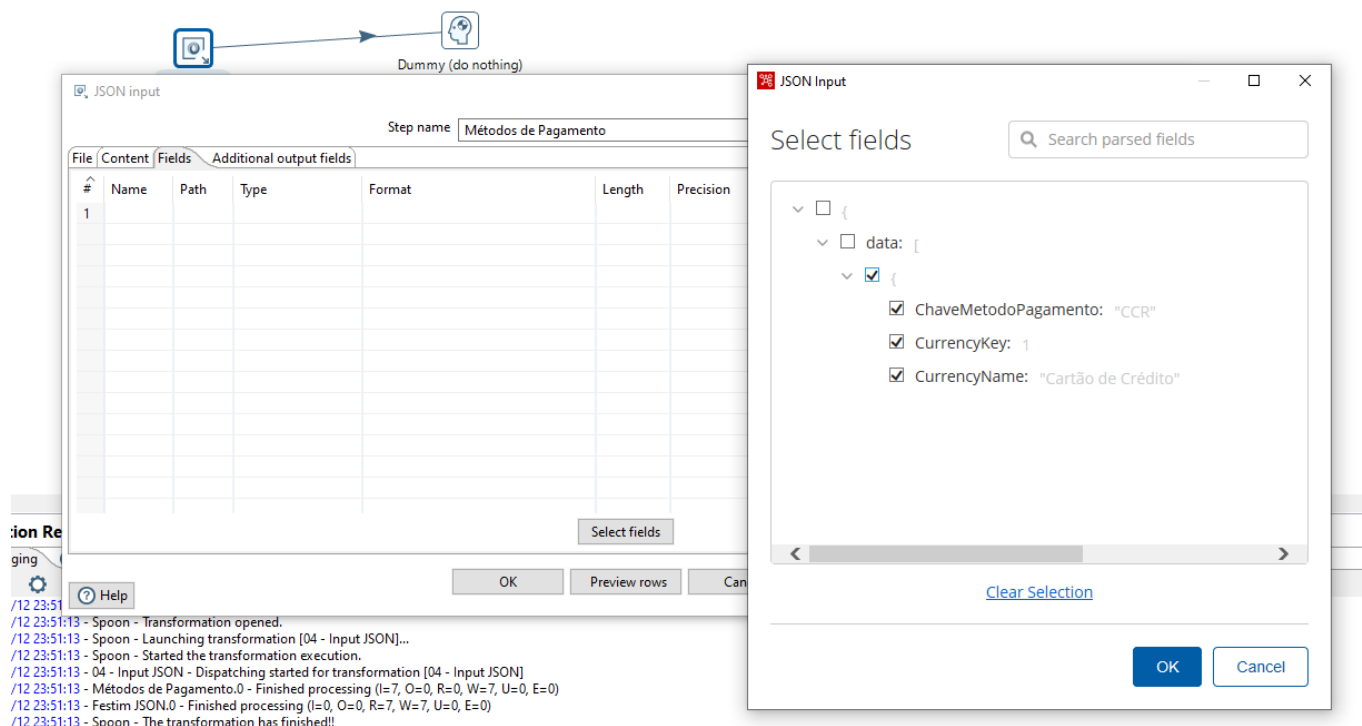
Exercício 04 – Input JSON

4.1 Crie uma nova transformação, chamada “04 – Input JSON”. Arraste o step “JSON input” e o “Dummy”, crie o HOP entre eles. Os passos abaixo indicam como configurar o arquivo de entrada:

- Mude o nome do step
- No botão “Browse”, selecione o arquivo de entrada
- Após selecionar o arquivo, clique no botão “Add”
- Vá para a aba “Fields”



- Clique em “Select Fields”
- Marque os campos de dados que serão carregados no step
- Valide a leitura na prévia dos dados



JSON input

Step name: Métodos de Pagamento

#	Name	Path	Type	Format	Length	Precision
1						

Select fields

Search parsed fields

data: {

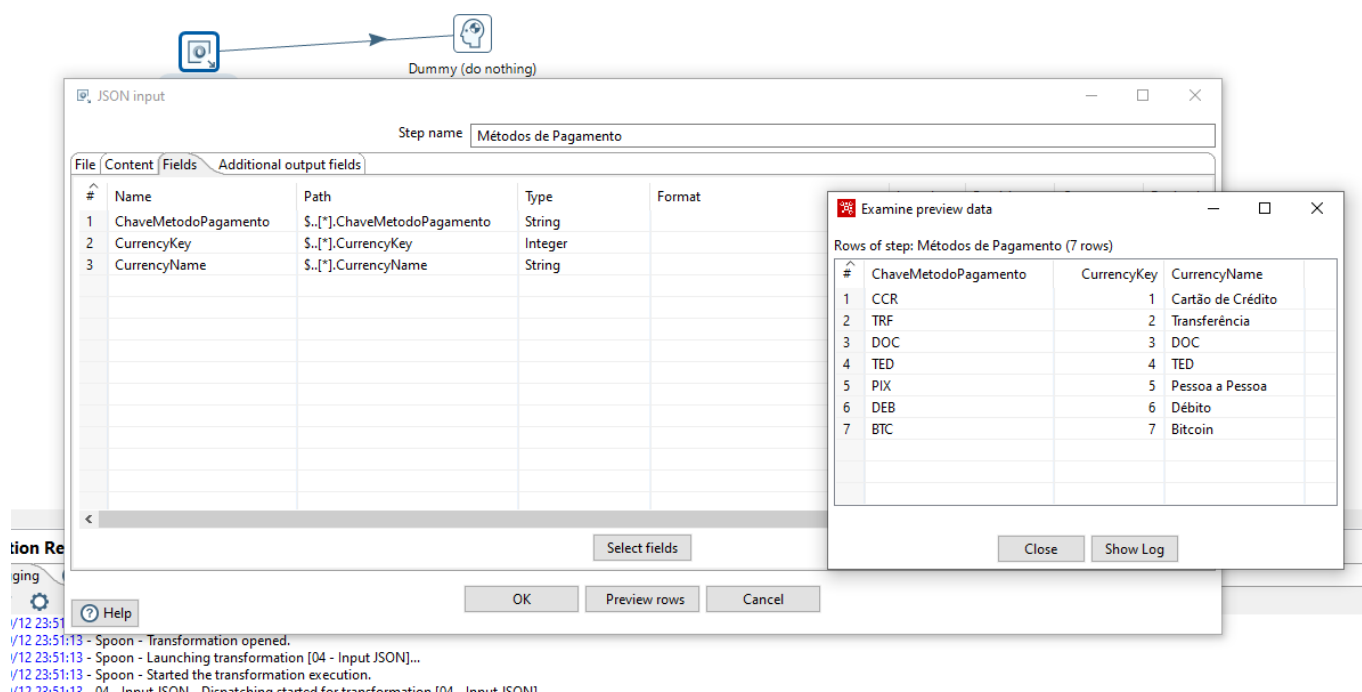
- ChaveMetodoPagamento: "CCR"
- CurrencyKey: 1
- CurrencyName: "Cartão de Crédito"

Clear Selection

OK Cancel

Help

/12 23:51:13 - Spoon - Transformation opened.
 /12 23:51:13 - Spoon - Launching transformation [04 - Input JSON]...
 /12 23:51:13 - Spoon - Started the transformation execution.
 /12 23:51:13 - 04 - Input JSON - Dispatching started for transformation [04 - Input JSON]
 /12 23:51:13 - Métodos de Pagamento.0 - Finished processing (I=7, O=0, R=0, W=7, U=0, E=0)
 /12 23:51:13 - Festim JSON.0 - Finished processing (I=0, O=0, R=7, W=7, U=0, E=0)
 /12 23:51:13 - Spoon - The transformation has finished!!



JSON input

Step name: Métodos de Pagamento

#	Name	Path	Type	Format
1	ChaveMetodoPagamento	\$.[*].ChaveMetodoPagamento	String	
2	CurrencyKey	\$.[*].CurrencyKey	Integer	
3	CurrencyName	\$.[*].CurrencyName	String	

Select fields

OK Preview rows Cancel

Examine preview data

Rows of step: Métodos de Pagamento (7 rows)

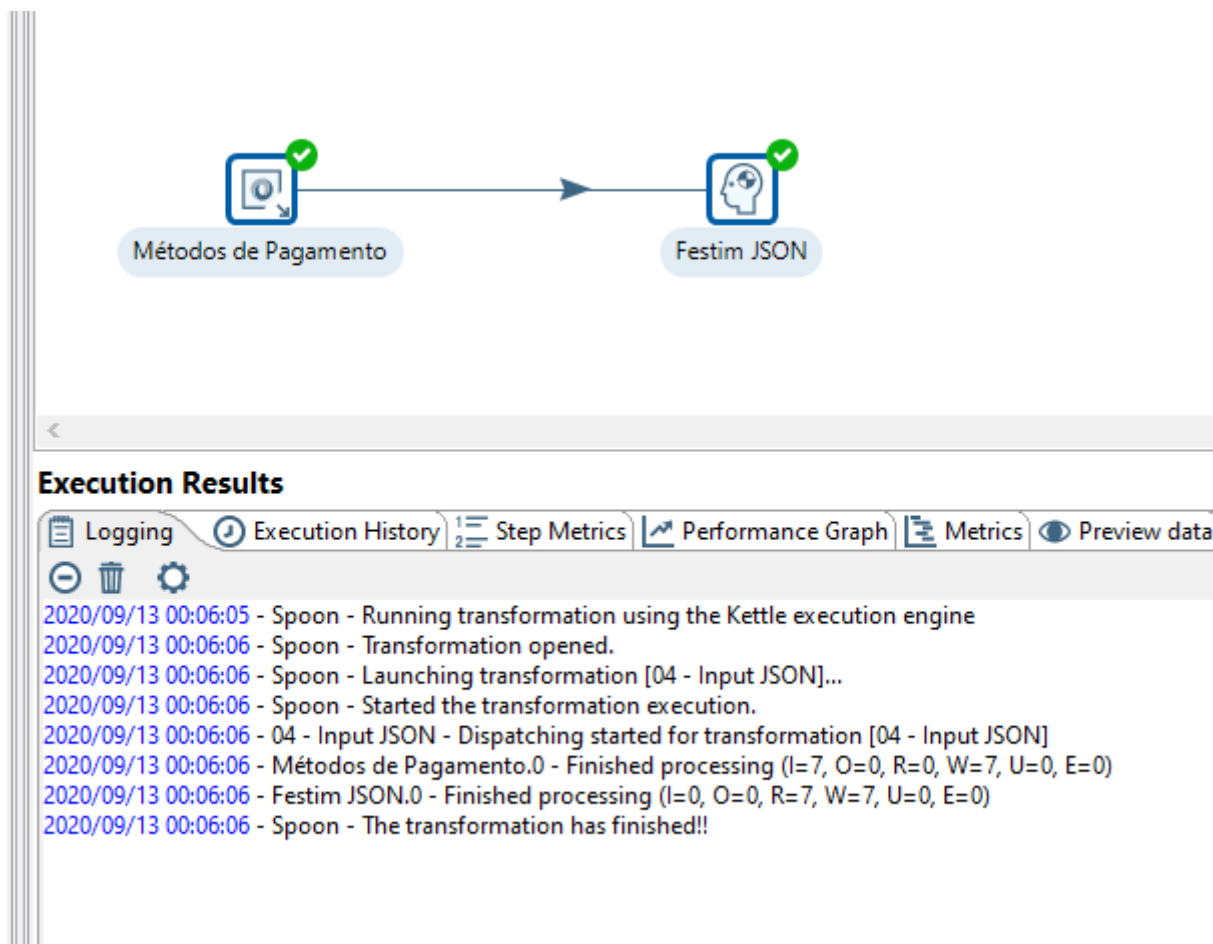
#	ChaveMetodoPagamento	CurrencyKey	CurrencyName
1	CCR	1	Cartão de Crédito
2	TRF	2	Transferência
3	DOC	3	DOC
4	TED	4	TED
5	PIX	5	Pessoa a Pessoa
6	DEB	6	Débito
7	BTC	7	Bitcoin

Close Show Log

Help

/12 23:51:13 - Spoon - Transformation opened.
 /12 23:51:13 - Spoon - Launching transformation [04 - Input JSON]...
 /12 23:51:13 - Spoon - Started the transformation execution.
 /12 23:51:13 - 04 - Input JSON - Dispatching started for transformation [04 - Input JSON]

4.2 Salve e execute a transformação:

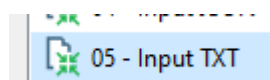


The screenshot displays the Kettle transformation interface. At the top, a flow diagram shows two steps: 'Métodos de Pagamento' (represented by a document icon) and 'Festim JSON' (represented by a head icon with a brain), connected by a right-pointing arrow. Both steps have a green checkmark in the top right corner, indicating successful execution. Below the flow diagram is a section titled 'Execution Results'. This section contains a horizontal menu with tabs for 'Logging', 'Execution History', 'Step Metrics', 'Performance Graph', 'Metrics', and 'Preview data'. The 'Logging' tab is currently selected. Below the tabs, there are icons for a minus sign, a trash can, and a gear. The logging text shows a series of timestamps and messages: '2020/09/13 00:06:05 - Spoon - Running transformation using the Kettle execution engine', '2020/09/13 00:06:06 - Spoon - Transformation opened.', '2020/09/13 00:06:06 - Spoon - Launching transformation [04 - Input JSON]...', '2020/09/13 00:06:06 - Spoon - Started the transformation execution.', '2020/09/13 00:06:06 - 04 - Input JSON - Dispatching started for transformation [04 - Input JSON]', '2020/09/13 00:06:06 - Métodos de Pagamento.0 - Finished processing (I=7, O=0, R=0, W=7, U=0, E=0)', '2020/09/13 00:06:06 - Festim JSON.0 - Finished processing (I=0, O=0, R=7, W=7, U=0, E=0)', and '2020/09/13 00:06:06 - Spoon - The transformation has finished!!'.

EXERCÍCIOS EXTRAS

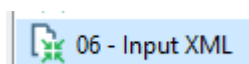
Exercício 05

- 5.1** Crie os mesmos processos, utilizando o input “Text file input”, utilizando sempre o Dummy como saída e salvando no repositório esta transformação:



Exercício 06

- 6.1** Crie os mesmos processos, utilizando o input “Get data from XML input”, utilizando sempre o Dummy como saída e salvando no repositório esta transformação:



Exercício 07

- 7.1** Crie os mesmos processos, utilizando o input “Microsoft Excel input”, utilizando sempre o Dummy como saída e salvando no repositório esta transformação:

