

- 1) Crie uma nova transformação, chamada **FatoDataMart**.
- 2) A esta transformação, acrescente um *step* do tipo **Table input**, com o nome de **Leitura das Fatos Rateadas**, com as propriedades abaixo:

Table input

Step name: **Leitura das Fatos Rateadas**

Connection: **Datawarehouse Sucos** [Edit... New... Wizard...]

SQL [Get SQL select statement...]

```

SELECT X.Cod_Cliente
      ,X.Cod_Produto
      ,X.Cod_Organizacional
      ,X.Cod_Fabrica
      ,X.Cod_Tempo
      ,X.Faturamento
      ,X.Imposto
      ,X.Custo_Variavel
      ,X.Quantidade_Vendida
      ,X.Unidade_Vendida
      ,(X.Quantidade_Vendida/Y.Quantidade_Vendida_002) * Y.Custo_Frete AS Frete_Rateio
      ,(X.Quantidade_Vendida/W.Quantidade_Vendida_003) * W.Custo_Fixo AS Custo_Fixo_Rateio
      ,(X.Quantidade_Vendida/Z.Quantidade_Vendida_004) * Z.Meta_Faturamento AS Meta_Faturamento_Rateio
      ,(X.Quantidade_Vendida/K.Quantidade_Vendida_005) * K.Meta_Custo AS Meta_Custo_Rateio
FROM Fato_001 X
INNER JOIN (
SELECT A.Cod_Cliente
      ,A.Cod_Produto
      ,A.Cod_Fabrica
      ,A.Cod_Tempo
      ,A.Custo_Frete
      ,B.Quantidade_Vendida_002
FROM Fato_002 A
INNER JOIN (
SELECT Cod_Cliente
      ,Cod_Produto
      ,Cod_Fabrica
      ,Cod_Tempo
      ,SUM(Quantidade_Vendida) AS Quantidade_Vendida_002
FROM Fato_001
GROUP BY Cod_Cliente
      ,Cod_Produto
      ,Cod_Fabrica
      ,Cod_Tempo) B
ON A.Cod_Cliente = B.Cod_Cliente AND A.Cod_Produto = B.Cod_Produto
AND A.Cod_Fabrica = B.Cod_Fabrica AND A.Cod_Tempo = B.Cod_Tempo ) Y
ON X.Cod_Cliente = Y.Cod_Cliente AND X.Cod_Produto = Y.Cod_Produto
AND X.Cod_Fabrica = Y.Cod_Fabrica AND X.Cod_Tempo = Y.Cod_Tempo
INNER JOIN (
SELECT A.Cod_Fabrica

```

Line 1 Column 0

Enable lazy conversion ☐

Replace variables in script? ☒

Insert data from step

Execute for each row? ☐

Limit size

[Help] [OK] [Preview] [Cancel]

A consulta contida no *step* é a seguinte:

```

SELECT X.Cod_Cliente
      ,X.Cod_Produto
      ,X.Cod_Organizacional
      ,X.Cod_Fabrica

```

```

,X.Cod_Tempo
,X.Faturamento
,X.Imposto
,X.Custo_Variavel
,X.Quantidade_Vendida

,X.Unidade_Vendida
,(X.Quantidade_Vendida/Y.Quantidade_Vendida_002) * Y.Custo_Frete
,(X.Quantidade_Vendida/W.Quantidade_Vendida_003) * W.Custo_Fixo
,(X.Quantidade_Vendida/Z.Quantidade_Vendida_004) * Z.Meta_Fatura
,(X.Quantidade_Vendida/K.Quantidade_Vendida_005) * K.Meta_Custo
FROM Fato_001 X
INNER JOIN (
SELECT A.Cod_Cliente
      ,A.Cod_Produto
      ,A.Cod_Fabrica
      ,A.Cod_Tempo
      ,A.Custo_Frete
      , B.Quantidade_Vendida_002
FROM Fato_002 A
INNER JOIN (
SELECT Cod_Cliente
      ,Cod_Produto
      ,Cod_Fabrica
      ,Cod_Tempo
      ,SUM(Quantidade_Vendida) AS Quantidade_Vendida_002
FROM Fato_001
GROUP BY Cod_Cliente
      ,Cod_Produto
      ,Cod_Fabrica
      ,Cod_Tempo) B
ON A.Cod_Cliente = B.Cod_Cliente AND A.Cod_Produto = B.Cod_Produto
AND A.Cod_Fabrica = B.Cod_Fabrica AND A.Cod_Tempo = B.Cod_Tempo
ON X.Cod_Cliente = Y.Cod_Cliente AND X.Cod_Produto = Y.Cod_Produto
AND X.Cod_Fabrica = Y.Cod_Fabrica AND X.Cod_Tempo = Y.Cod_Tempo
INNER JOIN (
SELECT A.Cod_Fabrica
      ,A.Cod_Tempo
      ,A.Custo_Fixo

```

```

        , B.Quantidade_Vendida_003
FROM Fato_003 A
INNER JOIN (
    SELECT Cod_Fabrica
        ,Cod_Tempo

        ,SUM(Quantidade_Vendida) AS Quantidade_Vendida_003
FROM Fato_001
GROUP BY Cod_Fabrica
        ,Cod_Tempo) B
ON A.Cod_Fabrica = B.Cod_Fabrica AND A.Cod_Tempo = B.Cod_Tempo
ON X.Cod_Fabrica = W.Cod_Fabrica AND X.Cod_Tempo = W.Cod_Tempo
inner join (
    SELECT A.Cod_Cliente
        ,A.Cod_Produto
        ,A.Cod_Organizacional
        ,A.Cod_Tempo
        ,A.Meta_Faturamento
        ,B.Quantidade_Vendida_004
FROM Fato_004 A
INNER JOIN (
    SELECT Cod_Cliente
        ,Cod_Produto
        ,Cod_Organizacional
        ,Cod_Tempo
        ,SUM(Quantidade_Vendida) AS Quantidade_Vendida_004
FROM Fato_001
GROUP BY Cod_Cliente
        ,Cod_Produto
        ,Cod_Organizacional
        ,Cod_Tempo) B ON
A.Cod_Cliente = B.Cod_Cliente AND A.Cod_Produto = B.Cod_Produto
AND A.Cod_Tempo = B.Cod_Tempo ) Z
ON X.Cod_Cliente = Z.Cod_Cliente AND X.Cod_Produto = Z.Cod_Produto
AND X.Cod_Tempo = Z.Cod_Tempo
INNER JOIN (
    SELECT A.Cod_Produto
        ,A.Cod_Fabrica
        ,A.Cod_Tempo

```

COPIAR CÓDIGO

Calculator

Step name

☒ Throw an error on non existing files

#	New field	Calculation	Field A	Field B	Field C	Value type	Length	Precision	Remove	Conversion mask	Decir
1	DATA_DATE	Create a copy of field A	Cod_Tempo			Date			N	yyyyMMdd	
2	DATA_FINAL	Create a copy of field A	DATA_DATE			Date			N	dd/MM/yyyy	

< >

? Help OK Cancel

4) Crie um novo *step*, do tipo **Database Lookup**, com o nome **Acha ID Fábrica**. Ligue o *step* **Conversão de Data** a ele e insira as propriedades abaixo:

Database lookup

Step name:

Connection:

Lookup schema:

Lookup table:

Enable cache? ☐

Cache size in rows (0=cache on):

Load all data from table ☐

The key(s) to look up the value(s):

#	Table field	Comparator	Field1	Field2
1	Cod_Fabrica	=	Cod_Fabrica	

Values to return from the lookup table:

#	Field	New name	Default	Type
1	ID_Fabrica		0	None

Do not pass the row if the lookup fails ☐

Fail on multiple results? ☐

Order by:

5) Crie um novo *step*, do tipo **Database Lookup**, com o nome **Acha ID de Tempo**. Ligue o *step* **Acha ID Fábrica** a ele e insira as propriedades abaixo:

Database lookup

Step name:

Connection:

Lookup schema:

Lookup table:

Enable cache? ☐

Cache size in rows (0=cache):

Load all data from table ☐

The key(s) to look up the value(s):

#	Table field	Comparator	Field1	Field2
1	Cod_Tempo	=	Cod_Tempo	

Values to return from the lookup table :

#	Field	New name	Default	Type
1	ID_Tempo		0	None

Do not pass the row if the lookup fails ☐

Fail on multiple results? ☐

Order by:

6) Crie um novo *step*, do tipo **Database Lookup**, com o nome **Acha ID de Produto**. Ligue o *step* **Acha ID de Tempo** a ele e insira as propriedades abaixo:

Database lookup

Step name:

Connection:

Lookup schema:

Lookup table:

Enable cache? ☐

Cache size in rows (0=cache):

Load all data from table ☐

The key(s) to look up the value(s):

#	Table field	Comparator	Field1	Field2
1	Cod_Produto	=	Cod_Produto	

Values to return from the lookup table :

#	Field	New name	Default	Type
1	ID_Produto		0	None

Do not pass the row if the lookup fails ☐

Fail on multiple results? ☐

Order by

7) Crie um novo *step*, do tipo **Dimension lookup/update**, com o nome **Acha ID de Organizacional**. Ligue o *step* **Acha ID de Produto** a ele e insira as propriedades abaixo:

Dimension lookup/update

Step name: **Acha ID de Organizacional**

Update the dimension? ☐

Connection: Data Mart Presidência Edit... New... Wizard...

Target schema: Browse...

Target table: dim_organizacional Browse...

Commit size: 100

Enable the cache? ☒

Pre-load the cache? ☐

Cache size in rows (0 = cache all): 5000

Keys **Fields**

Key fields (to look up row in dimension):

#	Dimension field	Field in stream
1	Cod_Vendedor	Cod_Organizacional

Technical key field: ID_Vendedor New name

Creation of technical key

☒ Use table maximum + 1

☐ Use sequence

☐ Use auto increment field

Version field: Versao

Stream Datefield: DATA_FINAL

Date range start field: DataInicial Min. year: 1900

Use an alternative start date? ☐ <Select Option>

Table date range end: DataFinal Max. year: 2199

OK Cancel Get Fields SQL

Help

8) Crie um novo *step*, do tipo **Dimension lookup/update**, com o nome **Acha ID Cliente**. Ligue o *step* **Acha ID de Organizacional** a ele e insira as propriedades abaixo:

Dimension lookup/update

Step name:

Update the dimension? ☐

Connection:

Target schema:

Target table:

Commit size:

Enable the cache? ☒

Pre-load the cache? ☐

Cache size in rows (0 = cache all):

Keys Fields

Key fields (to look up row in dimension):

#	Dimension field	Field in stream
1	Cod_Cliente	Cod_Cliente

Technical key field: New name:

Creation of technical key

☒ Use table maximum + 1

☐ Use sequence

☐ Use auto increment field

Version field:

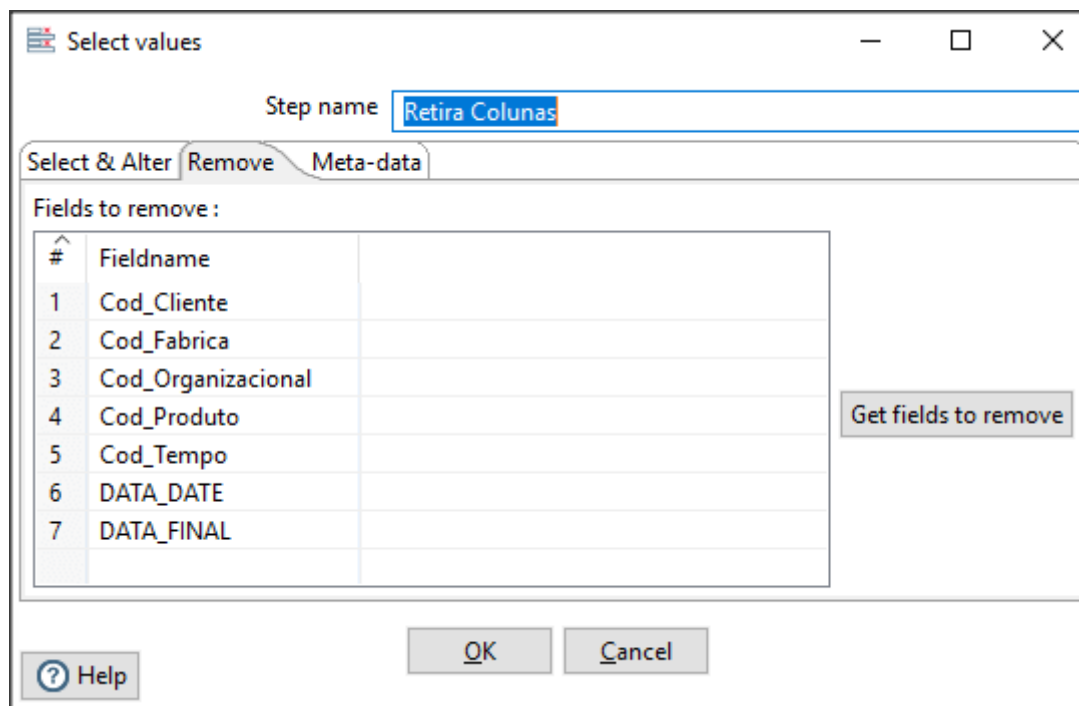
Stream Datefield:

Date range start field: Min. year:

Use an alternative start date? ☐

Table date range end: Max. year:

9) Crie um novo *step*, do tipo **Select Values**, com o nome **Retirar Colunas**. Ligue o *step* **Acha ID Cliente** a ele e insira as propriedades abaixo:



10) Crie um novo *step*, do tipo **Insert/Update**, com o nome **Atualiza Fato**. Ligue o *step* **Retirar Colunas** a ele e insira as propriedades abaixo:

Insert / update

Step name:

Connection:

Target schema:

Target table:

Commit size:

Don't perform any updates: ☐

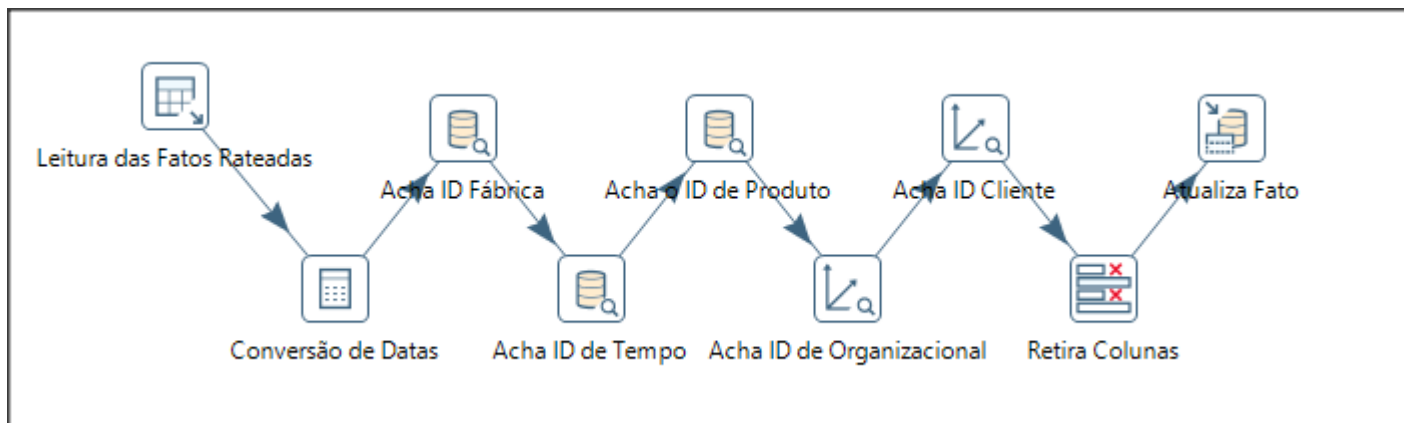
The key(s) to look up the value(s):

#	Table field	Comparator	Stream field1	Stream field2
1	ID_Cliente	=	ID_Cliente	
2	ID_Fabrica	=	ID_Fabrica	
3	ID_Produto	=	ID_Produto	
4	ID_Tempo	=	ID_Tempo	
5	ID_Vendedor	=	ID_Vendedor	

Update fields:

#	Table field	Stream field	Update
1	Faturamento	Faturamento	Y
2	Imposto	Imposto	Y
3	Custo_Variavel	Custo_Variavel	Y
4	Quantidade_Vendida	Quantidade_Vendida	Y
5	Unidade_Vendida	Unidade_Vendida	Y
6	Custo_Frete	Frete_Rateio	Y
7	Custo_Fixo	Custo_Fixo_Rateio	Y
8	Meta_Faturamento	Meta_Faturamento_Rateio	Y
9	Meta_Custo	Meta_Custo_Rateio	Y
10	ID_Fabrica	ID_Fabrica	N
11	ID_Tempo	ID_Tempo	N
12	ID_Produto	ID_Produto	N
13	ID_Vendedor	ID_Vendedor	N
14	ID_Cliente	ID_Cliente	N

11) Você terá o seguinte fluxo:

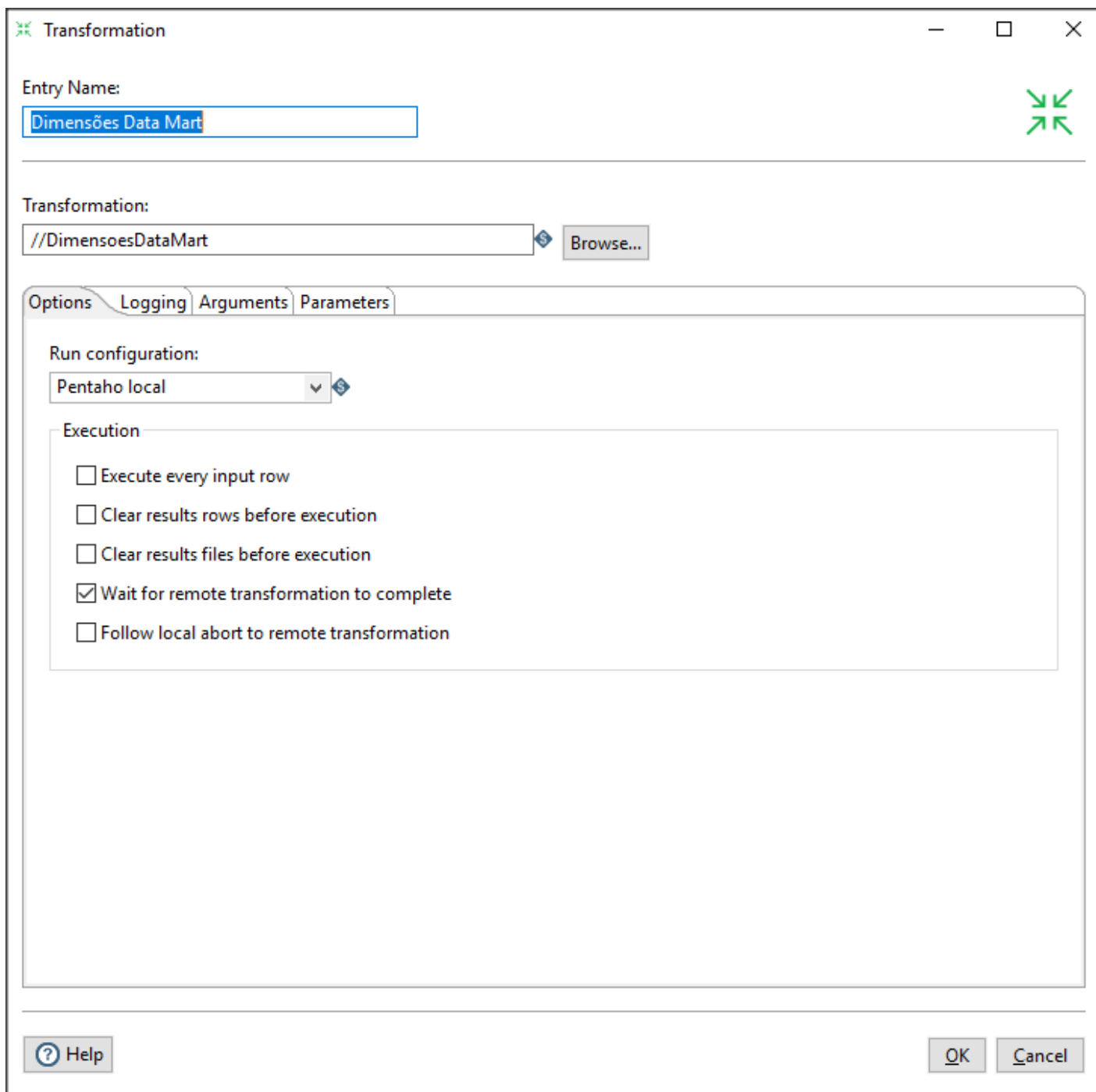


12) Salve a transformação e execute o processo de carga.

13) Crie um novo *job*, chamado **jobCargaDataMart**.

14) Adicione um *step* do tipo **Start**.

15) Adicione um *step* que execute a transformação **DimensoesDataMart**, e chame-o de **Dimensões Data Mart**. Ligue o *step* **Start** a este e preencha as suas propriedades:



The screenshot shows the 'Transformation' configuration window in Pentaho. The 'Entry Name' field is set to 'Dimensões Data Mart'. The 'Transformation' field is set to '//DimensoesDataMart'. The 'Options' tab is selected, showing the 'Run configuration' dropdown set to 'Pentaho local'. Under the 'Execution' section, the checkbox 'Wait for remote transformation to complete' is checked, while others are unchecked. The window has standard Windows controls (minimize, maximize, close) and buttons for 'Help', 'OK', and 'Cancel'.

Transformation

Entry Name:
Dimensões Data Mart

Transformation:
//DimensoesDataMart Browse...

Options Logging Arguments Parameters

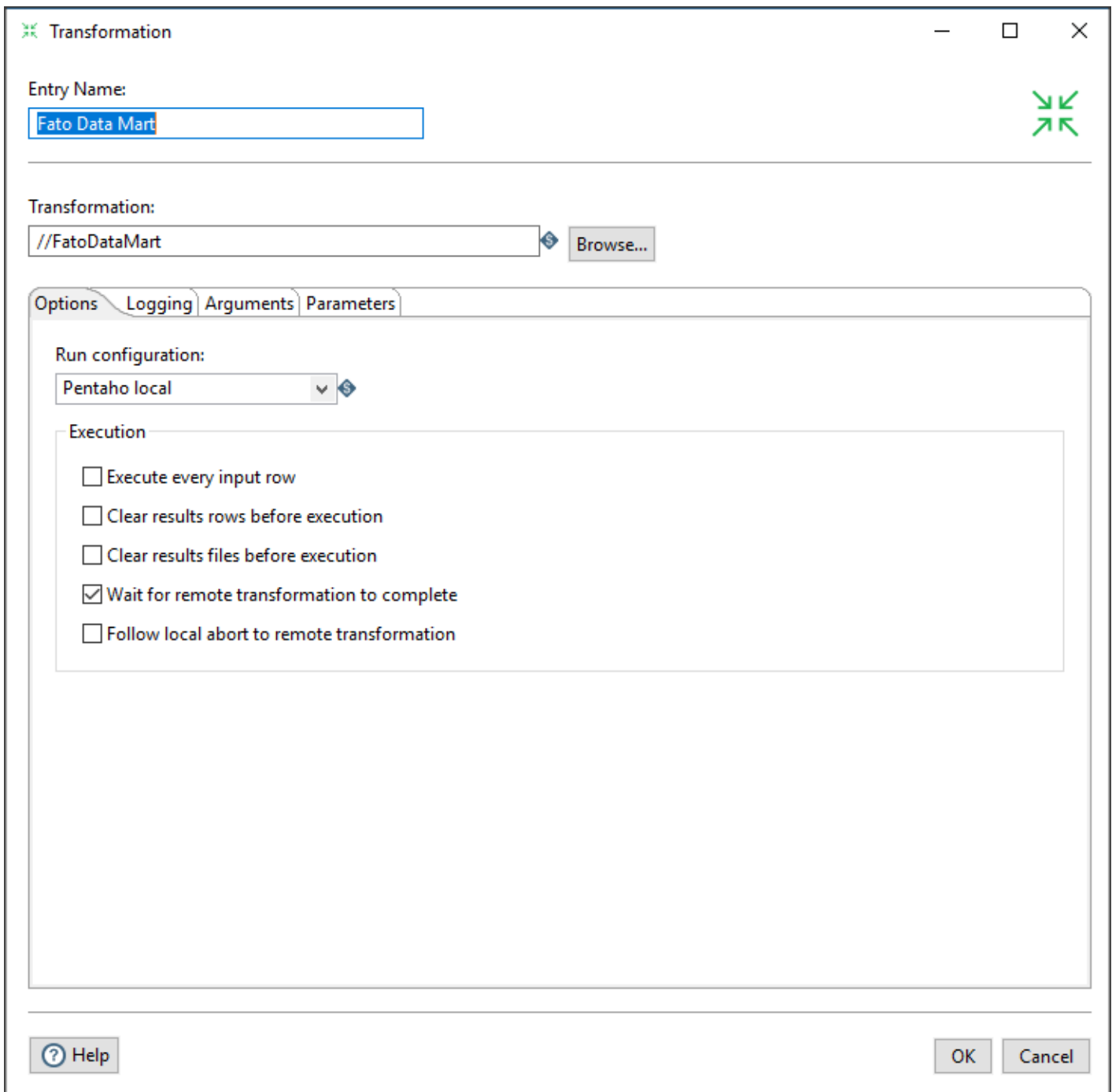
Run configuration:
Pentaho local

Execution

- ☐ Execute every input row
- ☐ Clear results rows before execution
- ☐ Clear results files before execution
- ☒ Wait for remote transformation to complete
- ☐ Follow local abort to remote transformation

Help OK Cancel

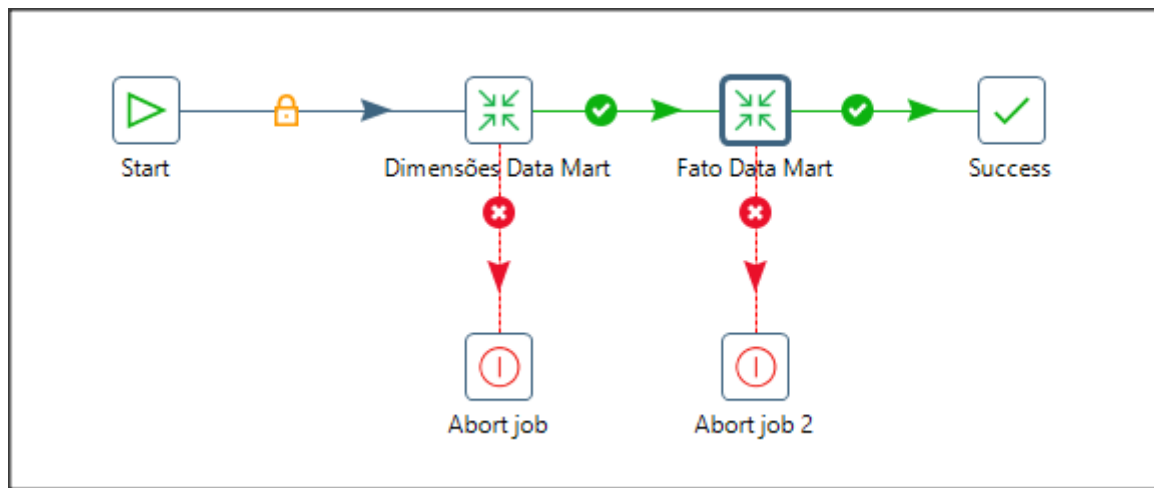
20) Acrescente um outro *step*, que executa a transformação **FatoDataMart**, e chame-o de **Fato Data Mart**. Ligue o *step* **Dimensões Data Mart** a ele, com a característica de execução, em caso de sucesso:



21) Ligue este último *step* ao *step* **Sucess**, para ser executado em caso de sucesso.

22) Saindo de **Dimensões Data Mart** e de **Fato Data Mart**, ligue conexões para *steps* do tipo **Abort**, para serem executados em caso de erros.

23) Você terá:

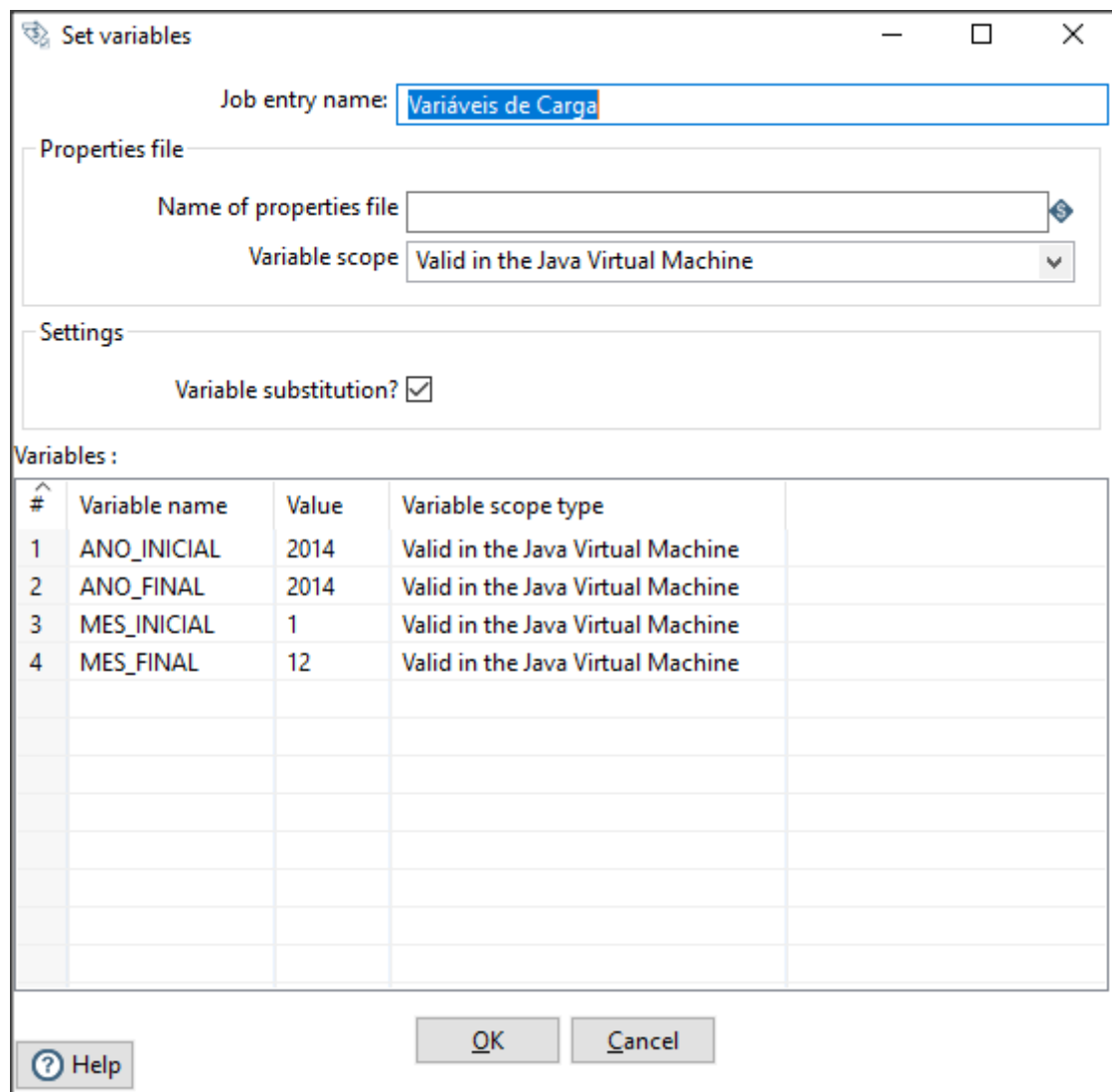


24) Salve e execute o processo completo.

25) Agora, orchestre o processo de carga do *Data Mart* com o processo de carga do *Data Warehouse*. Para isso, crie um novo job, chamado **jobCargaDwDm**.

26) Adicione o step **Start**.

27) Adicione outro step, do tipo **Set variables**, chamado **Variáveis de Carga**. Ligue o step **Start** a este e inclua as propriedades abaixo:



Set variables

Job entry name: Variáveis de Carga

Properties file

Name of properties file

Variable scope: Valid in the Java Virtual Machine

Settings

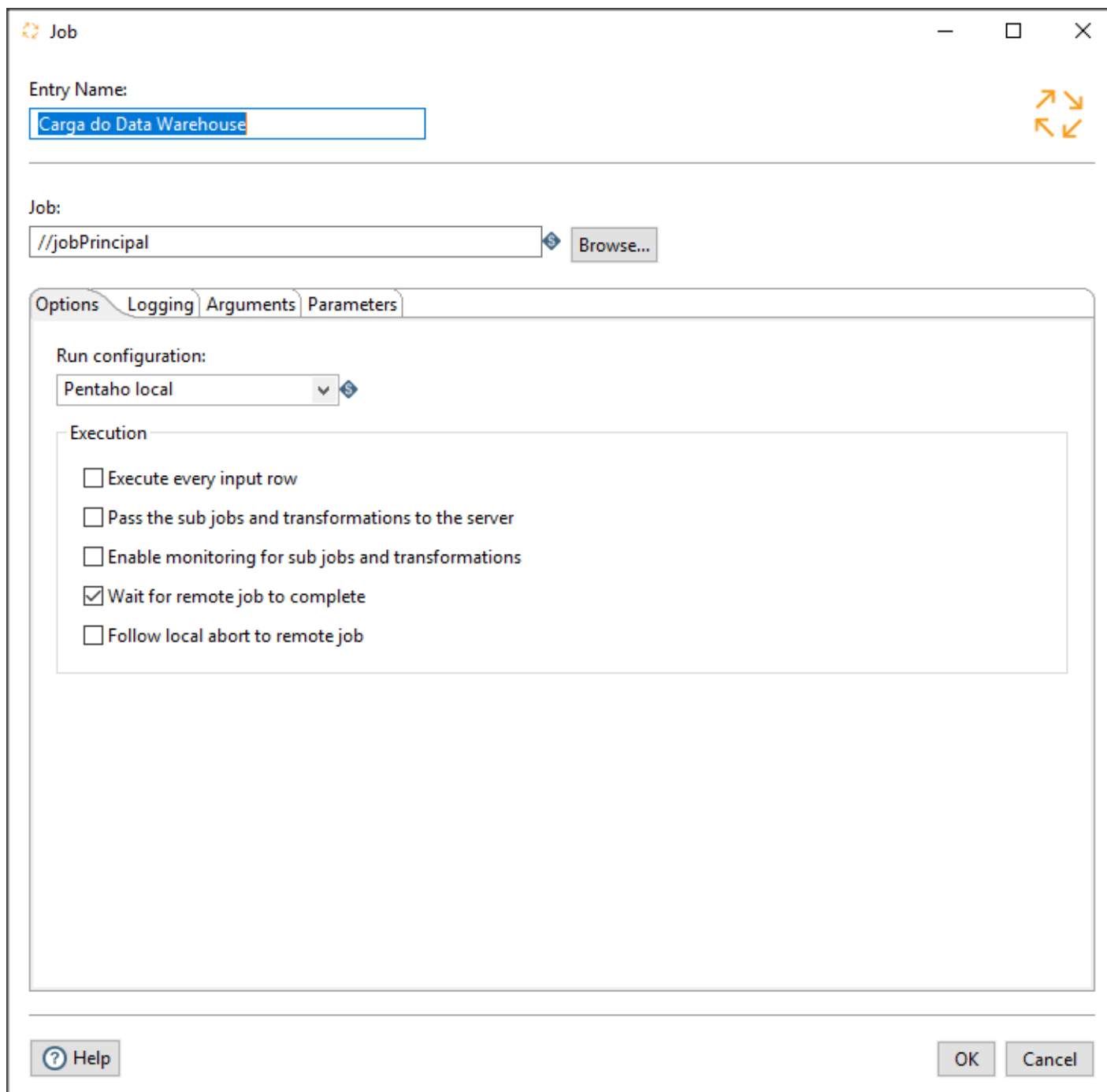
Variable substitution? ☒

Variables :

#	Variable name	Value	Variable scope type
1	ANO_INICIAL	2014	Valid in the Java Virtual Machine
2	ANO_FINAL	2014	Valid in the Java Virtual Machine
3	MES_INICIAL	1	Valid in the Java Virtual Machine
4	MES_FINAL	12	Valid in the Java Virtual Machine

Help OK Cancel

28) Adicione um novo *step* que execute um *job*, chamado **Carga do Data Warehouse**. Conecte o *step* **Variáveis de Carga** a ele e inclua as propriedades abaixo:



The image shows the 'Job' configuration window in Pentaho. The 'Entry Name' field is set to 'Carga do Data Warehouse'. The 'Job' field is set to '//jobPrincipal'. The 'Options' tab is selected, showing the 'Run configuration' dropdown set to 'Pentaho local'. Under the 'Execution' section, the following options are listed:

- ☐ Execute every input row
- ☐ Pass the sub jobs and transformations to the server
- ☐ Enable monitoring for sub jobs and transformations
- ☒ Wait for remote job to complete
- ☐ Follow local abort to remote job

At the bottom, there are 'Help', 'OK', and 'Cancel' buttons.

29) Adicione um novo *step*, chamado **Carga do Data Mart**, que irá executar o *job* **jobCargaDataMart**. Ligue o *step* **Carga do Data Warehouse** a ele, para ser executado em caso de sucesso. Além disso, inclua as propriedades abaixo:

Job

Entry Name:
Carga do Data Mart

Job:
//jobCargaDataMart Browse...

Options Logging Arguments Parameters

Run configuration:
Pentaho local

Execution

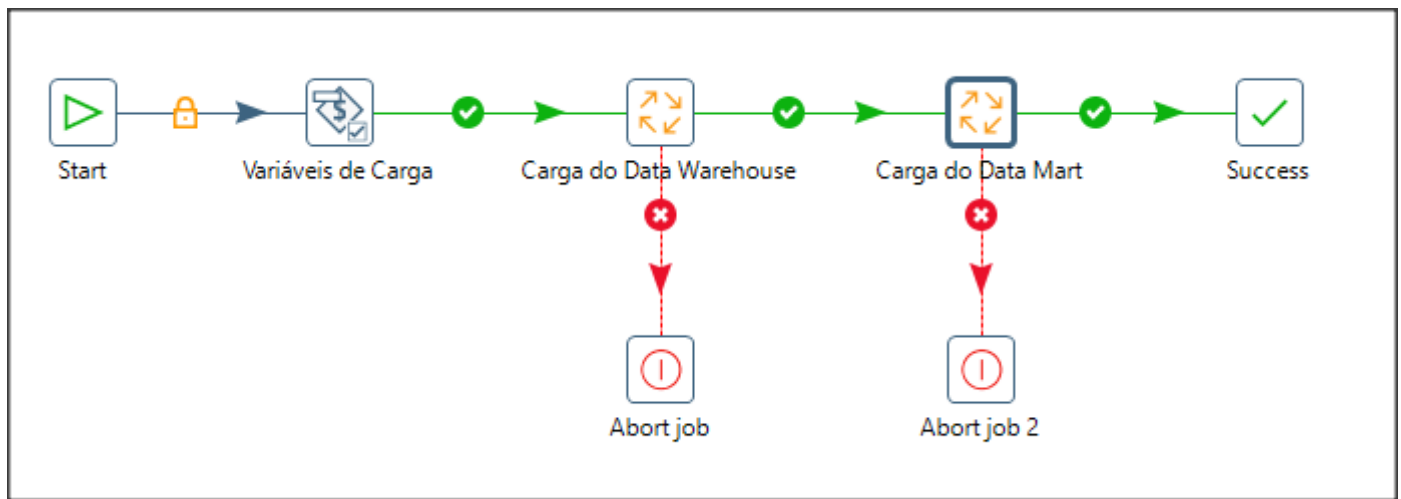
- ☐ Execute every input row
- ☐ Pass the sub jobs and transformations to the server
- ☐ Enable monitoring for sub jobs and transformations
- ☒ Wait for remote job to complete
- ☐ Follow local abort to remote job

Help OK Cancel

30) Ligue este último *step* a um *step Success*, para ser executado em caso de sucesso.

31) Ligue os *steps Carga do Data Warehouse* e *Carga do Data Mart* a *stepsAbort*, para serem executados em caso de erro.

32) Você terá o seguinte esquema:



33) Salve o *job* e execute o processo.