

APACHE HOP DATA WAREHOUSE

Grimaldo Oliveira

O que é o APACHE HOP

1. Abreviação de Hop Orchestration Platform, é uma plataforma de orquestração de dados e engenharia de dados que visa facilitar todos os aspectos da orquestração de dados e metadados.
2. A instalação padrão do Hop vem com cerca de 400 plugins ou componentes.
3. São criados fluxos de trabalho e pipelines em um ambiente de desenvolvimento visual chamado Hop Gui.
4. Com o APACHE HOP é possível combinar, enriquecer, limpar e de muitas outras maneiras manipular dados.

Sites Importantes

APACHE HOP

<https://hop.apache.org/>

Documentação

<https://hop.apache.org/manual/latest/getting-started/>

APACHE HOP Notícias

<https://hop.apache.org/blog/>

Arquitetura

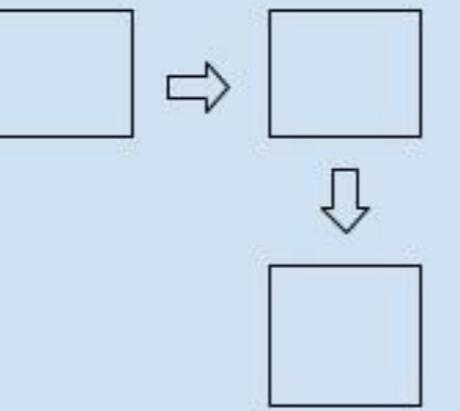
Como funciona a estrutura do APACHE HOP.

JAVA

APACHE HOP

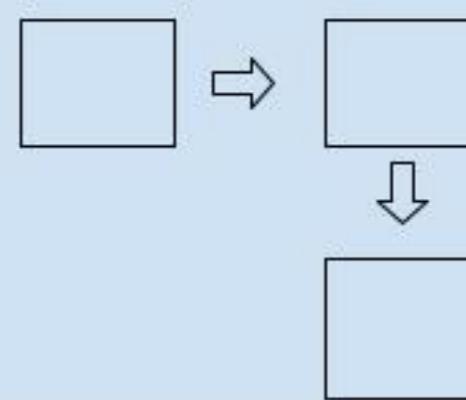
PIPELINE

TRANSFORMAÇÕES



WORKFLOW

FLUXO DE TRABALHO



APACHE HOP

Uma plataforma aberta para execução de pipelines e fluxos de trabalho.

APACHE HOP

Uma plataforma aberta destinada a criação de estruturas de ingestão de dados, tratamento de dados e fluxos de dados em diversas plataformas como bancos de dados relacionais, arquivos, bancos de dados NoSQL e dados na nuvem.

Tem por característica:

BASE DE DADOS: carga de grandes bases de dados, independente de plataforma seja on-premise ou cloud.

DATA WAREHOUSE: possui componentes ou plugins que facilitam a construção de projetos de Data Warehouse como Dimensões, controle de versionamento, criação de chaves artificiais, etc.

INTEGRAÇÃO: permite a integração e construção de projetos para popular bases heterogêneas, persistência em diversos banco de dados.

TRATAMENTO DE DADOS: perfilização e tratamento em dados de forma personalizada.

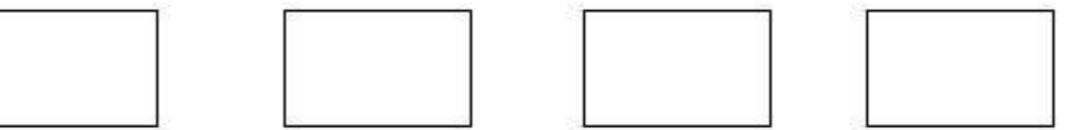
PROJETO

Faz todo o gerenciamento e
armazenamento dos metadados
criados no APACHE HOP.

PROJETO

Hop Projects é um agrupamento conceitual de configurações, variáveis, objetos metadados , fluxos de trabalho e pipelines. Pode haver mais de um ambiente dentro de um mesmo projeto no APACHE HOP.

DESENVOLVIMENTO



PRODUÇÃO



TESTE

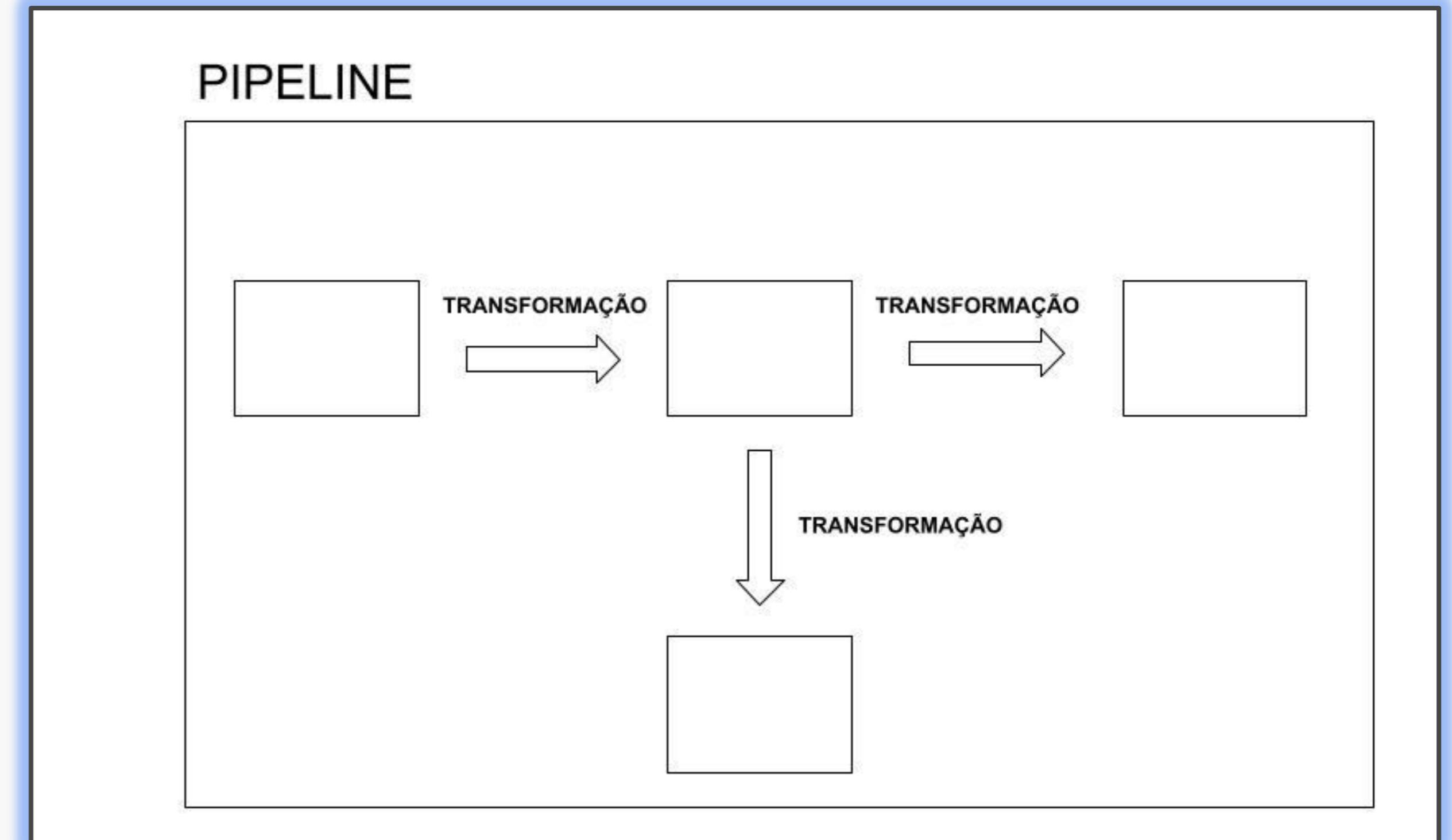


PIPELINE

São os componentes orquestrados de forma a resolver uma determinada operação.

PIPELINE

É uma composição de links e ações que são realizadas por quem desenvolve o pipeline. É aqui que os componentes são interligados e permitem a execução de uma determinada operação como limpeza, ajuste de campos, retirada de registros, gravação em banco de dados, etc.



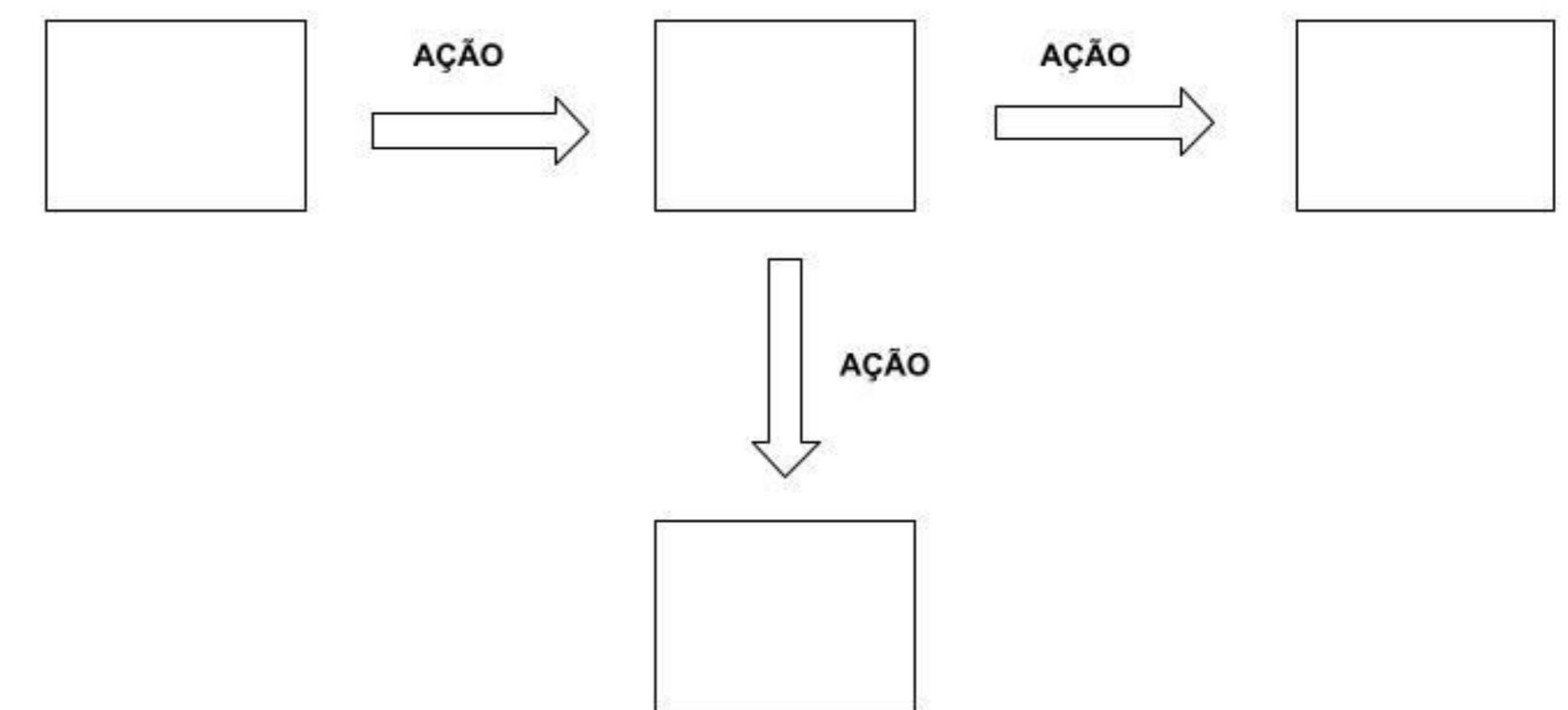
WORKFLOW

Determina uma sequência de operações.

WORKFLOW

É um orquestrador da sequência das operações criadas, também conhecido como fluxo de trabalho. Uma vez criados os pipelines, você poderá encadeá-los de forma a executá-los em ordem sequencial. Uma orquestração típica é verificar se o pipeline foi bem sucedido ou não.

WORKFLOW



HOP GUI

É local de trabalho dentro do APACHE HOP.

INTERFACE

Um espaço de trabalho, um ambiente para acessar todos os seus projetos. O espaço de trabalho organiza objetos, e fornece acesso a dados e recursos computacionais, pipelines e workflow.



HOP GUI - MENUS

Desatracando o local de trabalho
dentro do APACHE HOP.

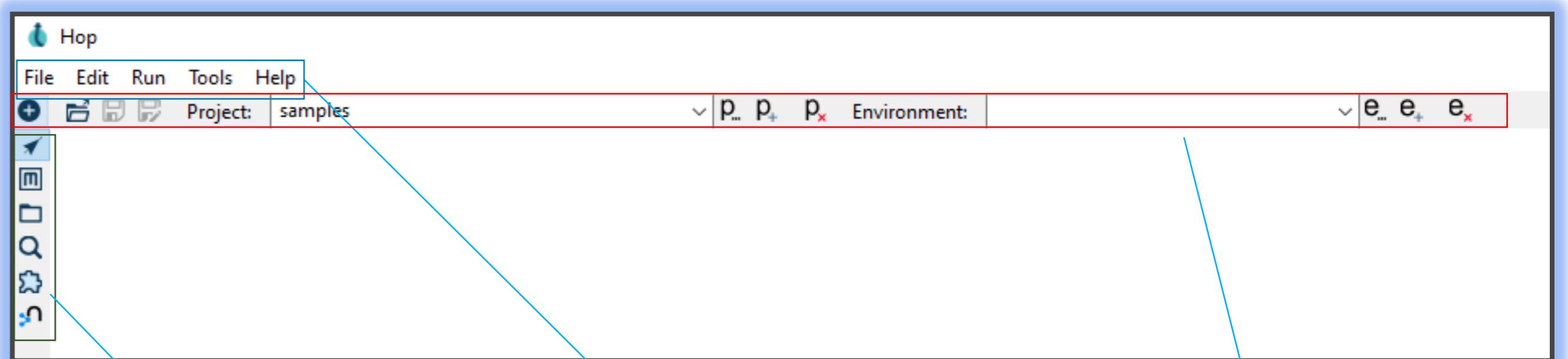
MENUS

Para o nosso trabalho temos três(03) menus principais .

Menu Barras: opções para o gerenciamento de pipelines e fluxos de trabalho.

Menu Principal: gerenciamento dos projetos, ambientes , etc.

Menu Perspectivas: Alterna entre opções nos outros menus com visual ampliado.



MENU PERSPECTIVAS

MENU BARRAS

MENU PRINCIPAL

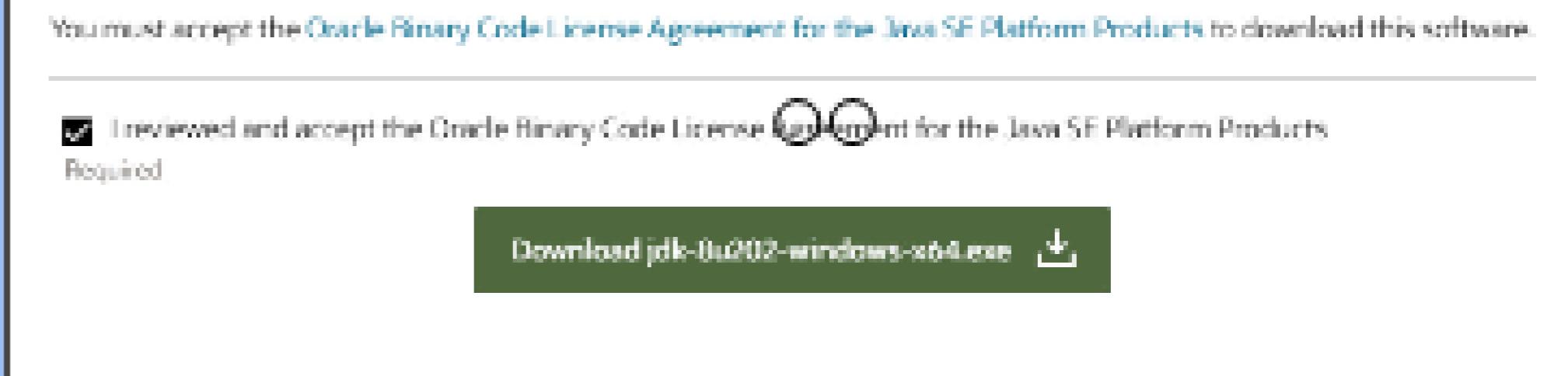
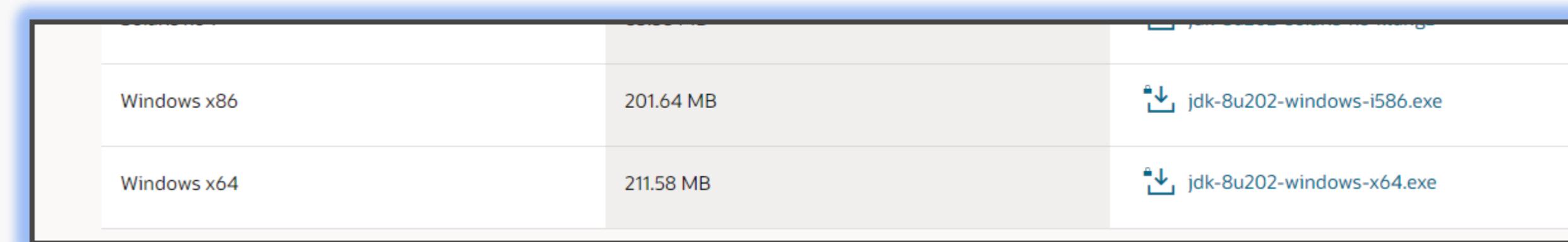
Instalação

Vamos nos preparar para a instalação do java jdk.

Instalação APACHE HOP

Vamos baixar o Java JDK e a aplicação APACHE HOP.

<https://www.oracle.com/java/technologies/javase/javase8-archive-downloads.html>



Instalação

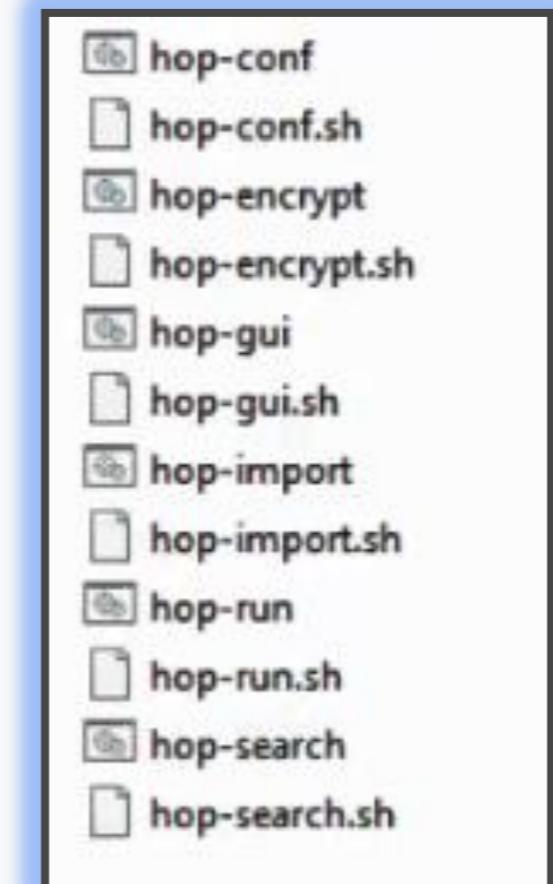
Vamos nos preparar para a instalação
do APACHE HOP.

Instalação APACHE HOP

Vamos baixar o Java JDK e a aplicação APACHE HOP.

<https://hop.apache.org/download/>

Version	Date	Description	Download Link	PGP Signature file of download	SHA512 Checksum file of download
1.0.0	October, 5th, 2021	Binaries	apache-hop-client-1.0.0-incubating.zip	asc	sha512
		Sources	apache-hop-1.0.0-incubating-src.tar.gz	asc	sha512
0.99	August, 5th, 2021	Binaries	apache-hop-client-0.99-incubating.zip	asc	sha512
		Sources	apache-hop-0.99-incubating-src.tar.gz	asc	sha512



Atenção Importante!

Vamos preparar e entender ajustes
para executar o APACHE HOP.

Preparação antes de executar o APACHE HOP

Preste atenção na configuração depois de instalados JAVA e APACHE HOP.

```
:NORMALSTART
REM set java primary is HOP_JAVA_HOME Fallback to JAVA_HOME or default java
if not "%HOP_JAVA_HOME%"==""
    set _HOP_JAVA="%HOP_JAVA_HOME%\bin\java"
) else if not "%JAVA_HOME%"==""
    set _HOP_JAVA="%JAVA_HOME%\bin\java"
) else (
    set _HOP_JAVA=java
)
```

DriverData	C:\Windows\System32\Drivers\DriverData
HOP_JAVA_HOME	C:\Java\jdk1.8
JAVA_HOME	C:\Java\jdk1.8\bin

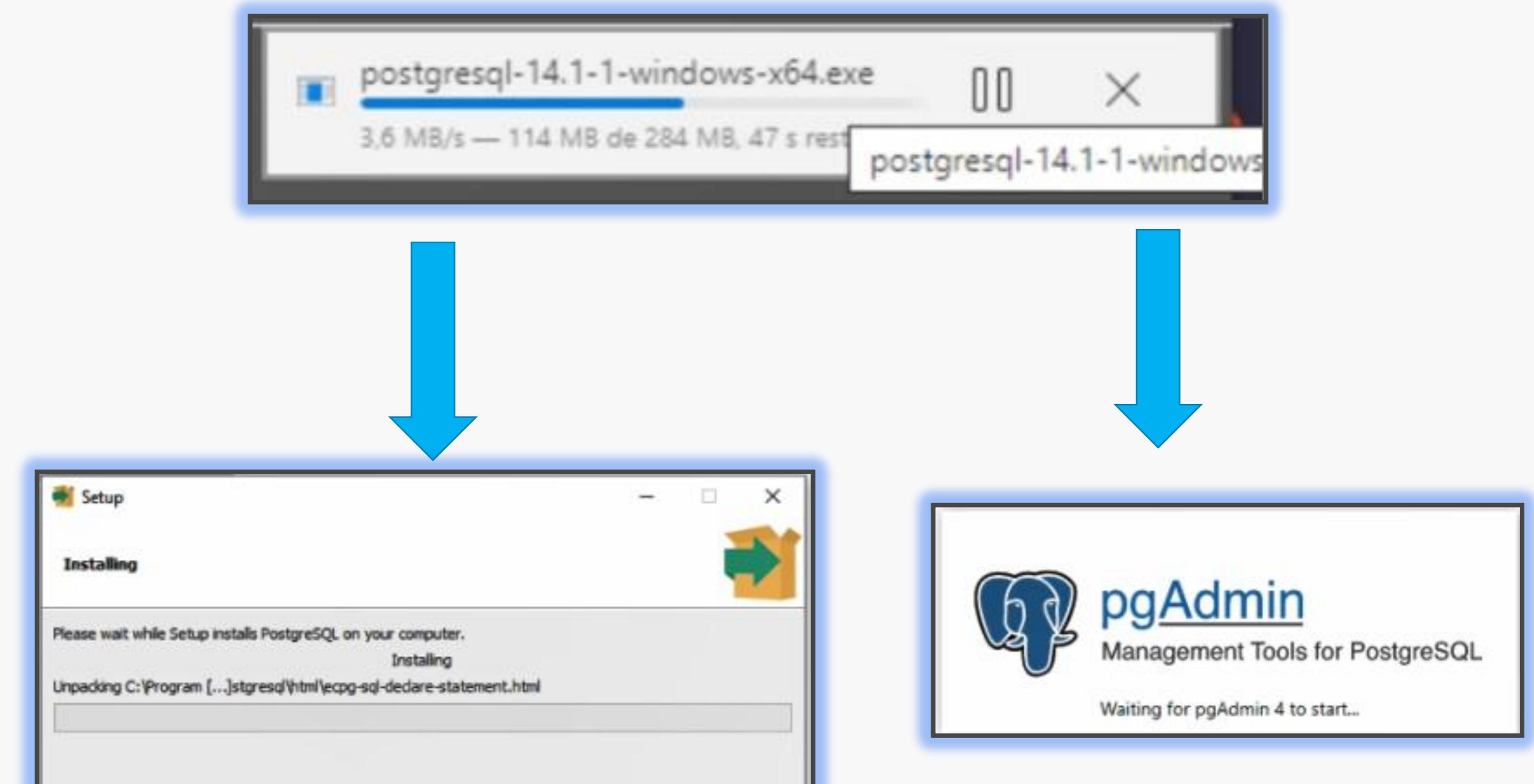
```
REM # Settings for all OSses
if "%HOP_OPTIONS%"=="" set HOP_OPTIONS=-Xmx2048m
```

Instalando Postgresql

Vamos instalar o banco de dados.

Instalação do Postgresql

Utilizaremos este banco de dados para leitura e gravação.



O QUE APRENDEREMOS

Entendimento sobre Business
Intelligence e Data Warehouse.

Entendimento sobre os conceitos

Precisamos entender como funciona um projeto de Business intelligence e de Data Warehouse.

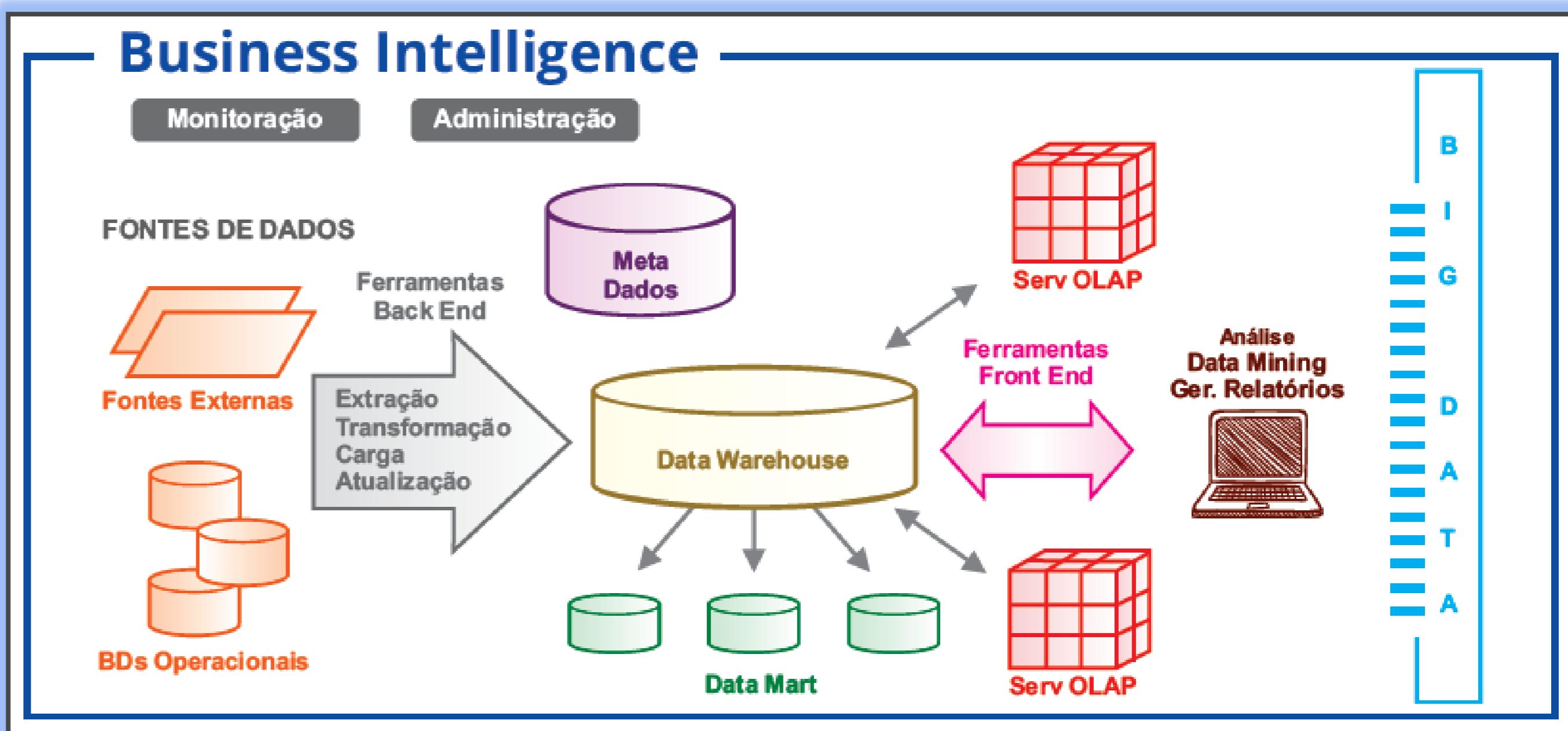
- O QUE É BUSINESS INTELLIGENCE
- O QUE É DATA WAREHOUSE
- MODELAGEM MULTIDIMENSIONAL
- CONSTRUÇÃO DO DW
 - CARGA DA STAGING DIMENSÃO
 - TRANSFORMAÇÃO DOS DADOS
 - CARGA TABELA STAGING FATO
 - CARGA DA DIMENSÃO
 - CARGA DA DIMENSÃO TEMPO
 - CARGA DA FATO

O QUE APRENDEREMOS

Entendimento sobre Business Intelligence.

Entendimento sobre Business Intelligence (BI)

Na tradução livre, significa Inteligência de negócios ou simplesmente “BI”. É um conjunto de técnicas que reunidas, possibilitam ao gestor a tomada de uma decisão com base em métricas ou valores que são extraídos de seus diversos sistemas, base de dados, Data Lake, dentre outros.



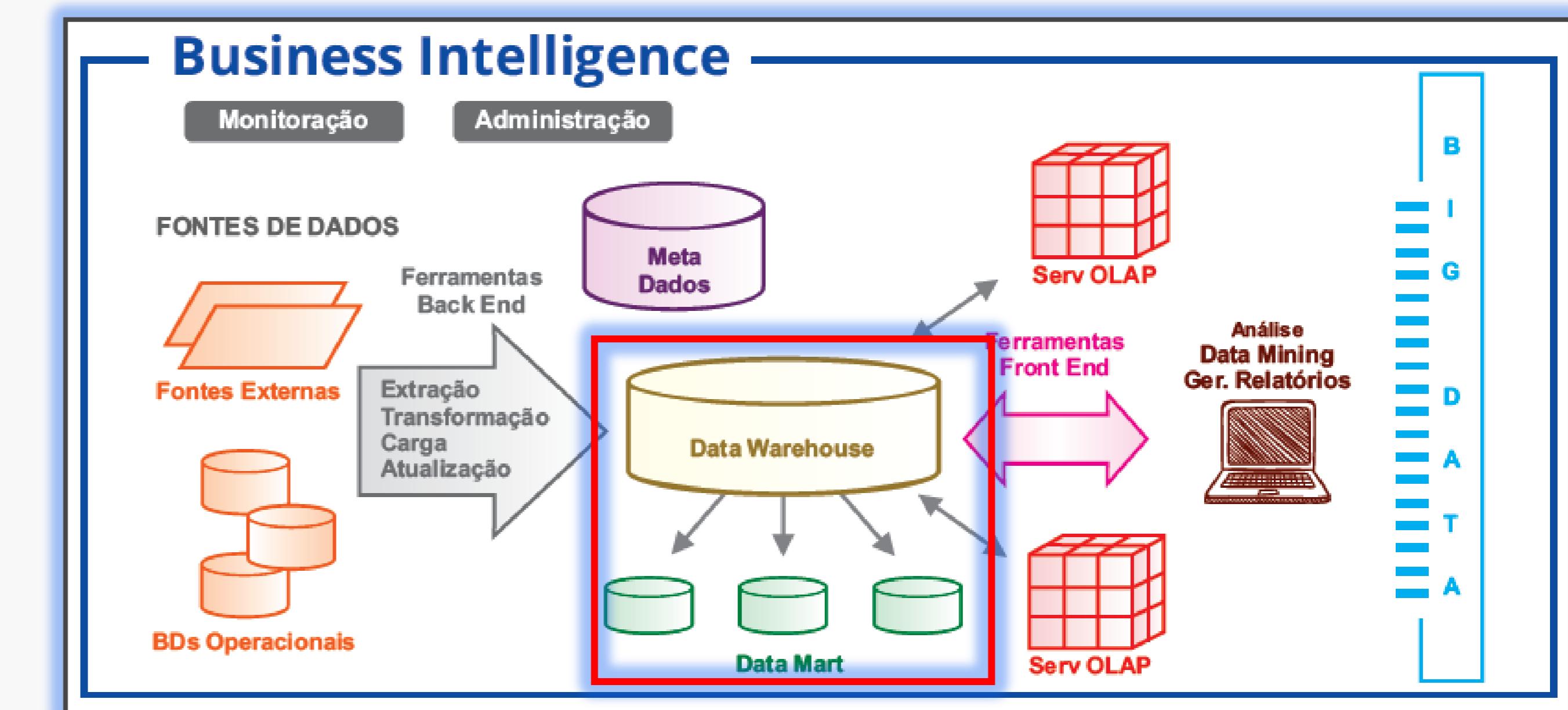
TUDO SOBRE BUSINESS INTELLIGENCE AQUI:
[<CLIQUE PARA ACESSAR DIRETAMENTE>](#)

O QUE APRENDEREMOS

Entendimento sobre Data Warehouse.

Entendimento sobre Data Warehouse (DW)

Também conhecido como “armazém de dados”, é um banco de dados que integra e consolida os diversos sistemas e fontes de dados (arquivos de texto, banco de dados, Data Lake, mensagens, dentre outras) de uma organização para apoio na tomada de decisão.



TUDO SOBRE DATA WAREHOUSE AQUI: [<CLIQUE PARA ACESSAR DIRETAMENTE>](#)

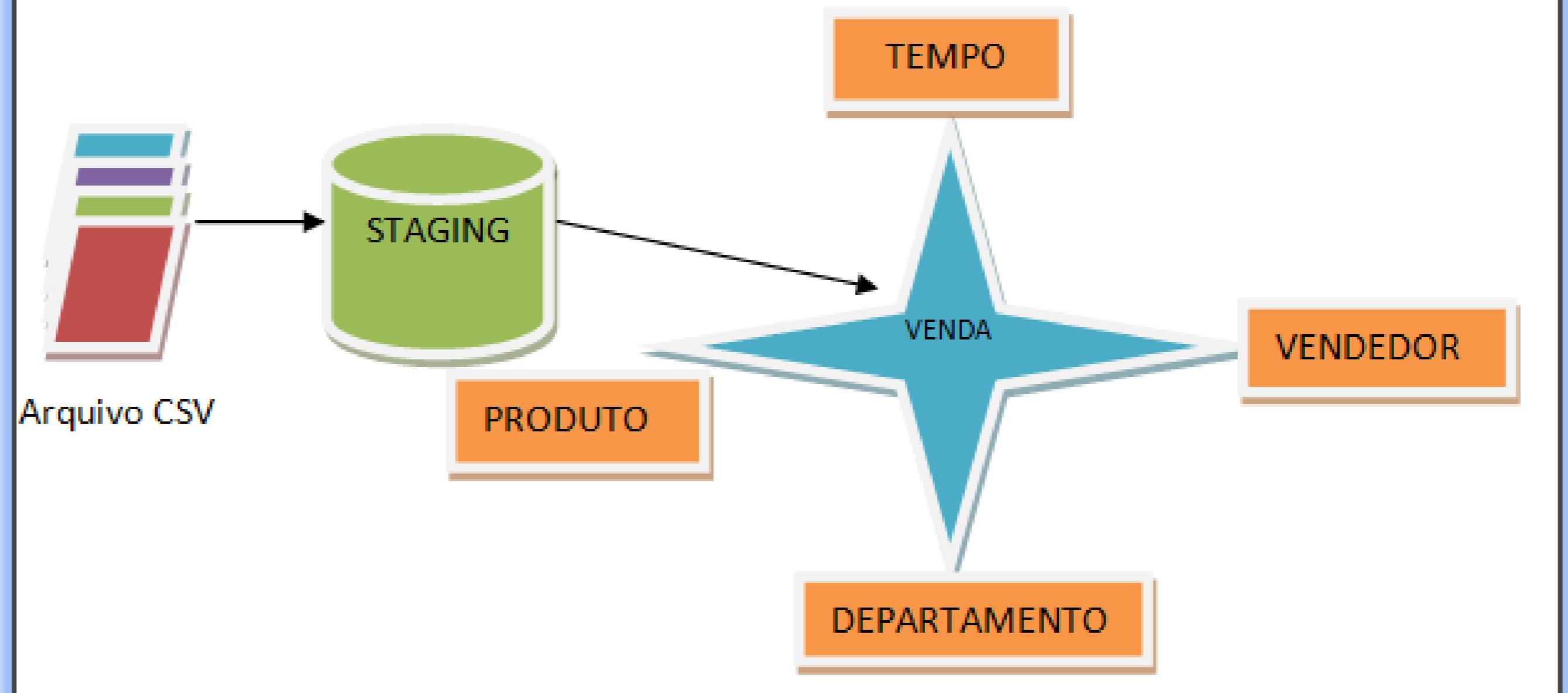
Construção do projeto de DW

Quais serão os dados que trabalharemos.

Construção sobre o projeto de Data Warehouse

Vejamos quais serão os dados a serem carregados e quais tabelas construiremos.

DATA WAREHOUSE DE VENDAS



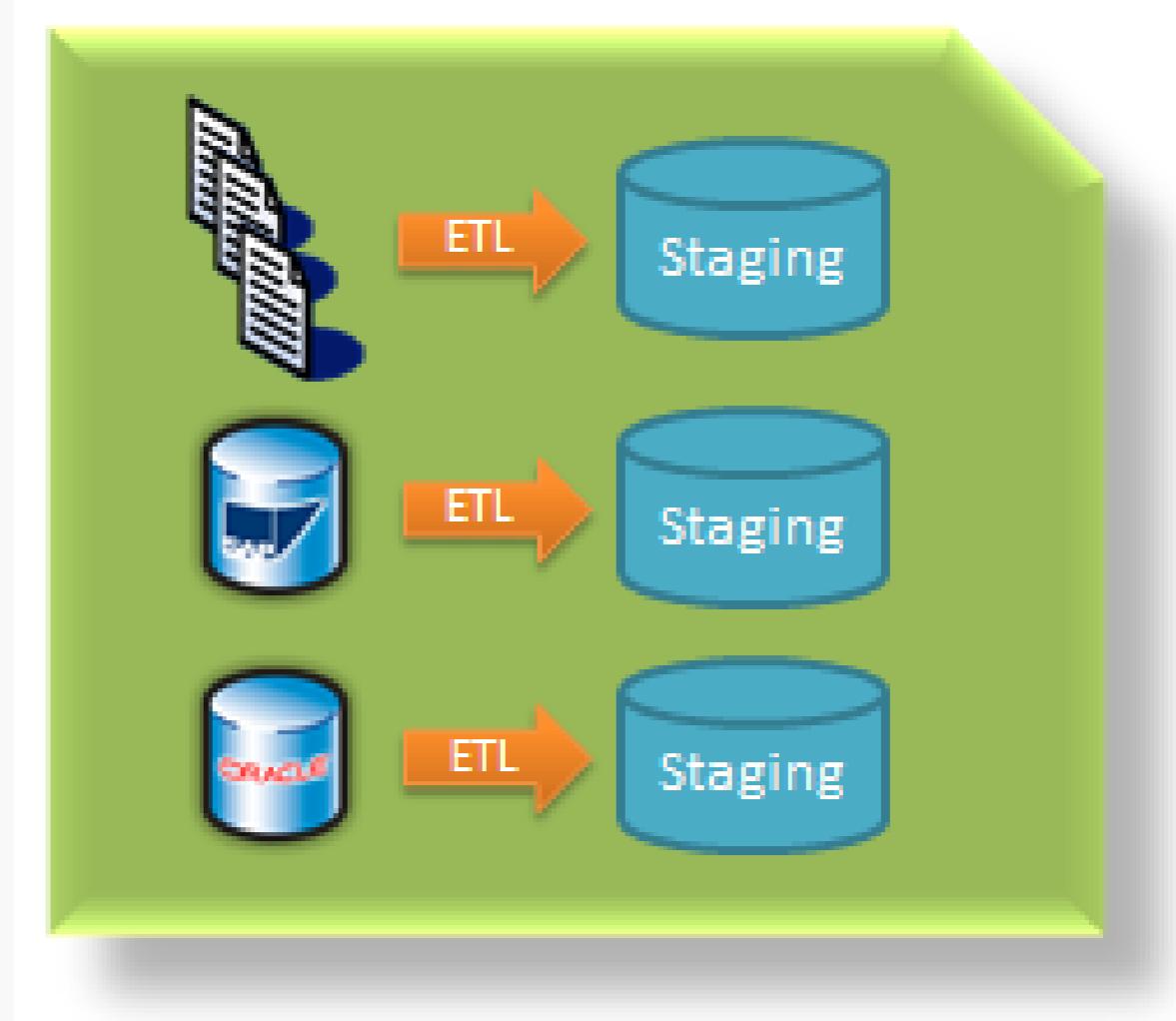
Entendendo sobre Staging Area

Local onde guardamos os dados lidos das fonte de dados (bases de dados, Data Lake, arquivos TXT, XLS, CSV, dentre outros.)

Construindo a Staging Area

Vejamos quais serão os dados a serem carregados e quais tabelas construiremos.

- Staging Area ou área auxiliar serve como ponto único para a carga efetiva no Data Warehouse.
- A cada carga seu conteúdo é limpo.
- Evita acesso à produção em caso de recarga durante o dia.



Entendendo sobre Dimensão e Fato

Local onde guardamos as métricas (Fato) e descritores (Dimensão).

Entendendo sobre Dimensão e Fato

Vejamos quais serão os dados a serem carregados e quais tabelas construiremos.

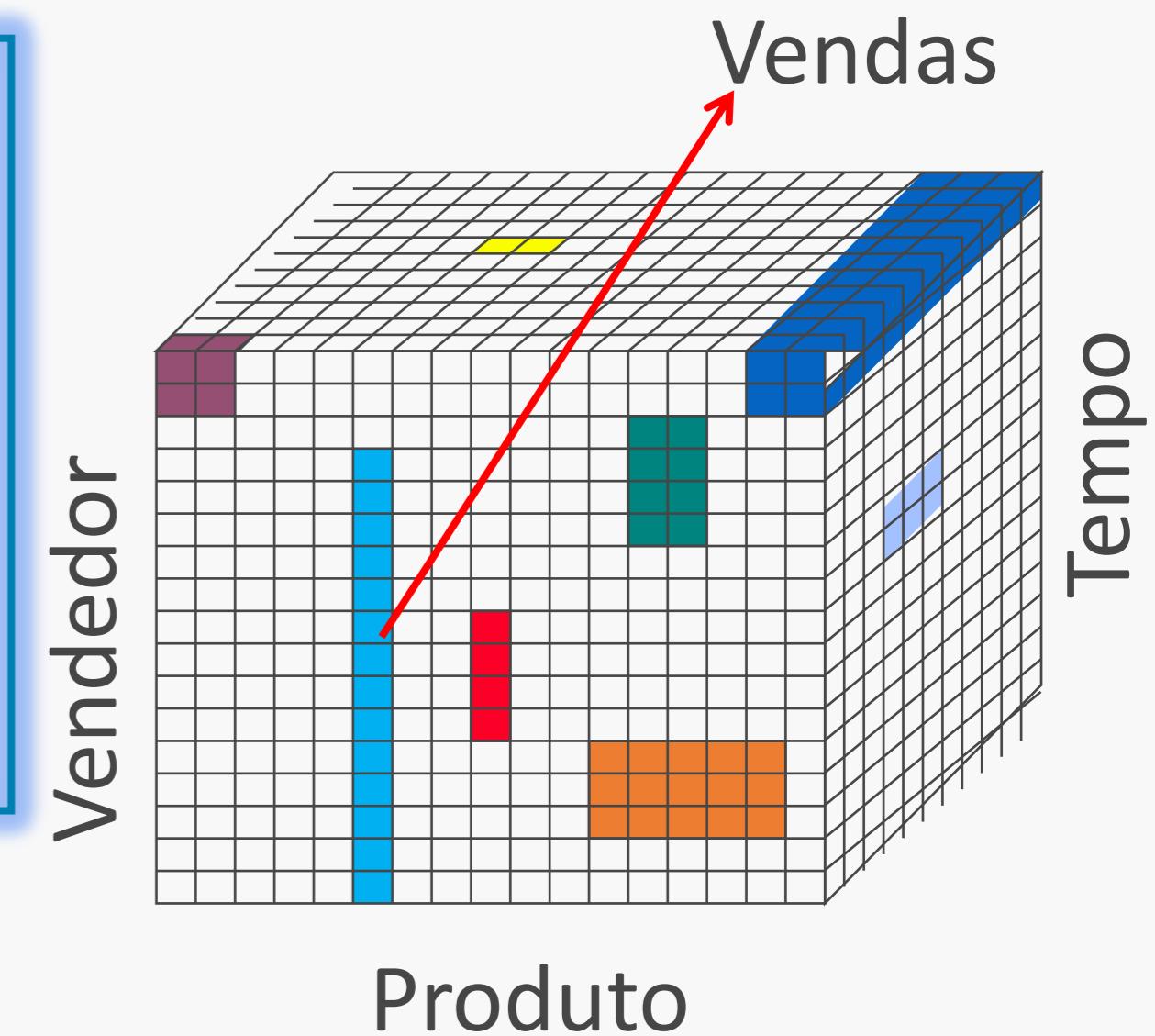
- **DIMENSÃO**
 1. Contém os descritores textuais do negócio.
Exemplo : Tempo , Cliente, Produto, Tipo de Embalagem, etc.
- **FATO**
 1. Termo utilizado para a medição do negócio.
Exemplo: quantidade de produtos vendidos, preço de compra, preço de venda, lucro, etc.

Dimensões

- Produto
- Vendedor
- Tempo (obrigatória)

Fato

- Vendas



Entendendo sobre Modelagem Multidimensional

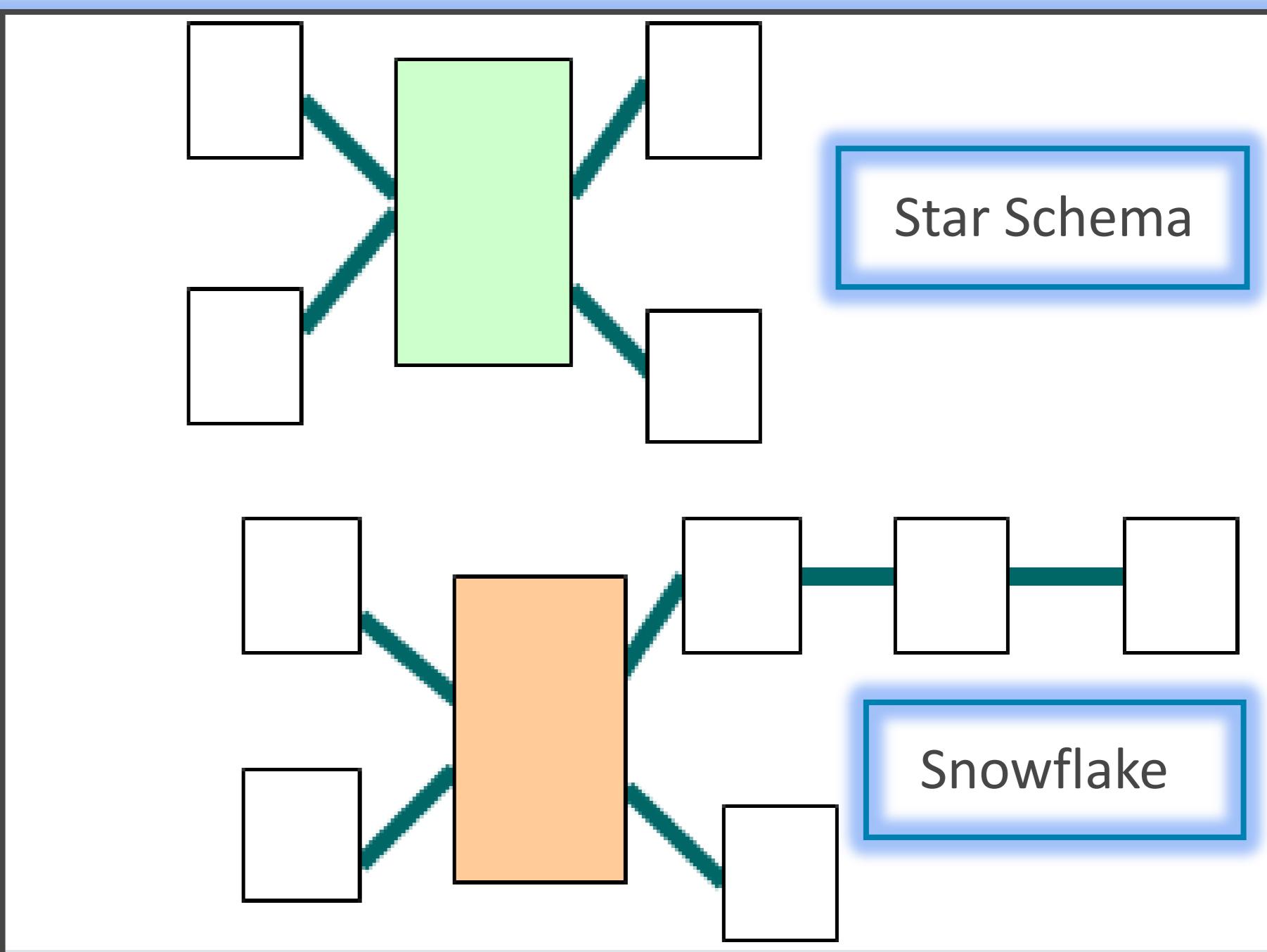
Entendendo os conceitos de STAR
SCHEMA e SNOWFLAKE

Entendendo sobre Modelagem Multidimensional

Vejamos quais serão os dados a serem carregados e quais tabelas construiremos.

Modelo Star Schema : dimensões desnormalizadas (alta performance, porém com requisitos de espaço de armazenamento em disco maior do que o modelo Snowflake).

Modelo Snowflake : dimensões normalizadas (performance menor porém com requisitos de espaço de armazenamento em disco menor do que o modelo Star Schema).



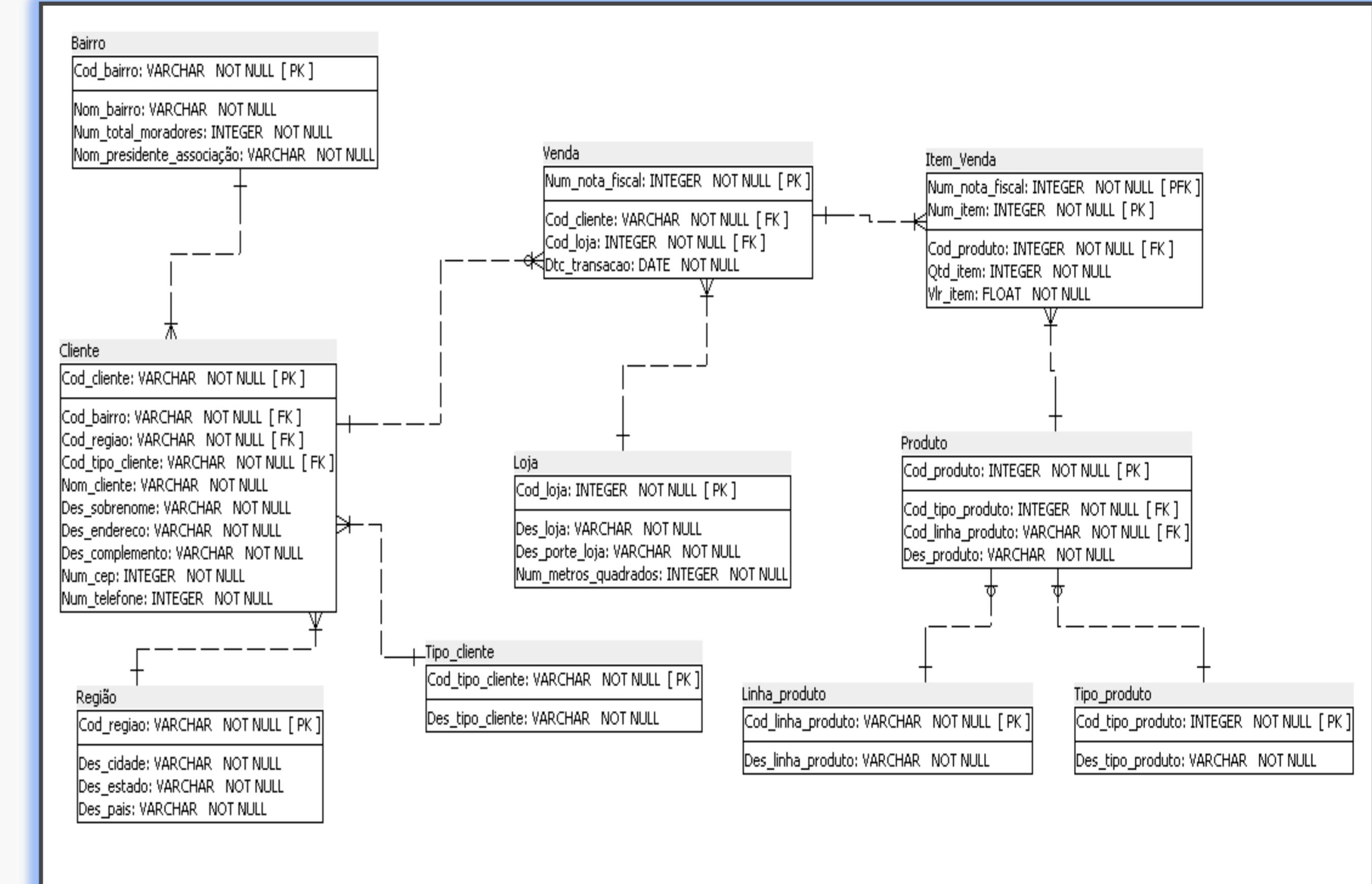
Entendendo sobre Modelagem Multidimensional

Entendendo os conceitos de STAR
SCHEMA e SNOWFLAKE

Entendendo sobre Modelagem Multidimensional

Vejamos um esquema normalizado de tabelas operacionais e vamos demonstrar quais seriam os possíveis modelos Star Schema e Snowflake.

Esquema normalizado Notas Fiscais

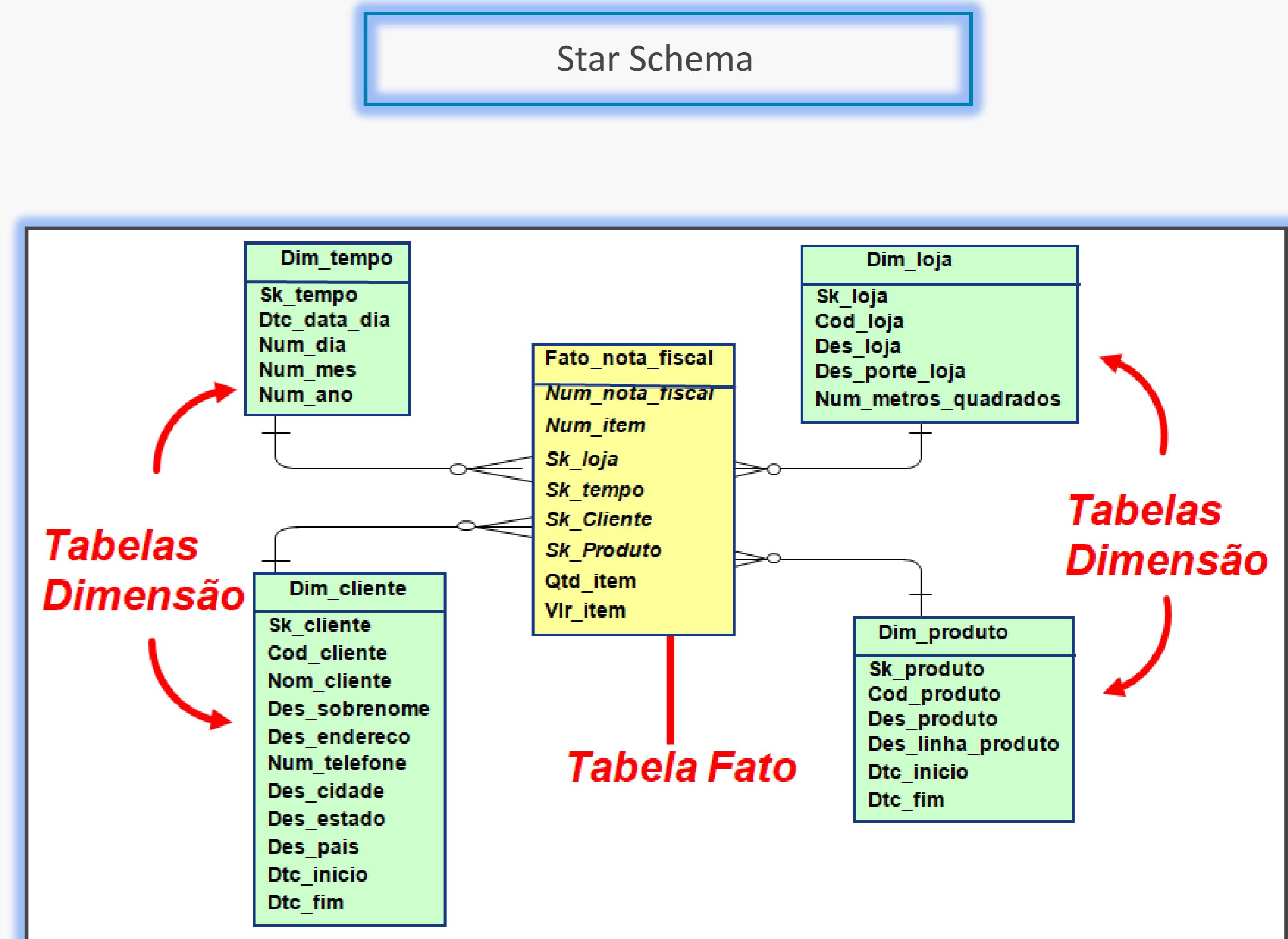


Entendendo sobre Modelagem Multidimensional

Entendendo os conceitos de STAR
SCHEMA e SNOWFLAKE

Entendendo sobre Modelagem Multidimensional

Vejamos como ficaria um modelo multidimensional na forma Star Schema.

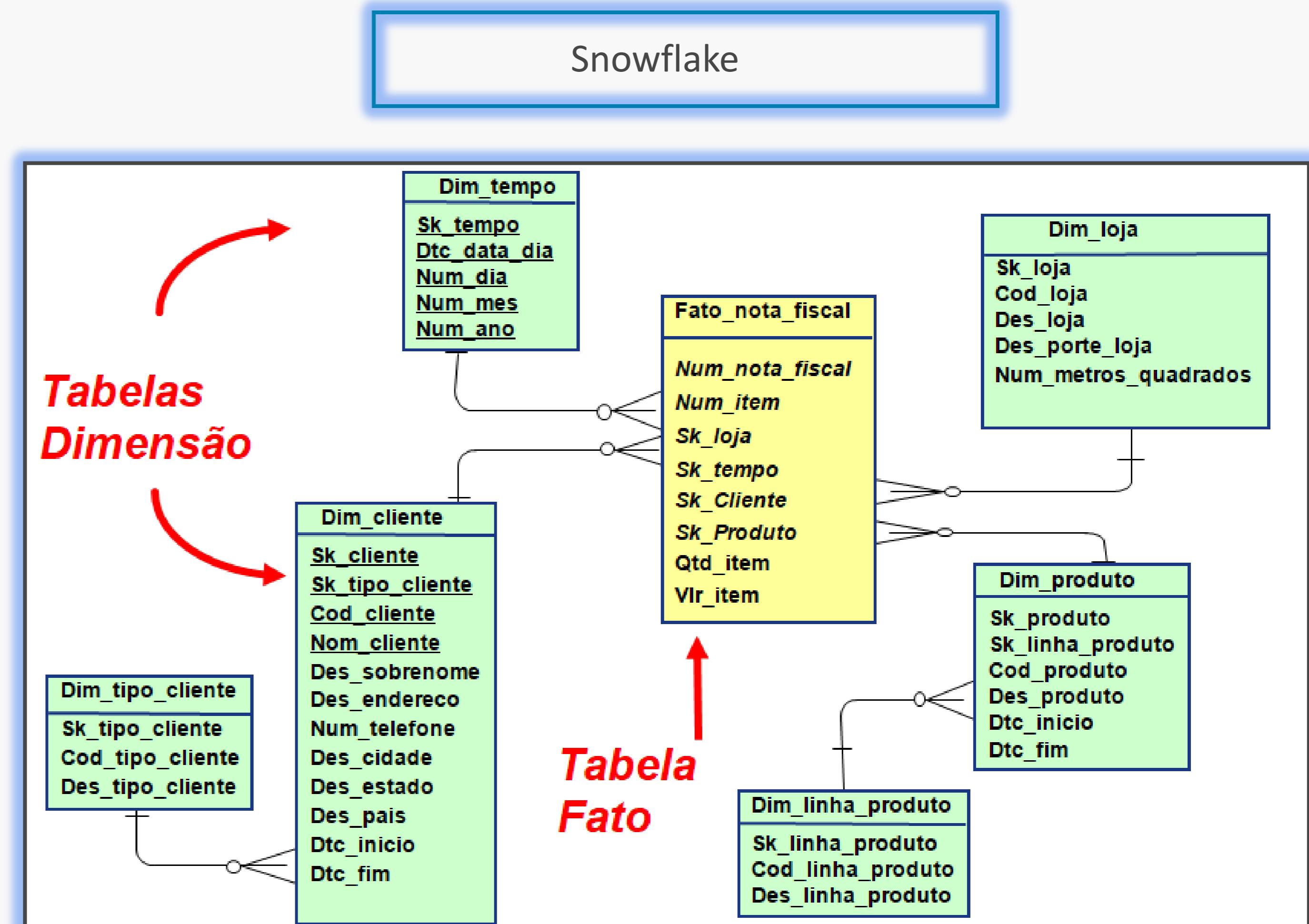


Entendendo sobre Modelagem Multidimensional

Entendendo os conceitos de STAR
SCHEMA e SNOWFLAKE

Entendendo sobre Modelagem Multidimensional

Vejamos como ficaria um modelo multidimensional na forma Snowflake.



Etapas de construção projeto Data Warehouse

Conhecendo as etapas de construção

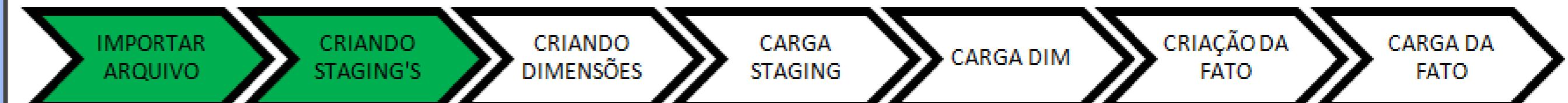
Etapas de construção projeto Data Warehouse

Confira o que construiremos para a carga do nosso Data Warehouse.

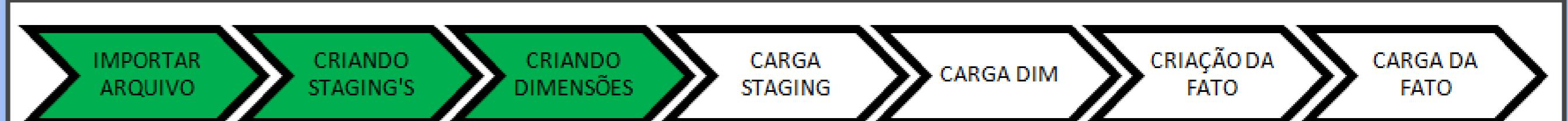
Importando os dados



Criando tabelas Staging Area



Criando tabelas Dimensões



Etapas de construção projeto Data Warehouse

Conhecendo as etapas de construção

Etapas de construção projeto Data Warehouse

Confira o que construiremos para a carga do nosso Data Warehouse.

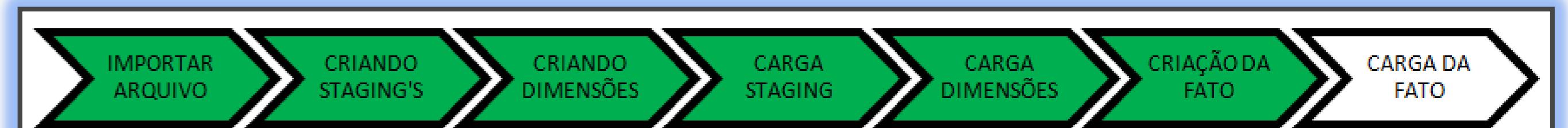
Criando carga Staging Area



Criando carga Dimensões



Criando tabela Fato



Etapas de construção projeto Data Warehouse

Conhecendo as etapas de construção

Etapas de construção projeto Data Warehouse

Confira o que construiremos para a carga do nosso Data Warehouse.

Criando carga Fato



Começando a trabalhar

Vamos carregar os primeiros dados.

Carregando dados

Entendendo o log, saídas diversas de análise.

```
7 - Hop - Pipeline opened.  
7 - Hop - Launching pipeline [estudo]...  
7 - Hop - Started the pipeline execution.  
7 - estudo - Executing this pipeline using the Local Pipeline Engine with run configuration 'local'  
7 - estudo - Expedindo início para Pipeline [estudo]  
7 - CSV file input.0 - Header row skipped in file 'C:\Users\grima\Downloads\Hop\Dados\vinhos_mundo.csv'  
3 - CSV file input.0 - Finished processing (I=129972, O=0, R=0, W=129971, U=0, E=0)  
3 - Value mapper.0 - Finished processing (I=0, O=0, R=129971, W=129971, U=0, E=0)  
3 - estudo - Pipeline duration : 0.805 seconds [ 0.805" ]  
3 - estudo - Execution finished on a local pipeline engine with run configuration 'local'
```

#	Nome do transform	Copia nr	Input	Read	Written	Output	Updated	Rejected	Errors	Buffers Input	Buffers Output	Duration	Speed	Status
1	CSV file input	0	129.972	0	129.971	0	0	0	0	0	0	0.738"	173.527	Finished
2	Value mapper	0	0	129.971	129.971	0	0	0	0	0	0	0.755"	169.897	Finished

Preview

View output Preview output Debug output Sniff output Add data probe

Basic

Edit Copy to clipboard Create hop Detach transform Show input fields Show output fields
Edit description Delete

Entendo os dados

Vemos entender quais serão os dados a serem carregados.

Preparando os dados

Vamos conhecer os dados que serão utilizados para o nosso projeto.

Venda.xlsx

id_venda	cod_vendedor	cod_produto	cod_departamento	dtc_venda	qtd_venda	val_venda	num nota
1	3	4		1 22/08/2015	1	1300	1033
2	3	10		1 15/08/2014	1	2,15	812
3	5	11		1 26/04/2014	2	0,5	447
4	9	19		1 09/01/2014	1	1500	509
5	10	16		6 24/05/2015	1	135	1337
6	4	11		3 25/07/2014	1	0,5	1337
7	4	9		5 12/03/2015	1	12,25	724
8	7	19		4 17/08/2014	2	1500	375
9	9	20		3 07/02/2015	2	350	416
10	9	15		3 07/10/2014	1	98,5	1085
11	3	16		5 02/07/2014	1	135	692
12	2	18		1 02/04/2014	3	850	74
13	4	9		3 03/10/2015	3	12,25	797
14	1	18		5 12/03/2015	3	850	391
15	12	8		2 13/03/2014	1	875	700
16	8	15		3 15/02/2014	3	98,5	241
17	9	21		4 04/03/2014	3	2100	403
18	1	8		4 15/10/2014	2	875	1427
19	1	7		6 09/11/2014	1	52,9	210
20	9	14		4 01/03/2014	2	25	849

Vendedor.xlsx

cod_vendedor	des_vendedor
1	Claudio Silva
2	Márcio Barroso
3	Issac Palmeiras
4	Daniela Matos
5	Maria Braga
6	Rodrigo Machado
7	João Oliveira
8	Márcia Brito
9	Diego Santos
10	Danilo Barreto
11	Filipe Almeida
12	Marilia Cardoso

Produto.xlsx

cod_produto	nom_produto
1	TV
2	Mouse
3	Teclado
4	Monitor
5	Computador
6	Notebook
7	Celular 4Gb
8	Telefone Sem Fio
9	Celular 8Gb
10	Papel A4
11	Caneta
12	Borracha
13	Estojo
14	Classificador
15	Baton
16	Perfume
17	Aparalho de Barbear Elétrico
18	Xbox
19	Fogão
20	Geladeira
21	microondas
22	Mesa
23	Sofá

Departamento.xlsx

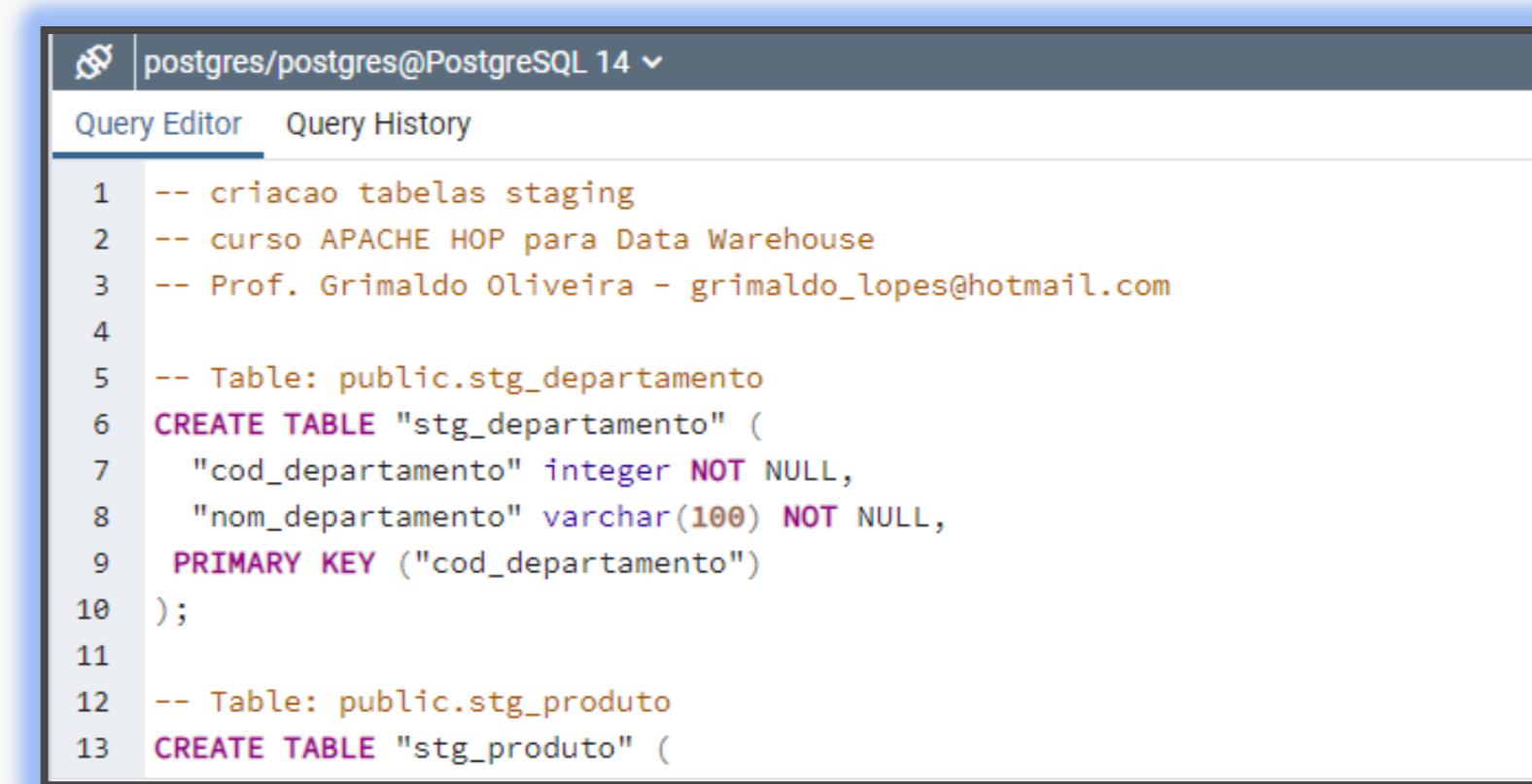
cod_departamento	des_departamento
1	Informática
2	Telefônia
3	Papelaria
4	cosmético
5	Eletrônico
6	Eletrodoméstico
7	Móveis

Começando a trabalhar

Vamos criar as tabelas Staging no Postgres.

Executando os scripts das tabelas Staging

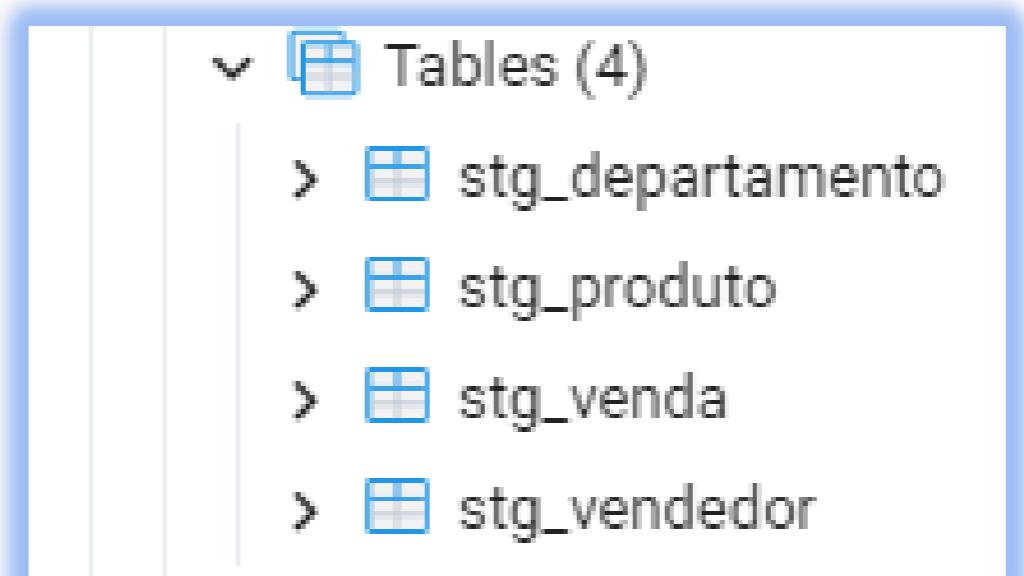
Vamos criar as tabelas no banco de dados postgres



```
-- criacao tabelas staging
-- curso APACHE HOP para Data Warehouse
-- Prof. Grimaldo Oliveira - grimaldo_lopes@hotmail.com

-- Table: public.stg_departamento
CREATE TABLE "stg_departamento" (
    "cod_departamento" integer NOT NULL,
    "nom_departamento" varchar(100) NOT NULL,
    PRIMARY KEY ("cod_departamento")
);

-- Table: public.stg_produto
CREATE TABLE "stg_produto" (
```

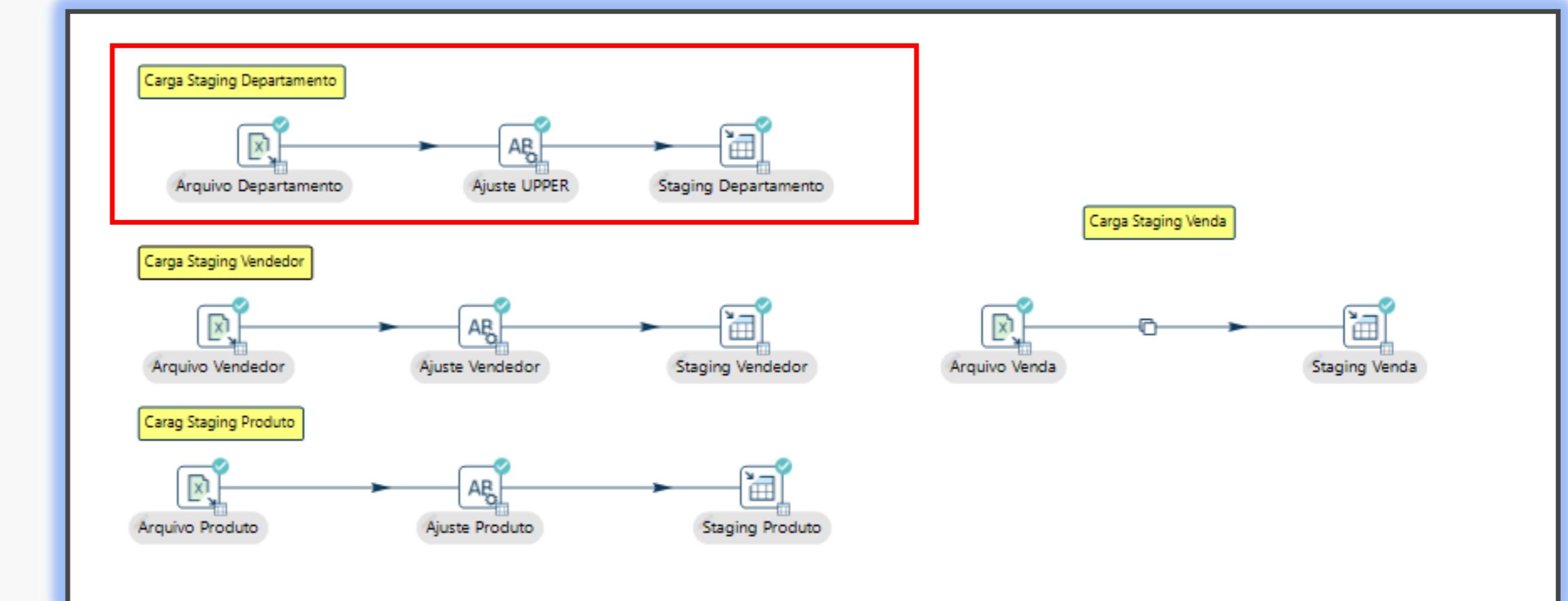


Construindo as cargas Staging

A carga Staging das tabelas.

Construção do pipeline das tabelas Staging

Vamos criar a carga que carrega os arquivos: **departamento**, vendedor, produto e venda.



The screenshot shows the Microsoft Data Flow interface with three main components:

- Microsoft Excel input:** Set to "Nome do Transform" "Arquivo Departamento". It maps two fields:

#	Name	Type	Length	Precision	Trim type	Repeat	Format	Current
1	cod_departamento	Number			none	N	0	
2	des_departamento	String			none	N		
3								
- String operations:** Set to "Transform name" "Ajuste UPPER". It processes one field:

#	In stream field	Out stream field	Trim type	Lower/Upper	Padding	Pad char
1	des_departamento		none	upper	none	
- Saída a Tabela:** Set to "Nome do Transform" "Staging Departamento". It connects to a "Datawarehouse" connection and specifies the target schema and table: "public.stg_departamento".

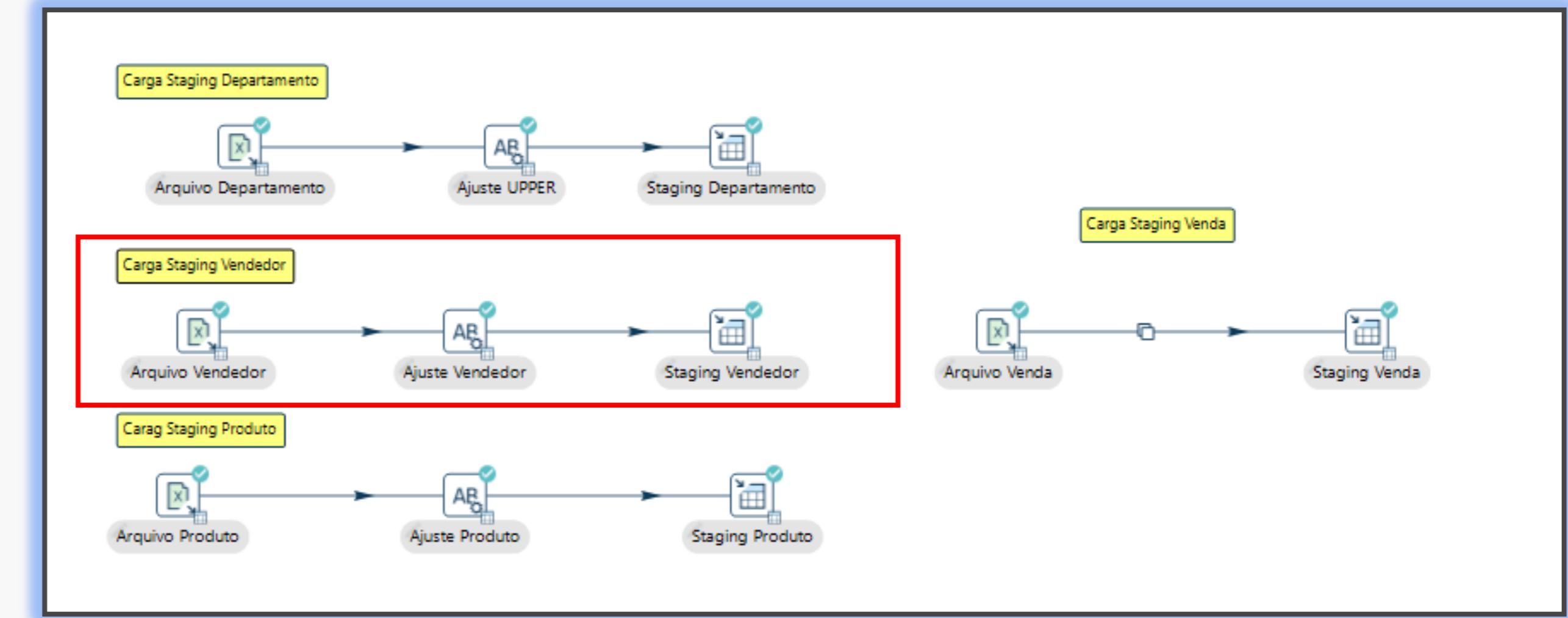
Nome do Transform	Connection	Target schema	Target table	Commit size	Truncate table	Ignore insert errors	Specify database fields
Staging Departamento	Datawarehouse		"public".stg_departamento	1000	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Construindo as cargas Staging

A carga Staging das tabelas.

Construção do pipeline das tabelas Staging

Vamos criar a carga que carrega os arquivos: departamento, **vendedor**, produto e venda.

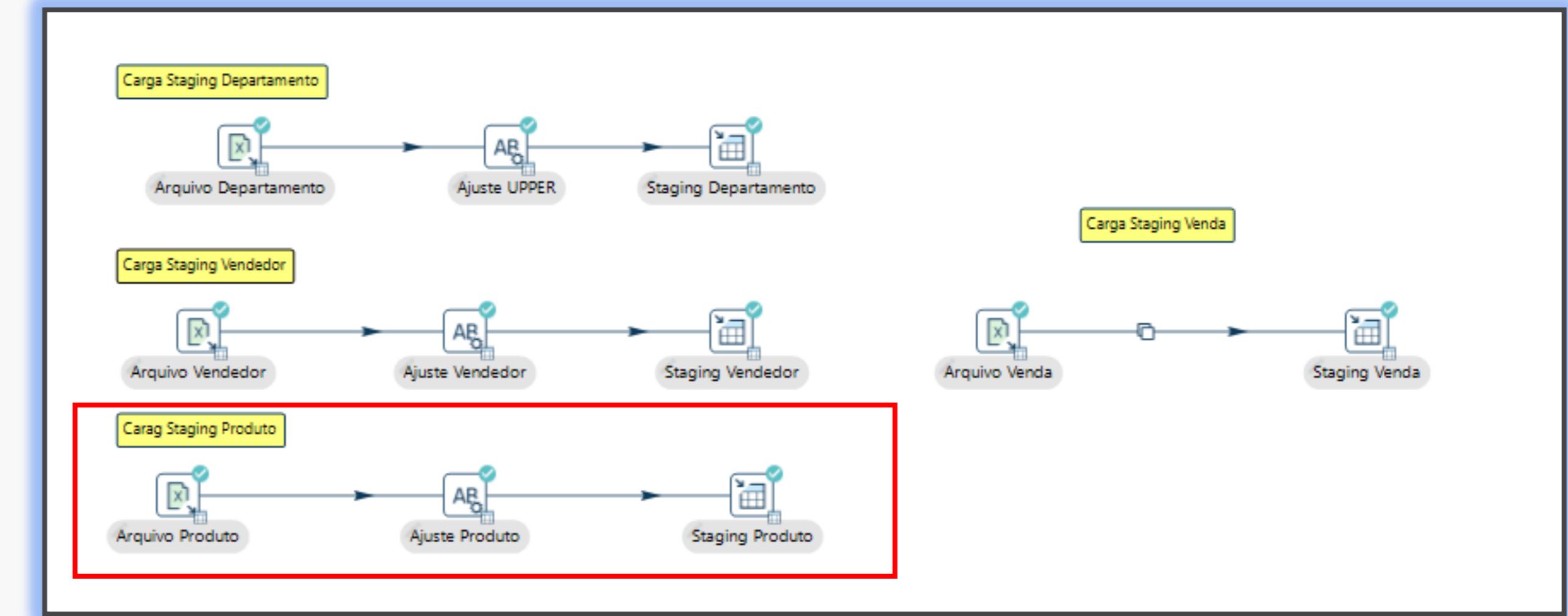


Construindo as cargas Staging

A carga Staging das tabelas.

Construção do pipeline das tabelas Staging

Vamos criar a carga que carrega os arquivos: departamento, vendedor, **produto** e venda.

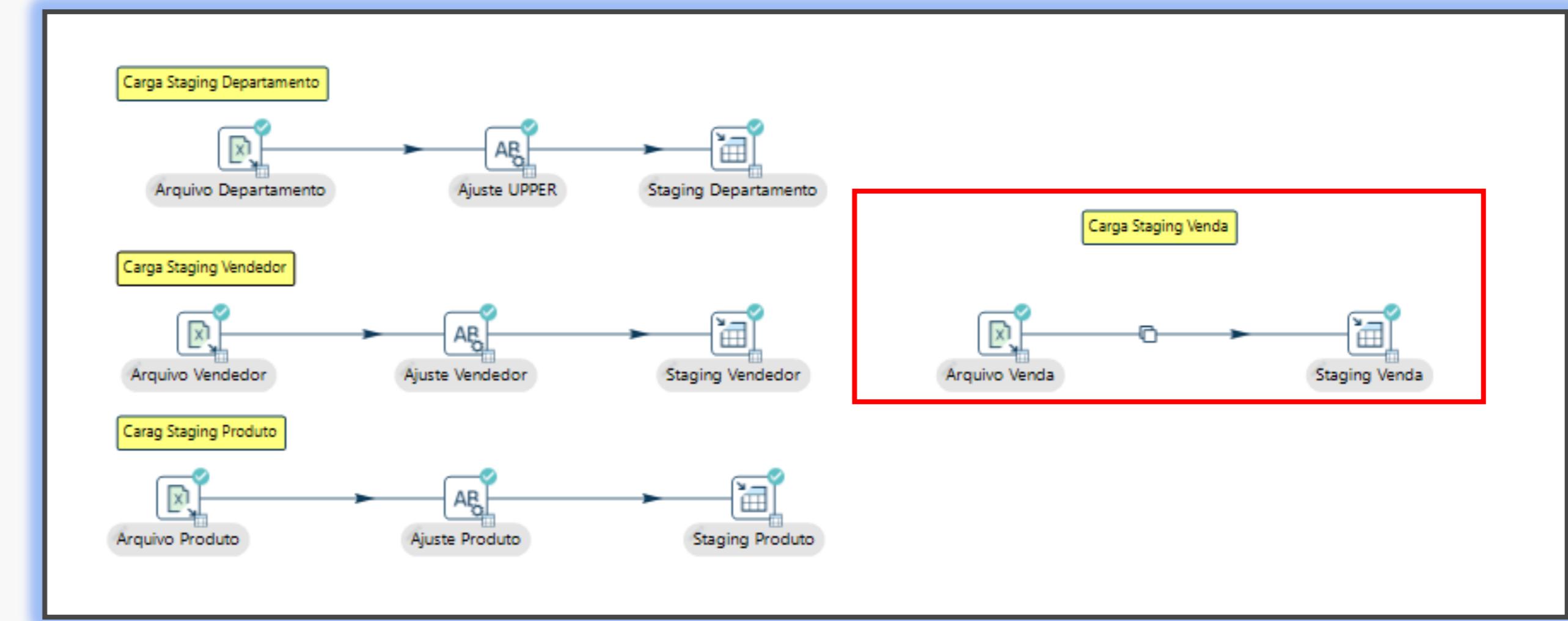


Construindo as cargas Staging

A carga Staging das tabelas.

Construção do pipeline das tabelas Staging

Vamos criar a carga que carrega os arquivos: departamento, vendedor, produto e **venda**.



Etapas de construção projeto Data Warehouse

Conhecendo as etapas de construção

Etapas de construção projeto Data Warehouse

Confira o que construiremos para a carga do nosso Data Warehouse.

Criando carga Dimensões



Criando tabela Fato



Construindo as cargas Dimensão

Entenda como funciona uma carga de
dimensão.

Entenda como funciona uma carga de dimensão

Vamos compreender os tipos de dimensão.

Tipo 1 : Sobrescrever os Dados

- O novo registro substitui o registro original. Só existe um registro no banco de dados - os dados atuais.
- Não mantém histórico

Construindo as cargas Dimensão

Entenda como funciona uma carga de dimensão.

Entenda como funciona uma carga de dimensão

Vamos compreender os tipos de dimensão.

Supplier_key	Supplier_Name	Supplier_State
001	Phlogistical Sociedade de Abastecimento	CA

Supplier_key	Supplier_Name	Supplier_State
001	Phlogistical Sociedade de Abastecimento	IL

Construindo as cargas Dimensão

Entenda como funciona uma carga de dimensão.

Entenda como funciona uma carga de dimensão

Vamos compreender os tipos de dimensão.

Tipo 2 : Mantém o histórico dos dados

- Mantém histórico

Um novo registro é adicionado na tabela de dimensão. Dois registros existentes no banco de Dados.

- os dados atuais e dados da história anterior
- É recomendável para 99% dos casos**

Construindo as cargas Dimensão

Entenda como funciona uma carga de dimensão.

Entenda como funciona uma carga de dimensão

Vamos compreender os tipos de dimensão.

Supplier_key	Supplier_Code	Supplier_Name	Supplier_State	Data_inicial	Data_final
001	ABC	Phlogistical Sociedade de Abastecimento	CA	01 de janeiro-2000	21-Dez-2004
002	ABC	Phlogistical Sociedade de Abastecimento	IL	22-Dez-2004	

Construindo as cargas Dimensão

Entenda como funciona uma carga de dimensão.

Vamos criar as tabelas de dimensão no Postgresql

Vamos criar as tabelas de dimensão antes de iniciarmos o pipeline no Apache HOP.

```
CREATE TABLE "dim_departamento" (
    "sk_departamento" integer NOT NULL,
    "cod_departamento" integer NULL,
    "nom_departamento" varchar(100)NULL,
    "dtc_inicio" date,
    "dtc_fim" date,
    "versao" integer,
    CONSTRAINT sk_departamento_pkey PRIMARY KEY (sk_departamento)
);

CREATE TABLE "dim_vendedor" (
    "sk_vendedor" integer NOT NULL,
    "cod_vendedor" integer NULL,
    "nom_vendedor" varchar(255)NULL,
    "dtc_inicio" date,
    "dtc_fim" date,
    "versao" integer,
    CONSTRAINT sk_vendedor_pkey PRIMARY KEY (sk_vendedor)
);

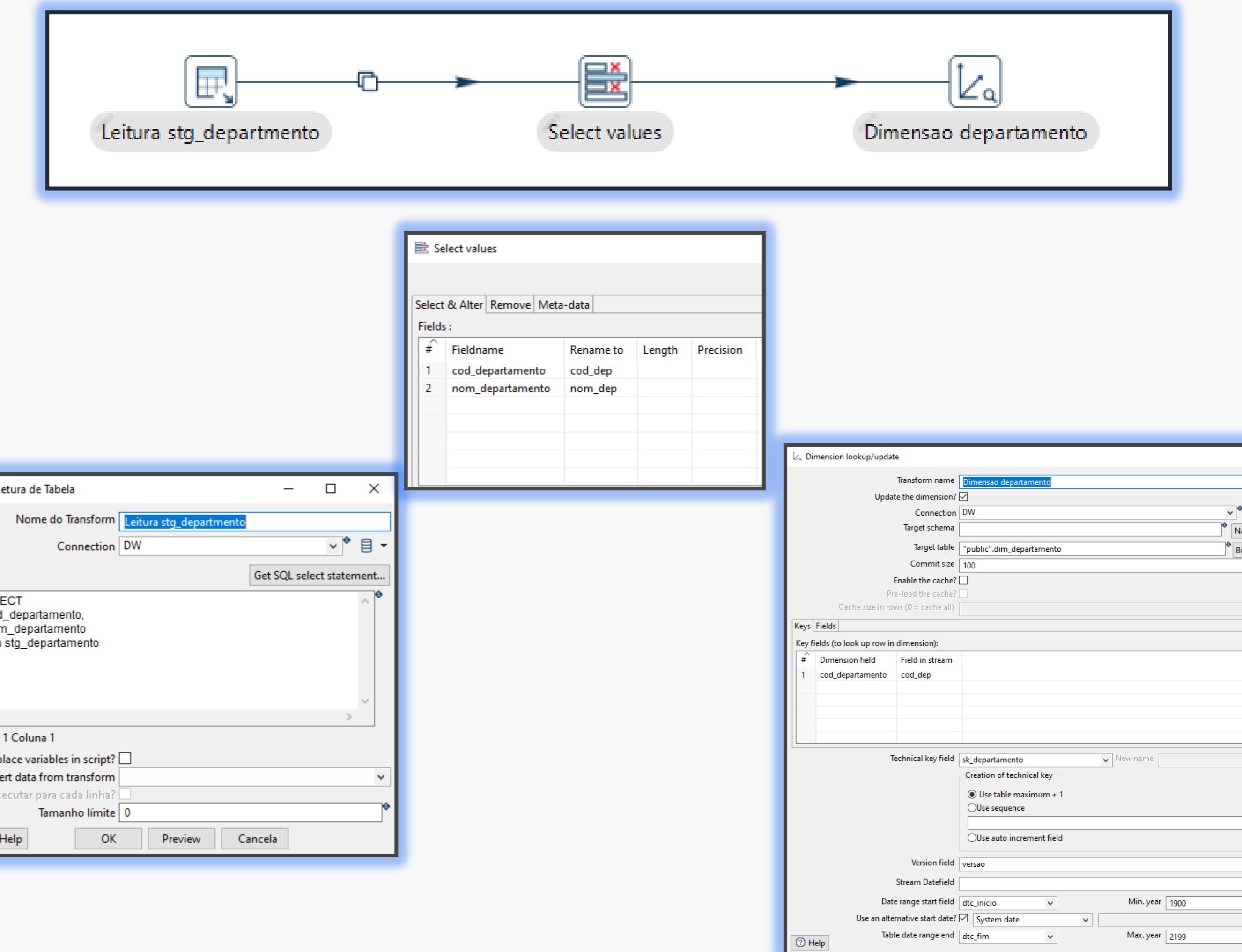
CREATE TABLE "dim_produto" (
    "sk_produto" integer NOT NULL,
    "cod_produto" integer NULL,
    "nom_produto" varchar(255)NULL,
    "dtc_inicio" date,
    "dtc_fim" date,
    "versao" integer,
    CONSTRAINT sk_produto_pkey PRIMARY KEY (sk_produto)
);
```

Construindo as cargas Dimensão

Entenda como funciona uma carga de dimensão.

Construção do pipeline da dimensão departamento

Vamos criar a carga que carrega os dados para a dimensão departamento.

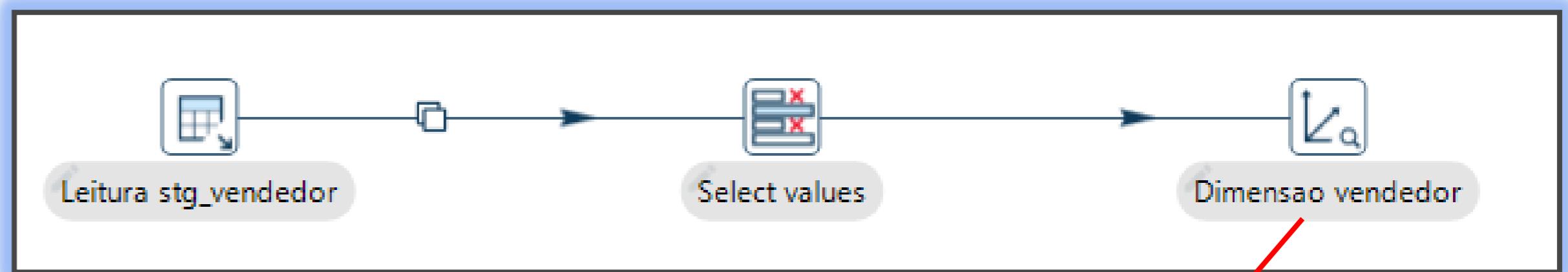


Construindo as cargas Dimensão

Entenda como funciona uma carga de dimensão.

Construção do pipeline da dimensão vendedor

Vamos criar a carga que carrega os dados para a dimensão vendedor.



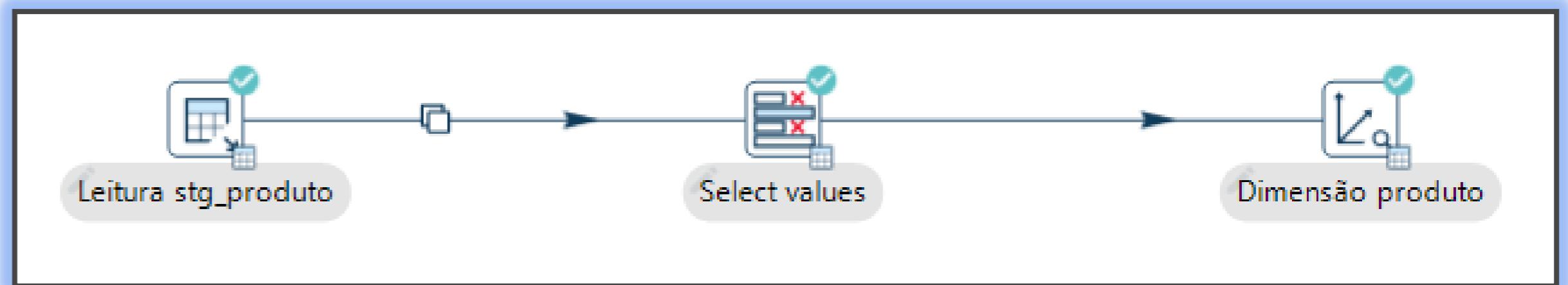
Keys		Fields
Lookup/Update fields		
#	Dimension field	Stream field to compare with
1	nom_vendedor	nom_vend
		Type of dimension update
		Update

Construindo as cargas Dimensão

Entenda como funciona uma carga de dimensão.

Construção do pipeline da dimensão produto

Vamos criar a carga que carrega os dados para a dimensão produto.



Como tratar informações não existentes

Por prática em projetos, é importante
criar um registro com sk=-1.

Informações inexistentes

Quando a carga da tabela fato for executada, caso exista algum código ligado a dimensão que não exista, gravaremos o **valor = -1**

```
INSERT INTO dim_departamento(  
    sk_departamento, cod_departamento, nom_departamento, dtc_inicio, dtc_fim, versao)  
VALUES (-1, -1, 'SEM INFORMACAO', '1900-01-01', NULL, 1);  
  
INSERT INTO dim_vendedor(  
    sk_vendedor, cod_vendedor, nom_vendedor, dtc_inicio, dtc_fim, versao)  
VALUES (-1, -1, 'SEM INFORMACAO', '1900-01-01', NULL, 1);  
  
INSERT INTO dim_produto (  
    sk_produto, cod_produto, nom_produto, dtc_inicio, dtc_fim, versao)  
VALUES (-1, -1, 'SEM INFORMACAO', '1900-01-01', NULL, 1);
```

Criação da Dimensão Tempo

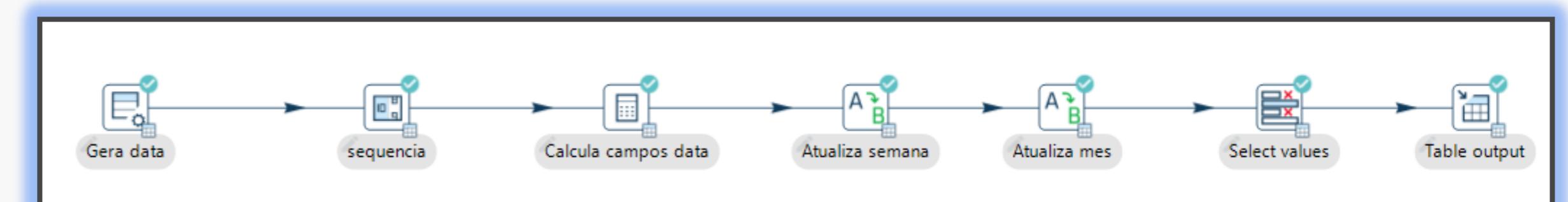
Criaremos a dimensão tempo e demonstraremos a sua carga.

Dimensão tempo

Todo projeto e DW tem uma dimensão tempo é obrigatório

```
CREATE TABLE "dim_tempo" (
    "ano" integer NOT NULL,
    "data" date NOT NULL,
    "dia_mes" integer NOT NULL,
    "dia_semana_desc" character varying(7),
    "mes" integer,
    "mes_desc" character varying(9),
    "sk_tempo" integer,
    CONSTRAINT sk_tempo_pkey PRIMARY KEY (sk_tempo)
);
```

	ano integer	data date	dia_mes integer	dia_semana_desc character varying (7)	mes integer	mes_desc character varying (9)	sk_tempo [PK] integer
1	2010	2010-01-01	1	Sexta	1	Janeiro	0
2	2010	2010-01-02	2	Sábado	1	Janeiro	1
3	2010	2010-01-03	3	Domingo	1	Janeiro	2
4	2010	2010-01-04	4	Segunda	1	Janeiro	3
5	2010	2010-01-05	5	Terça	1	Janeiro	4
6	2010	2010-01-06	6	Quarta	1	Janeiro	5
7	2010	2010-01-07	7	Quinta	1	Janeiro	6



Criação da Dimensão Tempo

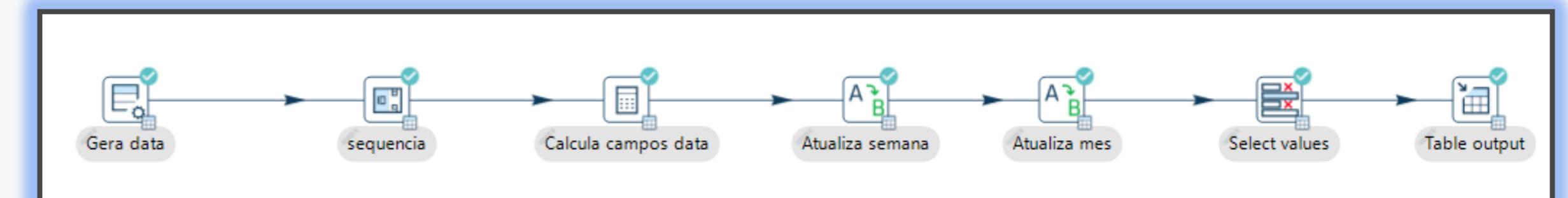
Criaremos a dimensão tempo e demonstraremos a sua carga.

Dimensão tempo

Todo projeto e DW tem uma dimensão tempo é obrigatório

```
CREATE TABLE "dim_tempo" (
    "ano" integer NOT NULL,
    "data" date NOT NULL,
    "dia_mes" integer NOT NULL,
    "dia_semana_desc" character varying(7),
    "mes" integer,
    "mes_desc" character varying(9),
    "sk_tempo" integer,
    CONSTRAINT sk_tempo_pkey PRIMARY KEY (sk_tempo)
);
```

	ano integer	data date	dia_mes integer	dia_semana_desc character varying (7)	mes integer	mes_desc character varying (9)	sk_tempo [PK] integer
1	2010	2010-01-01	1	Sexta	1	Janeiro	0
2	2010	2010-01-02	2	Sábado	1	Janeiro	1
3	2010	2010-01-03	3	Domingo	1	Janeiro	2
4	2010	2010-01-04	4	Segunda	1	Janeiro	3
5	2010	2010-01-05	5	Terça	1	Janeiro	4
6	2010	2010-01-06	6	Quarta	1	Janeiro	5
7	2010	2010-01-07	7	Quinta	1	Janeiro	6



Etapas de construção projeto Data Warehouse

Conhecendo as etapas de construção

Etapas de construção projeto Data Warehouse

Confira o que construiremos para a carga do nosso Data Warehouse.

Criando tabela Fato



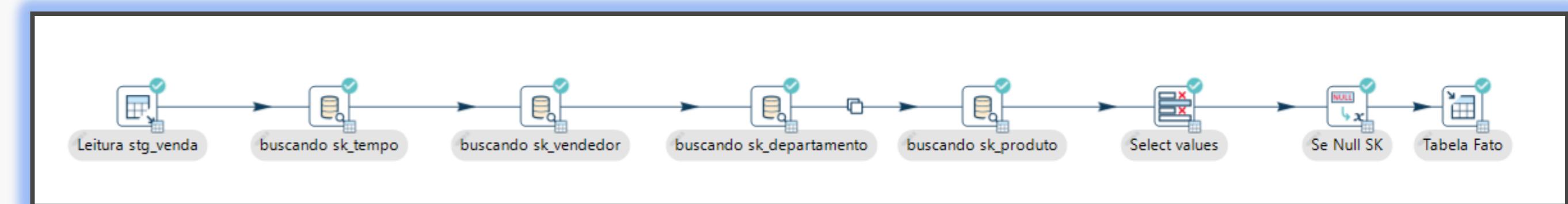
Criação da tabela e carga Fato

Criaremos a tabela fato venda e demonstraremos a sua carga.

Carga Fato

Faremos a construção da tabela e carga da fato, uma das mais importantes no DW.

```
CREATE TABLE fat_venda (
    "id_venda" integer NOT NULL,
    "num_nota" integer NOT NULL,
    "sk_tempo" integer NULL,
    "sk_produto" integer NULL,
    "sk_vendedor" integer NULL,
    "sk_departamento" integer NULL,
    "qtd_venda" integer DEFAULT NULL,
    "val_venda" decimal(10,2) DEFAULT NULL,
    CONSTRAINT id_venda_pkey PRIMARY KEY (id_venda)
)
```



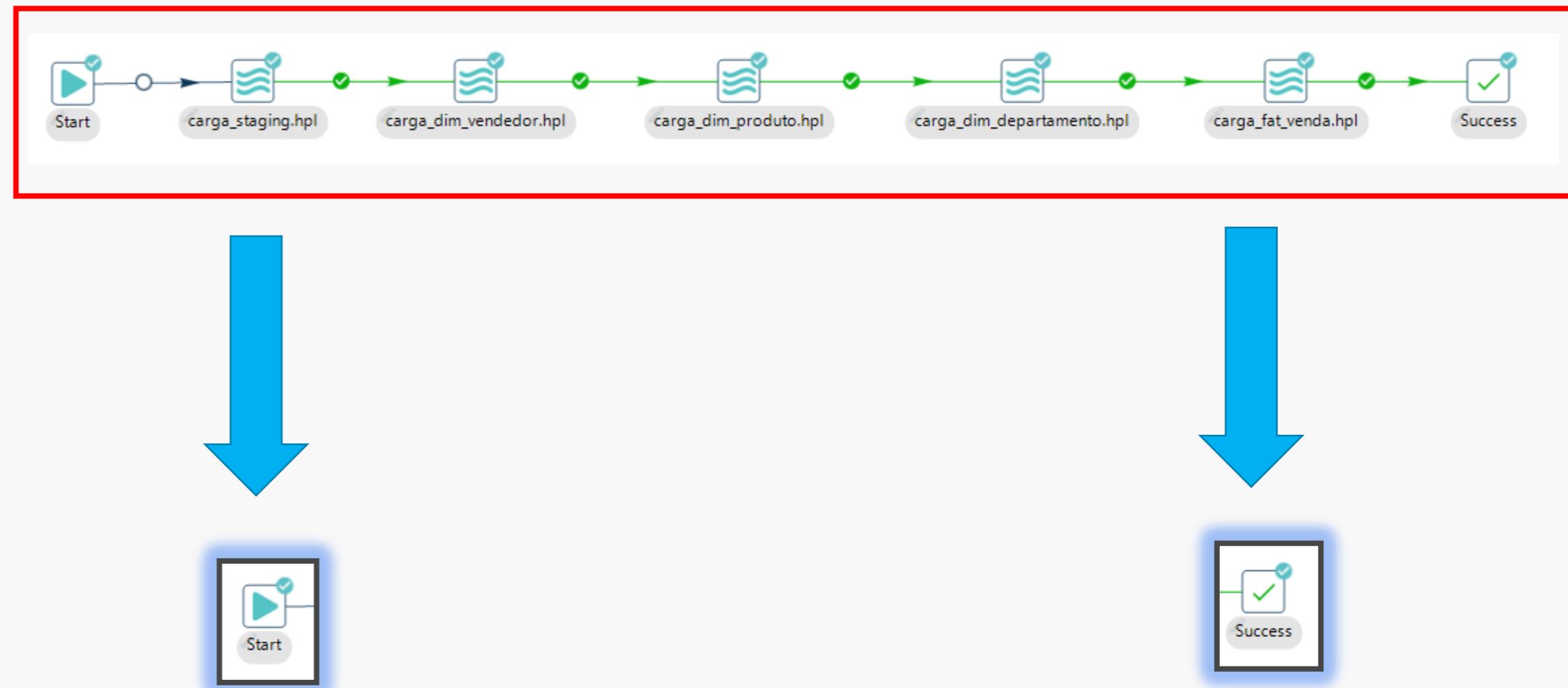
	id_venda [PK] integer	num_nota integer	sk_tempo integer	sk_produto integer	sk_vendedor integer	sk_departamento integer	qtd_venda integer	val_venda numeric (10,2)
1	1	1033	2983	4	2	1	1	1300.00
2	2	812	2983	10	2	1	1	2.15
3	3	447	2983	11	4	1	2	0.50
4	4	509	2983	19	8	1	1	1500.00
5	6	1337	2983	11	3	3	1	0.50
6	7	724	2983	9	3	5	1	12.25
7	8	375	2983	19	6	4	2	1500.00
8	9	416	2983	20	8	3	2	350.00
9	10	1085	2983	15	8	3	1	98.50

Trabalhando com Workflow

Vamos preparar um workflow de execução dos nossos pipelines.

Workflow de execução dos pipelines

Criaremos o encadeamento das rotinas para execução.



Automatizando Pipeline e Workflow

Automatizar pipeline e workflow

Automatizando Pipeline e Workflow

Criando uma automatização para as cargas.

```
===[Environment Settings - hop-run.bat]=======
Java identified as "C:\Program Files\Java\jdk-11.0.9\bin\java"
HOP_OPTIONS="-Xmx2048m" -DHOP_PLATFORM_OS=Windows -DHOP_PLATFORM_RUNTIME=Run -DHOP_AUTO_CREATE_CONFIG=Y

Consolidated parameters to pass to HopRun are
-j treinamento -r local -f C:\Users\grima\Downloads\Hop\Pipeline\trata_banco.hpl

Command to start HopRun will be:
"C:\Program Files\Java\jdk-11.0.9\bin\java" -classpath lib\*;libswt\win64\* -Djava.library.path=lib "-Xmx2048m" -DHOP_PLATFORM_OS=Windows -DHOP_PLATFORM_RUNTIME=Run -DHOP_AUTO_CREATE_CONFIG=Y org.apache.hop.run.HopRun -j treinamento -r local -f C:\Users\grima\Downloads\Hop\Pipeline\trata_banco.hpl

===[Starting HopRun]=======
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hop.core.plugins.HopURLClassLoader (file:/C:/Users/grima/Downloads/Hop/lib/hop-core-0.99.jar) to field java.net.URLClassLoader.ucp
WARNING: Please consider reporting this to the maintainers of org.apache.hop.core.plugins.HopURLClassLoader
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
2021/12/02 16:19:43 - HopRun - Enabling project 'Treinamento'
2021/12/02 16:19:54 - HopRun - Starting pipeline: C:\Users\grima\Downloads\Hop\Pipeline\trata_banco.hpl
2021/12/02 16:19:54 - trata_banco - Executing this pipeline using the Local Pipeline Engine with run configuration 'local'
2021/12/02 16:19:54 - trata_banco - Expedindo início para Pipeline [trata_banco]
2021/12/02 16:19:55 - Tabela NOVO_AUTOR.0 - Connected to database [treinamento] (commit=1000)
2021/12/02 16:19:55 - Tabela Autor.0 - Finished reading query, closing connection.
2021/12/02 16:19:55 - Tabela Autor.0 - Finished processing (I=7, O=0, R=0, W=7, U=0, E=0)
2021/12/02 16:19:55 - Muda genero.0 - Finished processing (I=0, O=0, R=7, W=7, U=0, E=0)
2021/12/02 16:19:55 - Seleciona campos.0 - Finished processing (I=0, O=0, R=7, W=7, U=0, E=0)
2021/12/02 16:19:55 - Tabela NOVO_AUTOR.0 - Finished processing (I=0, O=7, R=7, W=7, U=0, E=0)
2021/12/02 16:19:55 - trata_banco - Pipeline duration : 1.256 seconds [ 1.256" ]
2021/12/02 16:19:55 - trata_banco - Execution finished on a local pipeline engine with run configuration 'local'
```

HOP RUN

Link: <https://hop.apache.org/manual/latest/hop-run/index.html>

PRÁTICA

ENVIE AO PROFESSOR

PREPARAR UM PROJETO DE DATA WAREHOUSE

- O aluno deverá construir um projeto de DW do zero utilizando a ferramenta APACHE HOP.
- O projeto deve contemplar: Tabelas Staging, tabelas Dimensão, tabela Fato, cargas Staging, cargas Dimensão e carga Fato.
- O aluno deve executar as cargas e popular as tabelas de Dimensão e Fato.
- O aluno deverá enviar o um vídeo do projeto ao professor e demonstrar a execução do Data Warehouse.
- O aluno utilizará os dados de uma Folha de Pagamento para realizar o projeto, o arquivo encontrasse na área do aluno no curso.
- AO FINAL DEVEM SER CRIADOS:
- DIMENSÃO: Cargo, Departamento, Divisão, Funcionário
- FATO: Folha de Pagamento

