# Relatório Final

# Produtos 02, 03, 04

# Novembro/2018

# Aprovação

_____

CGCIG/SPU/MPOG

# Controle de Modificações

| Versão: 1 Revisão: 0 | Data: 13/11/2018 |
|---|---|
| Entrega da versão inicial. | |

# Sumário

# Introdução

A Secretaria do Patrimônio da União é o órgão responsável pela administração do patrimônio imobiliário da União bem como pela incorporação de novos bens e sua regularidade dominial. Para o cumprimento de sua missão a SPU contrata e produz uma variedade de produtos cartográficos. Hoje a SPU possui um acervo com abrangência nacional, com diferentes tipos de produtos, com diferentes tipos de armazenamentos e com graus distintos de conhecimentos sobre esses produtos.

Nesse cenário podemos identificar alguns desafios:

1. Catalogação dos produtos existentes;
2. Padronização Sintática e Semântica dos dados e metadados;
3. Digitalização do acervo analógico;
4. Georreferenciamento do acervo digitalizado;
5. Criação de mosaicos dos conjuntos de fotografias aéreas;
6. Vetorização e validação do produtos de natureza vetorial;
7. Conversão Sintática e Semântica dos dados para adequação aos padrões definidos em 2;
8. Disponibilização dos dados e metadados por meio de geoserviços; e
9. Integrar os Geoserviços a Infraestrutura Nacional de Dados Espaciais (INDE).

# Objetivos

O objetivo deste relatório é a proposição de soluções processuais, técnicas e tecnológicas para os seguintes problemas: Integração sintática e semântica dos metadados geoespaciais produzidos pela SPU na aplicação catálogo de metadados da SPU; Diretrizes para tratamento e armazenamento de dados Matriciais; e estratégia de arquitetura para disponibilização do módulo de geoinformação da SPUNet.

Durantes as reuniões com a SPU, essa consultoria sugeriu a criação de algumas aplicações que vieram a convergir em uma arquitetura que consiga abarcar e solucionar os desafios expostos.

# Reuniões

Durante o período da consultoria foram realizadas as seguintes reuniões:

| Tema da Reunião | Data | Participantes |
|---|---|---|
| Reunião de Requisitos | 28/9/2017 | Cárita Sampaio |
| | | Diego Moreira |
| | | Angélica Gabbi |
| | | Clauber Teixeira |
| | | Patrícia Petri |
| Reunião de Requisitos | 5/10/2017 | Cárita Sampaio |
| | | Diego Moreira |
| | | Angélica Gabbi |
| | | Clauber Teixeira |
| | | Patrícia Petri |
| | | Tarcísio Petter |
| Requisitos Geovisualizador | 9/10/2017 | Cárita Sampaio |
| | | Diego Moreira |
| | | Angélica Gabbi |
| | | Patrícia Patri |
| | | Starlone |
| | | Diego Barreto |
| Reunião de Geo | 13/10/2017 | Cárita Sampaio |
| | | Diego Moreira |
| | | Angélica Gabbi |
| | | Patrícia Patri |
| | | Gabriel Valderrama |
| Arquitetura Geo | 16/10/2017 | Cárita Sampaio |
| | | Diego Moreira |
| | | Clauber Teixeira |
| | | Gabriel Valderrama |
| | | Diego Barreto |
| | | Starlone Oliverio |
| | | Claudia Divina |

| | | |
|---|---|---|
| Videoconferência SPU /MG | 19/10/2017 | Cárita Sampaio |
| | | Diego Moreira |
| | | Jessica Carvalho |
| | | Tatiana Batista |
| | | Clodoveu |
| | | Guilherme |
| | | Clauber Teixeira |
| | | Gabriel Valderrama |
| | | Alexandre Nepomuceno |
| | | Natália Guimarães |
| | | Wagneide Rodrigues |
| | | Diego Barreto |
| | | Marcelo Fernandes |
| | | Starlone Oliverio |
| Validar Requisitos | 20/10/2017 | Cárita Sampaio |
| | | Diego Moreira |
| | | Claudia Divina |
| | | Patrícia Petri |
| | | Alexandre Nepomuceno |
| | | Natália Guimarães |
| Fechamento de Sprint | 11/14/2017 | Cárita Sampaio |
| | | Diego Moreira |
| | | Angélica Gabbi |
| | | Patrícia Patri |
| | | Starlone |
| | | Diego Barreto |
| Sistema de Gerenciamento de produção cartográfica | 11/24/2017 | Cárita Sampaio |
| | | Diego Moreira |
| | | Karla Souza |
| | | Clauber Teixeira |
| | | Patrícia Petri |
| | | Carolina Formiga |
| Fechamento de Sprint | 12/22/2017 | Cárita Sampaio |
| | | Diego Moreira |
| | | Joziel Pereira |

| | | |
|---|---|---|
| | | Starlone Oliverio |
| | | Francisco das Chagas |
| | | Patrícia Petri |
| | | Nathália Guimaraes |
| | | Gabriel Sales |
| | | Karla Souza |
| | | Alexandre Nepomuceno |
| | | Daniel Menezes |
| | | Clauber Teixeira |
| | | Wesley Sousa |
| | | Rodrigo Oliveira |
| | | Ricardo Adriano |
| | | Emerson Rodrigues |
| Reunião de requisitos | 1/18/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Karla Souza |
| | | Gabriel Valderrama |
| | | Patrícia Petri |
| | | Daniel Menezes |
| | | Rodrigo Oliveira |
| | | Tarcísio Petter |
| Reunião para Levantamento de Requisitos | 2/1/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Claúdia Divina |
| | | Patrícia Petri |
| | | Karla Souza |
| | | Daniel Menezes |
| | | Clauber Teixeira |
| | | Diego Barreto |
| | | Rodrigo de Oliveira |
| | | Gabriel Sales |
| Reunião de Requisitos GPC | 2/9/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Claúdia Divina |
| | | Clauber Teixeira |
| | | Gabriel Valderrama |

| | | |
|---|---|---|
| | | Natália Guimaraes |
| Encerramento/Iniciação de Sprint - Geovis | 2/15/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Patrícia Petri |
| | | Gabriel Sales |
| | | Karla Souza |
| | | Daniel Menezes |
| | | Claudia Divina |
| | | Clauber Teixeira |
| | | Diego Barreto |
| | | Rodrigo Oliveira |
| Definição do GPC | 2/23/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Clauber Teixeira |
| | | Gabriel Valderrama |
| | | Natália Guimaraes |
| | | Alexandre Nepomuceno |
| | | Daniel Menezes |
| Definição de GPC | 2/26/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Clauber Teixeira |
| | | Gabriel Valderrama |
| | | Natália Guimaraes |
| | | Alexandre Nepomuceno |
| | | Daniel Menezes |
| | | Karla Souza |
| | | Tarcísio Petter |
| Reunião de Encerramento do TED n°40/2013 - Parceria SPU e Exército Brasileiro | 3/7/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Clauber Teixeira |
| | | Gabriel Valderrama |
| | | Natália Guimaraes |
| | | Alexandre Nepomuceno |
| | | Daniel Menezes |
| | | Renato Fuscaldi |

| | | |
|---|---|---|
| | | Fantonio |
| | | Péricles |
| | | Felix Pessoa |
| | | Oto Buregio |
| | | Diego Barreto |
| | | João Carneiro |
| | | Vicente Zica |
| | | Sidrack de Oliveira |
| | | Reinaldo Guimarães |
| | | Valéria Grilanda |
| | | Márcia Hiroko |
| | | Antônio Ferreira |
| | | Dinarte Vaz |
| | | Erika Kimura |
| Encerramento/Iniciação - Geovis | 3/12/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Patrícia Petri |
| | | Gabriel Sales |
| | | Karla Souza |
| | | Daniel Menezes |
| | | Clauber Teixeira |
| | | Diego Barreto |
| | | Rodrigo Oliveira |
| Reunião Geovis | 3/26/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Clauber Teixeira |
| | | Alexandre Nepomuceno |
| | | Natália Guimarães |
| | | Gabriel Valderrama |
| | | Gabriel Sales |
| | | Karla Souza |
| | | Diego Barreto |
| | | Patrícia Petri |
| | | Daniel Menezes |
| | | Rodrigo Oliveira |
| Análise do Fluxo do GPC | 4/5/2018 | Cárita Sampaio |

| | | |
|---|---|---|
| | | Diego Moreira |
| | | Clauber Teixeira |
| | | Gabriel Valderrama |
| | | Natália Guimaraes |
| | | Alexandre Nepomuceno |
| | | Daniel Menezes |
| | | Diego Barreto |
| Consultoria GEOVIS | 4/13/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Alexandre Nepomuceno |
| | | Natália Guimarães |
| | | Gabriel Valderrama |
| | | Daniel Menezes |
| | | Clauber Teixeira |
| Consultoria Geo | 4/27/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Alexandre Nepomuceno |
| | | Natália Guimarães |
| | | Gabriel Valderrama |
| | | Daniel Menezes |
| | | Clauber Teixeira |
| Consultoria Geo | 5/9/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Alexandre Nepomuceno |
| | | Natália Guimarães |
| | | Daniel Menezes |
| | | Clauber Teixeira |
| Análise de Arquitetura e Integração dos Modúlos SPUNET | 5/11/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Clauber Teixeira |
| | | Oto Buregio |
| | | Diego Barreto |
| | | Rodrigo de Oliveira |
| | | Cedrick Lamalle |

| | | |
|---|---|---|
| Arquitetura SPUnet/GEO | 5/18/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Clauber Teixeira |
| | | Diego Barreto |
| | | Marcelo Marques |
| | | Rodrigo de Oliveira |
| | | Daniel Andrade |
| | | Elias Amadeu |
| | | Cedrick Lamalle |
| Reunião Arquitetura SPUNET | 6/4/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Oto Buregio |
| | | Marcelo Marques |
| | | Elias Amadeu |
| | | Gabriel Valderrama |
| | | Diego Barreto |
| | | Thiago César |
| | | Daniel Andrade |
| | | Marcelo Florêncio |
| | | Fábio Vilela |
| | | Clauber Teixeira |
| Geovis-GPC | 7/4/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Clauber Teixeira |
| GPC | 7/27/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Clauber Teixeira |
| | | Daniel Menezes |
| | | Gabriel Valderrama |
| Consultoria CGCIG | 9/6/2018 | Cárita Sampaio |
| | | Diego Moreira |
| | | Gabriel Valderrama |
| | | Alexandre Nepomuceno |
| | | Gabriel Sales |
| Consultoria Geo | 9/14/2018 | Cárita Sampaio |
| | | Diego Moreira |

| | | |
|---|---|---|
| | | Gabriel Valderrama |
| | | Alexandre Nepomuceno |
| | | Gabriel Sales |
| | | Clauber Teixeira |
| Entrega dos produtos - Consultoria GEO | 10/17/2018 | Cárita Sampaio |
| | | Diego Moreira |

# Contextualização

Durante o processo de construção da consultoria foram realizadas diversas reuniões com desenvolvedores, técnicos, gestores e outros consultores a fim de conseguir entender a dimensão dos desafios enfrentados e consensuar entre os diversos atores quais são os desafios que serão o foco da consultoria.

## Consensuando os desafios

Durante diversas reuniões foi possível observar que embora existisse por parte dos técnicos e da gestão uma visão sobre os problemas enfrentados pela CGCIG faltava ainda um entendimento da natureza dos desafios enfrentados e como relacioná-los para propor uma solução que fosse eficaz.

Com o intuito de  criar esse consenso foi proposto pela consultoria algumas reuniões com os técnicos da CGCIG para se criar um entendimento comum sobre os desafios. Dessas reuniões tiramos os seguintes *insights*:

1. A SPU lida com informações geográficas em diversas escalas possuindo documentos cartográficos, topográficos e arquitectónicos, que juntos formam o acervo de dados geográficos da SPU.
2. O negócio da SPU é intrinsecamente geoespacial, todo Imóvel da União possui um ou mais dos seguintes atributos: posição; comprimento e área.
3. Embora o conjunto de dados geográficos possuam elementos relacionados aos modelos de negócio da SPU esses dados devem estar separados.
    a. Por exemplo, no sistema de geoidentifcação a geometria de um imóvel deve ser armazenada junto com seu dados tabulares mesmo que ela esteja em um produto cartográfico.
4. Diversos sistemas fazem uso de geoinformação e precisam publicar e visualizar espacialmente seu dados.
5. O catálogo de metadados deve ser aderente aos padrões sintáticos e semânticos propostos pela CONCAR.
6. Os dados que tiverem seus metadados catalogados devem estar armazenado de forma adequada que propicie a fácil recuperação dos dados por qualquer um.
7. O processo de gerenciamento da produção cartográfica deve ser informatizado.

O grande esforço da consultoria nessa fase foi o de que a equipe técnica da CGCIG fosse capaz de enunciar os problemas e a partir daí ter condições de

propor conjuntamente as melhores estratégias para solucionar os problemas apresentados.

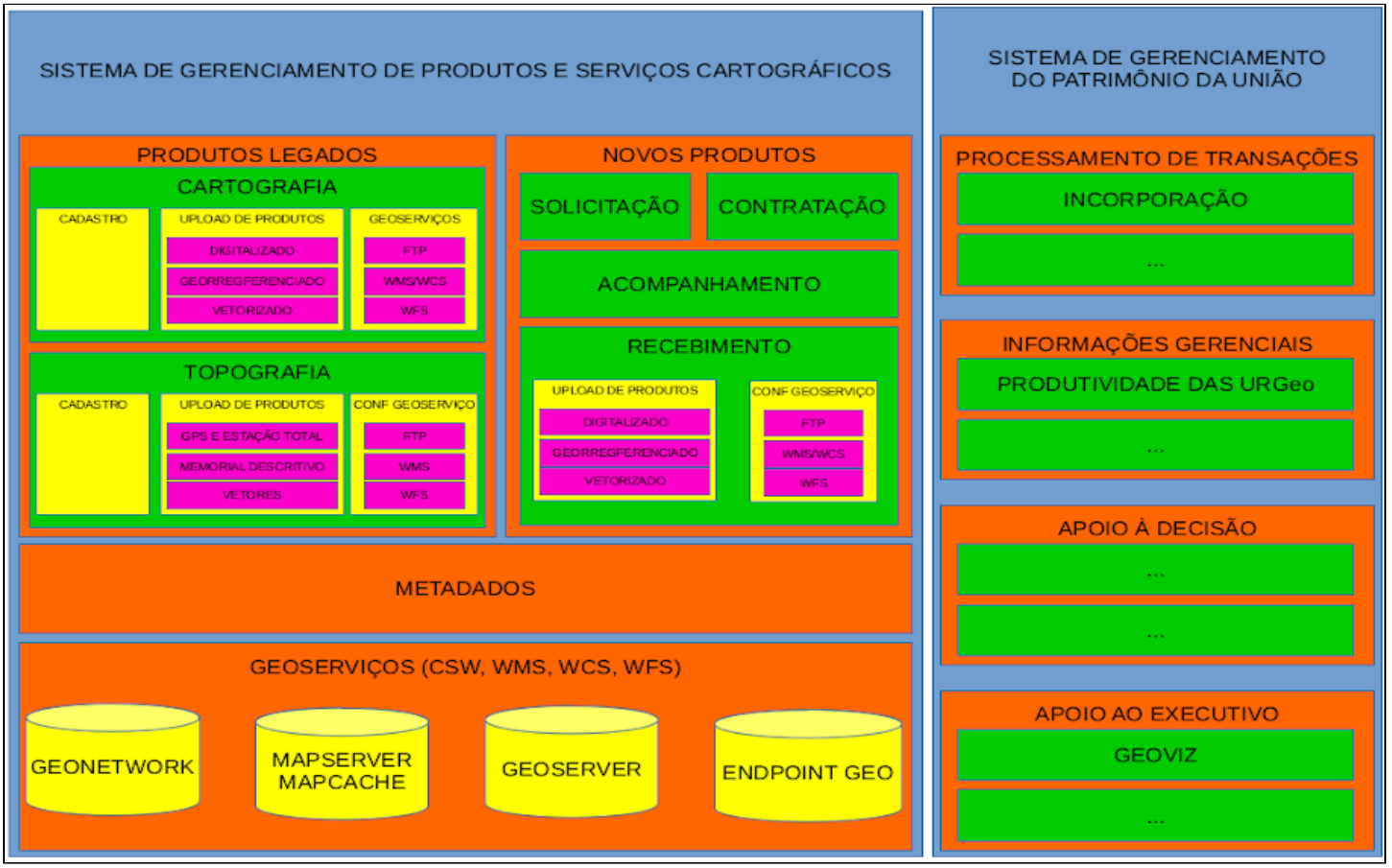
## Produção cartografia X negócio da SPU

A CGCIG lida com dois “negócios” que devem ser tratados de formas distintas.

1. Gerenciamento de Produtos Cartográficos/Topográficos (Não é o Negócio da SPU, apoia/suporta o negócio da SPU)
2. Gerenciamento do Patrimônio da União (Negócio da SPU)

Dessa forma, parece simples que devam existir duas soluções, uma para o Gerenciamento da Produção Cartográfica e uma outra para o Gerenciamento do Patrimônio da União.

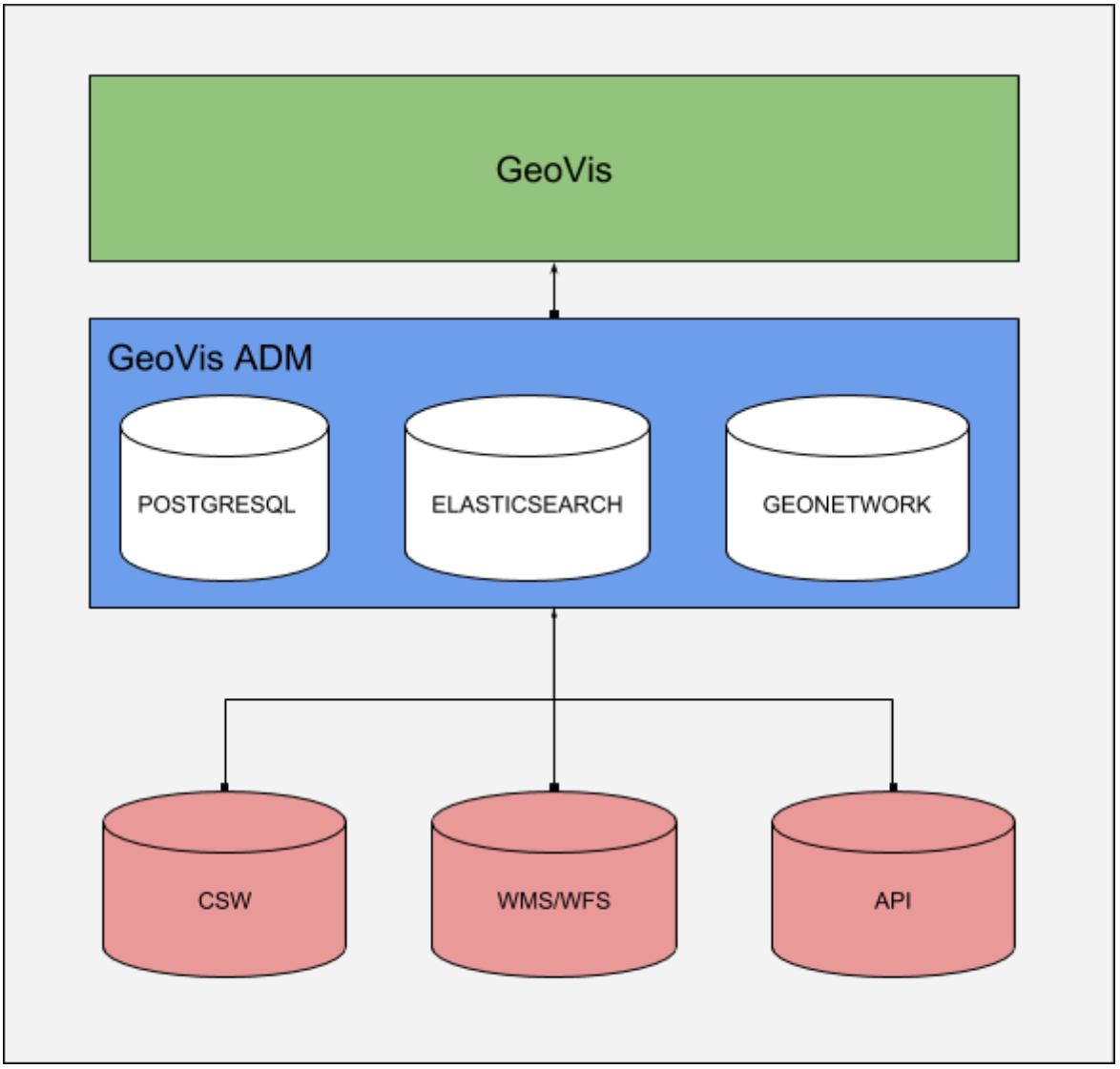
Essa consultoria propôs inicialmente o seguinte diagrama para representar esses dois grandes sistemas.



Esse diagrama foi apresentado em novembro de 2017 e a consultoria seguiu pelo aprimoramento somente da parte de Gerenciamento de produtos e Serviços cartográficos.
Também fez parte dos esforços dessa consultoria a conceituação e desenho de uma arquitetura para o GEOVIS.

## GEOVIS

O GeoVis foi concebido para ser um SIG genérico para visualização e consulta de dados e metadados.
Sua arquitetura permite que sejam acopladas diversas fontes de dados e APIs de outros sistema da SPU.

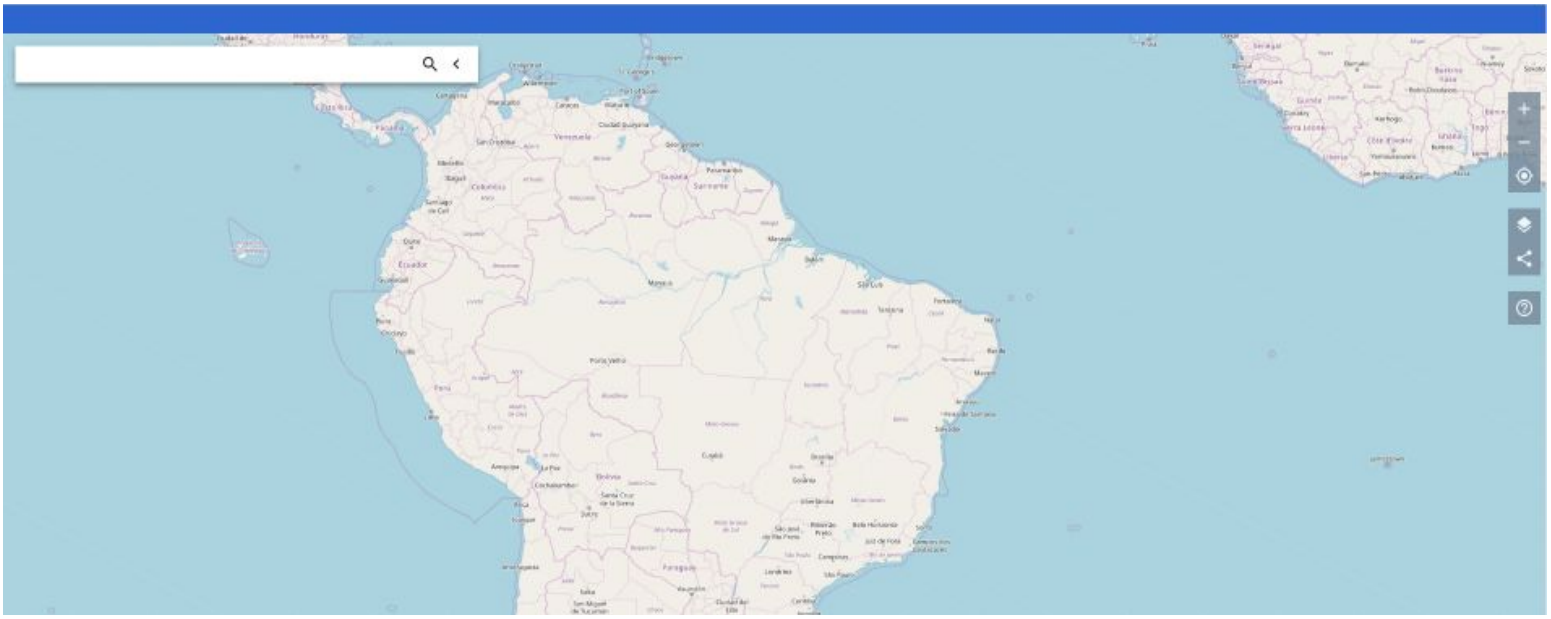



O GeoVis possui dois componentes, um componente de administração e um componente de visualização.
Os dados de administração do GeoVis ADM são armazenados em um Banco PostgreSQL e os metadados e informações sobre camadas são armazenados no Geonetwork e no Elasticsearch para aumentar o desempenho

O componente de administração possui 3 elementos basilares:

1. **Cadastro de Catálogo de metadados** - Esse cadastro permite a inclusão de catálogos de metadados que seguem o padrão *Catalogue Service* da OGC. Uma vez que um catálogo é cadastrado, é feito o *harvesting* para a instância de geonetwork do GEOVIS, para otimizar as consultas.
2. **Cadastro de Camadas de Informação via serviço -** Esse cadastro permite a inclusão de camadas disponibilizadas através do padrão WMS e/ou WFS.
3. **Cadastro de APIs de consulta -** Esse cadastro permite a inclusão de APIs de consulta que retornem arquivos no formato GEOJSON. É possível cadastrar os parâmetros de consulta e expressões regulares.
    a. Por exemplo: Uma API onde seja possível pesquisar imóveis a partir do cpf ou cnpj do proprietário, no sistema seria cadastrado uma regex para identificar cpf e cnpj e quando um cpf ou cnpj for digitado no campo de busca do GeoVis, ele saberá redirecionar a consulta para a API que atenda aquela regex. Por exemplo:
        i. **cpf** - /^\d{3}\.\d{3}\.\d{3}\-\d{2}$/
        ii. **cnpj** - /^\d{2}\.\d{3}\.\d{3}\/\d{4}\-\d{2}$/

O GeoVis é uma interface minimalista com apenas um campo onde é possível consultar metadados, dados vindos das APIs cadastradas e camadas de informações.
Toda complexidade deve ficar longe dos olhos dos usuários.



# Gerenciador da Produção Cartográfica

Com o amadurecimento da idéia do Sistema de Gerenciamento de Produtos e Serviços Cartográficos surgiu o Gerenciador da Produção Cartográfica - GPC. O GPC foi pensado de forma que os fluxos dos processos de trabalho possam ser criados em função de cada tipo de produto cartográfico. Dessa forma o gestor tem liberdade para criar fluxos que melhor atendam às suas especificidades tanto no que tange aos produtos como as equipes disponíveis.
Podemos dividir o GPC nas seguintes partes:

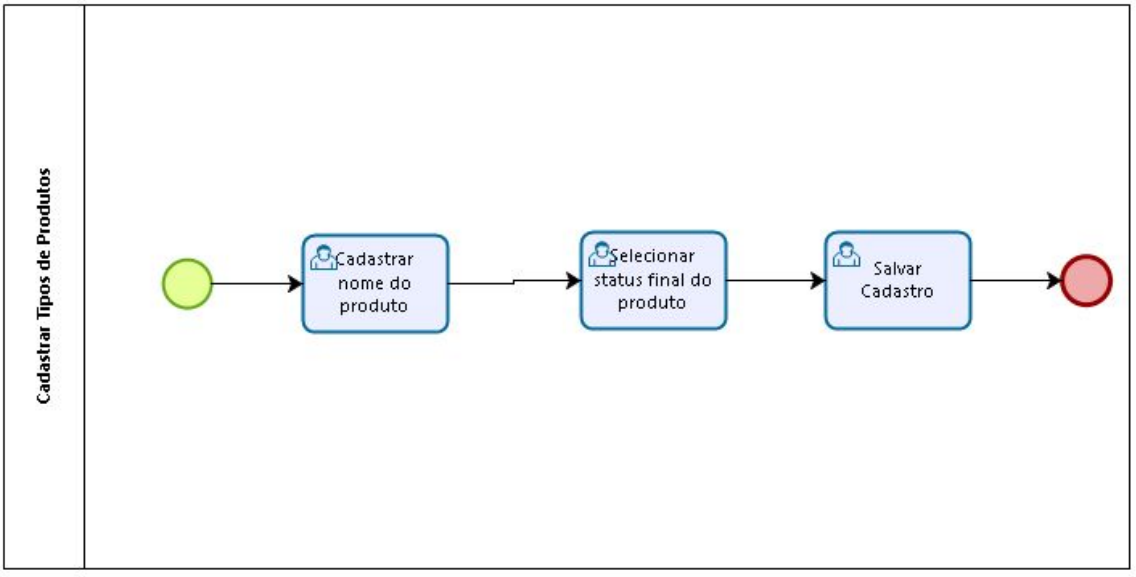
## Gerenciamento das linhas de produção

Neste componente estão listados os processos referentes aos itens de sistemas que representarão os fluxos de Trabalho

### Cadastro de tipos de produto

Aqui devem ser cadastrados os tipos de produtos e o status que o produto deve alcançar quando passar por toda a esteira de produção:

Ex.: ortofoto, ortofotocarta, planta topográfica, fotografia aérea



## Cadastro de status do produto

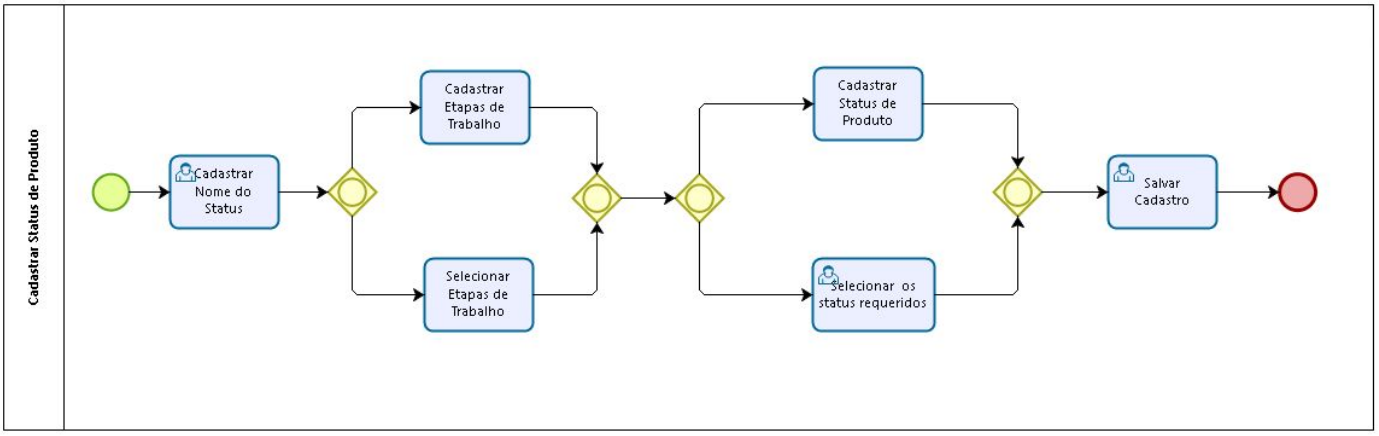
Aqui é apresentado o fluxo de cadastramento de Status. A idéia principal do GPC é que os produtos possuem um status inicial, status intermediários e status finais e que as execução das etapas de trabalho leva um produto de um status a outro até alcançar um status final.

Ex: Produto ortofotocarta

**Status Inicial** - Analógico

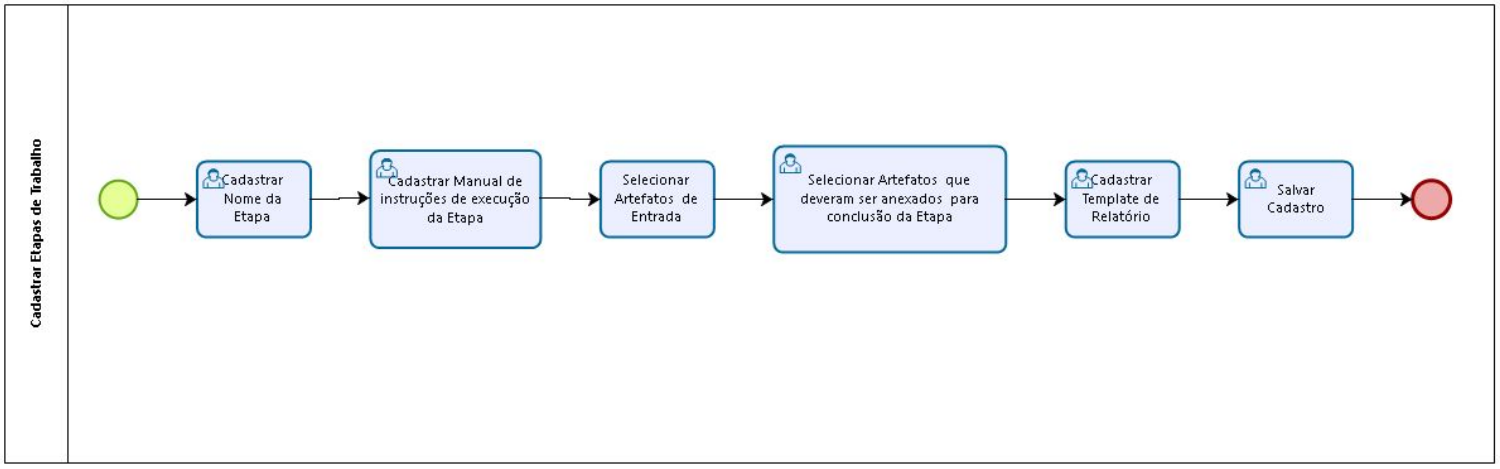
**Status Intermediários** - Digitalizado, georreferenciado, vetorizado, editado, validado(processo cartográfico)

**Status Finais** - Rejeitado(Caso não atenda a algum padrão de qualidade das fases intermediárias), Finalizado



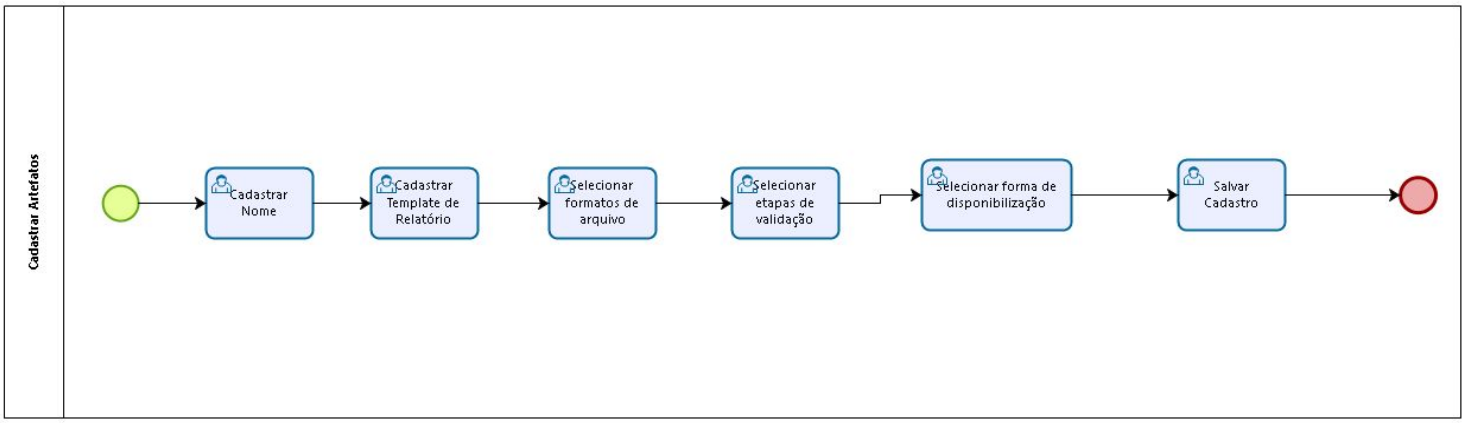
## Cadastro de etapas de trabalho

As etapas de trabalhos são os elementos responsáveis para levar um produto de um status a outro. Ela é composta de um manual de instrução para a execução da tarefa, os artefatos de entrada (Mapa Impresso, Arquivo digitalizado), os artefatos que serão gerados na etapa de trabalho(Arquivo Digitalizado, Arquivo georreferenciado) e o template de relatório.
O template de relatório deve seguir o padrão XLS Form, disponível em: http://xlsform.org/en/. Esse padrão possibilita a criação de formulários dinâmicos na WEB dando facilidade ao gestor para configurar os relatórios de entrada.



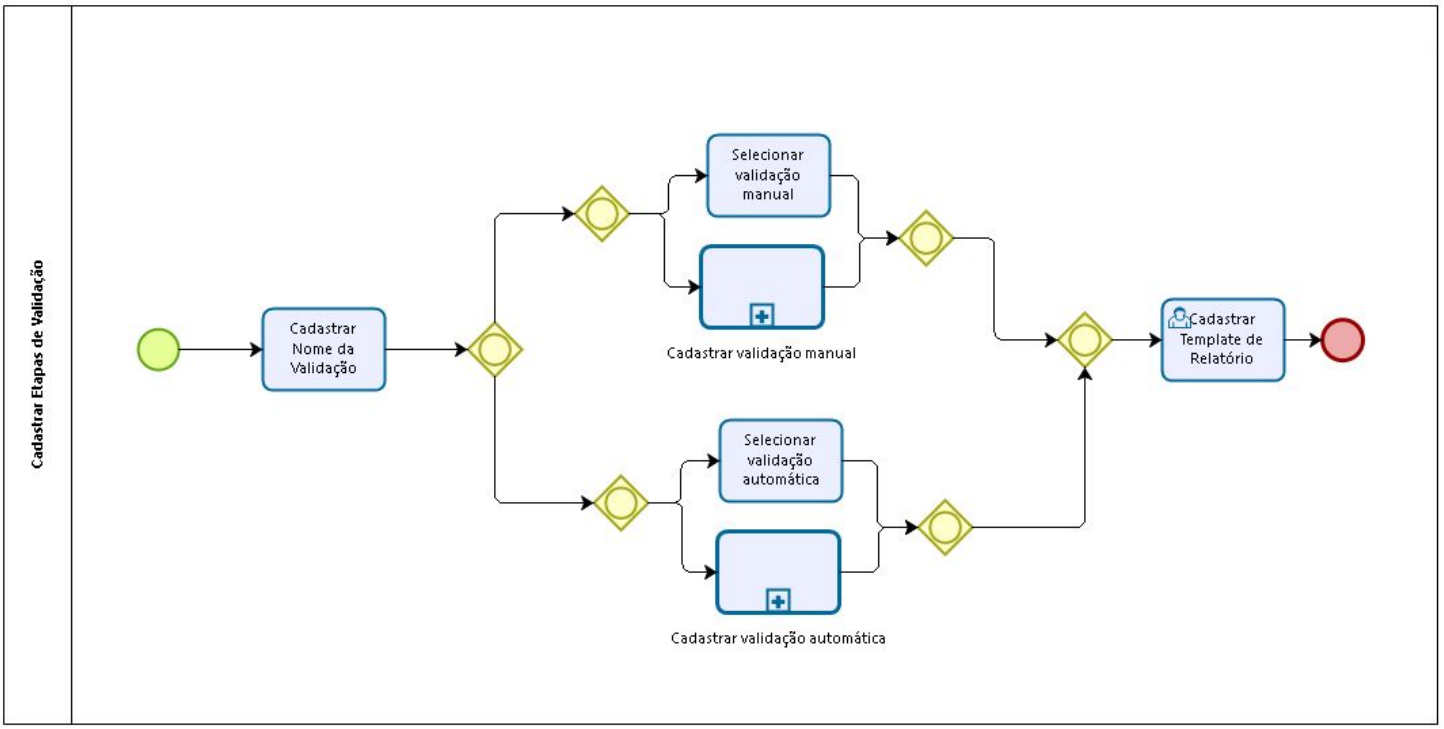

## Cadastro de artefatos

O artefato é a materialização de um status. Por exemplo, um status *matricial georreferenciado* é materializado através de um arquivo de imagem no formato TIFF / BigTIFF / GeoTIFF (.tif) que pode ser disponibilizado através de ftp, wms e/ou wcs. As etapas de validação que podem ser manual e/ou automática tem por finalidade conferir se os artefatos estão de acordo com os padrões que foram definidos para ele e com a forma de disponibilização selecionada.



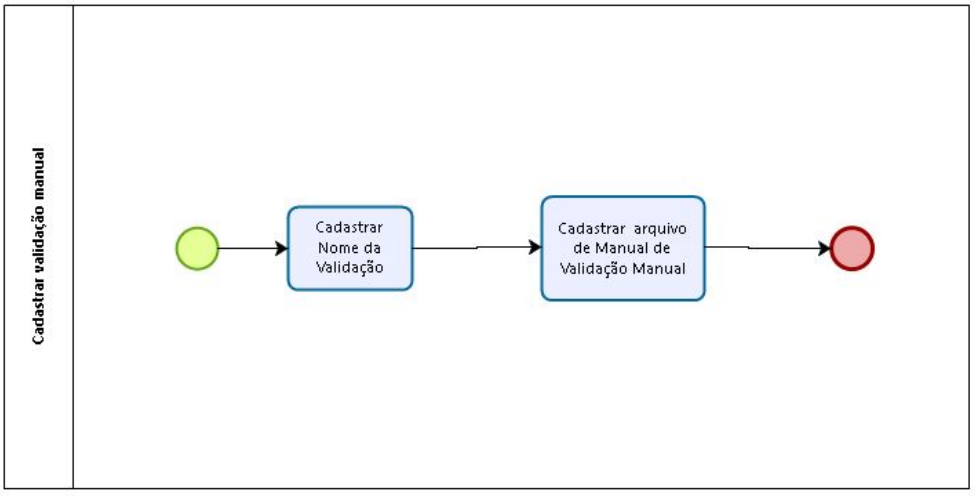
### Cadastro de Etapas de validação

Aqui são cadastradas as etapas de validação pela qual um artefato deve ser submetido. As etapas podem ser manuais ou automáticas.



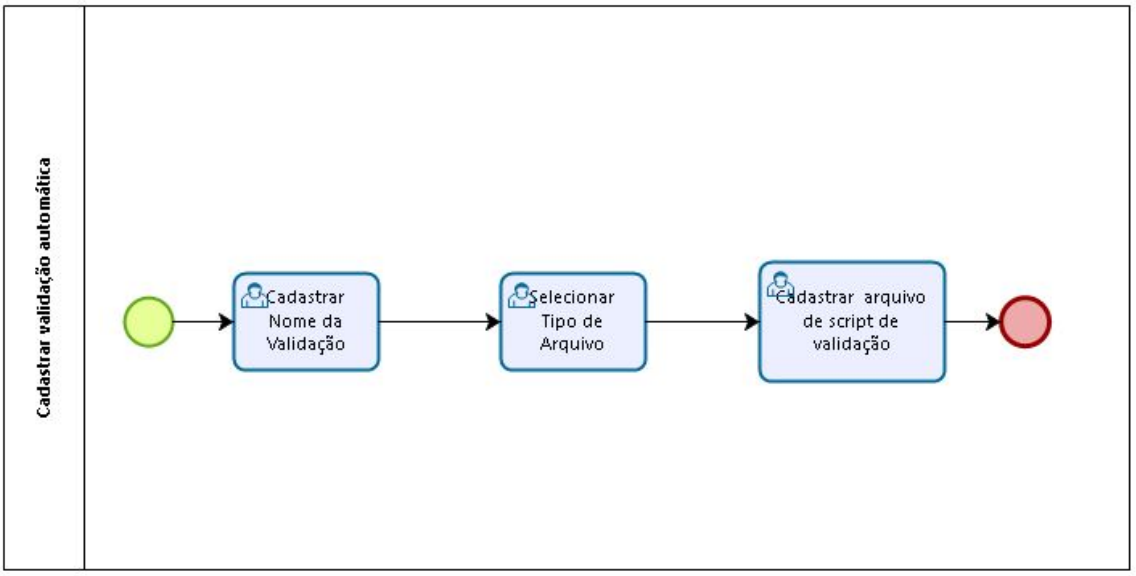

### Cadastro de validação manual

É considerado validação manual toda e qualquer validação onde é necessária a interação de um técnico com um artefato. Por exemplo, a validação do georreferenciamento de uma imagem. Pode ser feita a partir da visualização do arquivo georreferenciado e da conferência do RMS. O cadastro da validação manual imprescinde de ter um manual para a validação.



**Cadastro de validação automática**

A inclusão da validação automática é um avanço pois cria a possibilidade de dispensar a validação manual. No exemplo anterior do georreferenciamento, pode se exigir que o técnico faça um upload dos pontos que foram utilizados no georreferenciamento e dos pontos de controle. Desta forma um script lê a imagem digitalizada refaz o georreferenciamento, calcula o RMS e verifica os erros dos pontos de controle. Para isso é necessário descrever os tipos de arquivos e fazer o upload do script de validação.

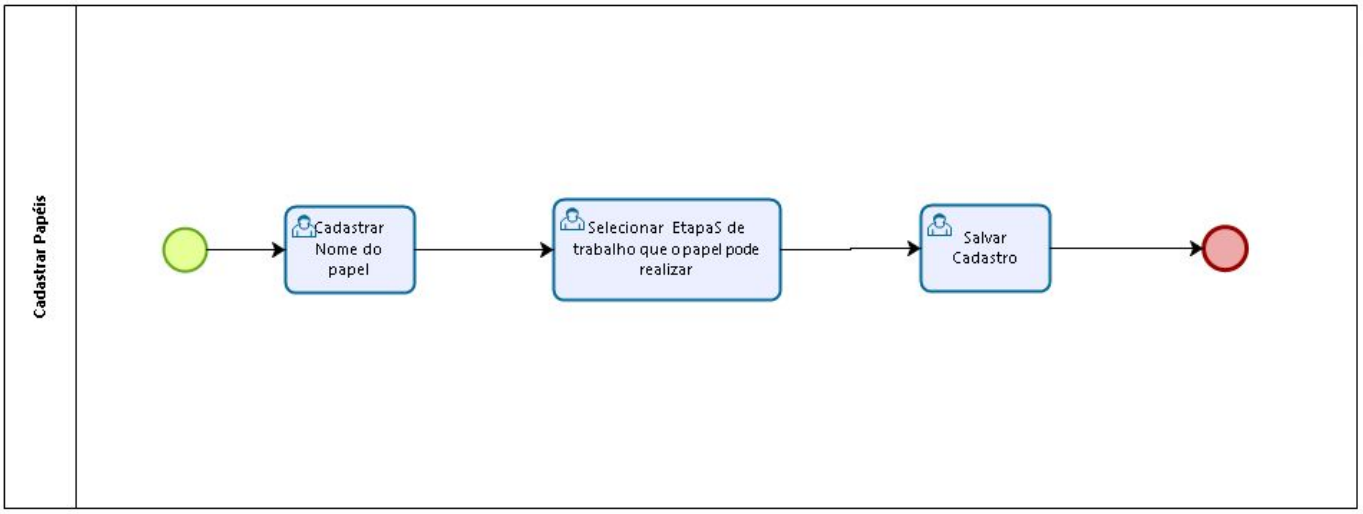


## Gerenciamento de Pessoal

O componente de gerenciamentos de papéis tem como responsabilidade o cadastro dos profissionais e os papéis que eles podem exercer.

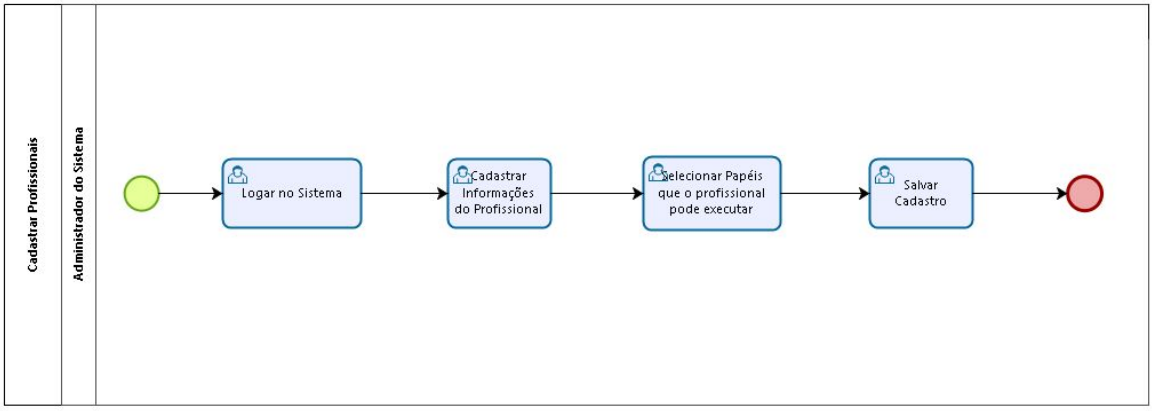
### Cadastro de papéis

Os papéis agrupam as etapas de trabalho que um profissional pode realizar.



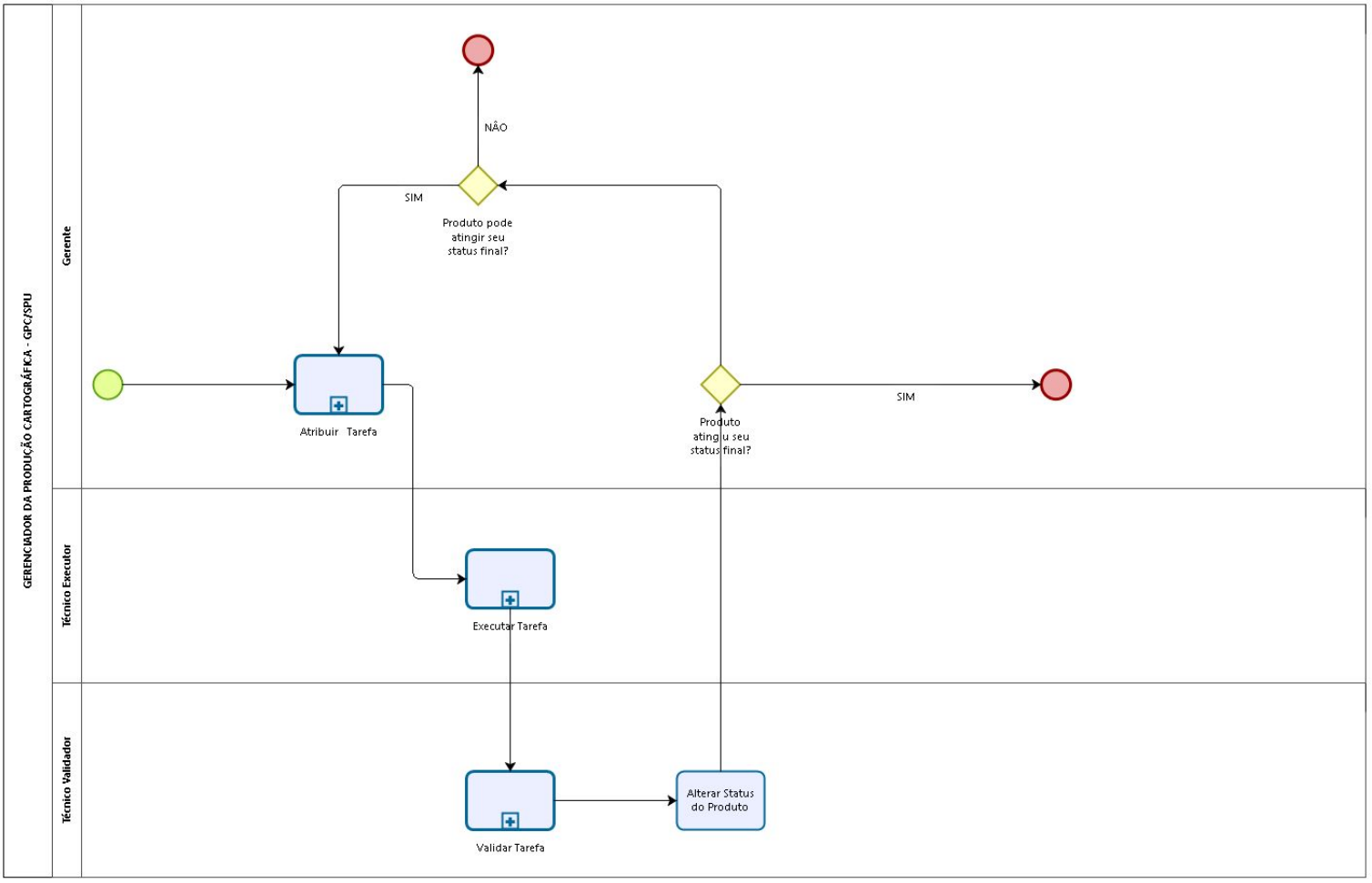

### Cadastro de profissionais

No cadastro do profissional existirá todas as informações necessárias para identificação de um profissional bem como os papéis que ele pode exercer.



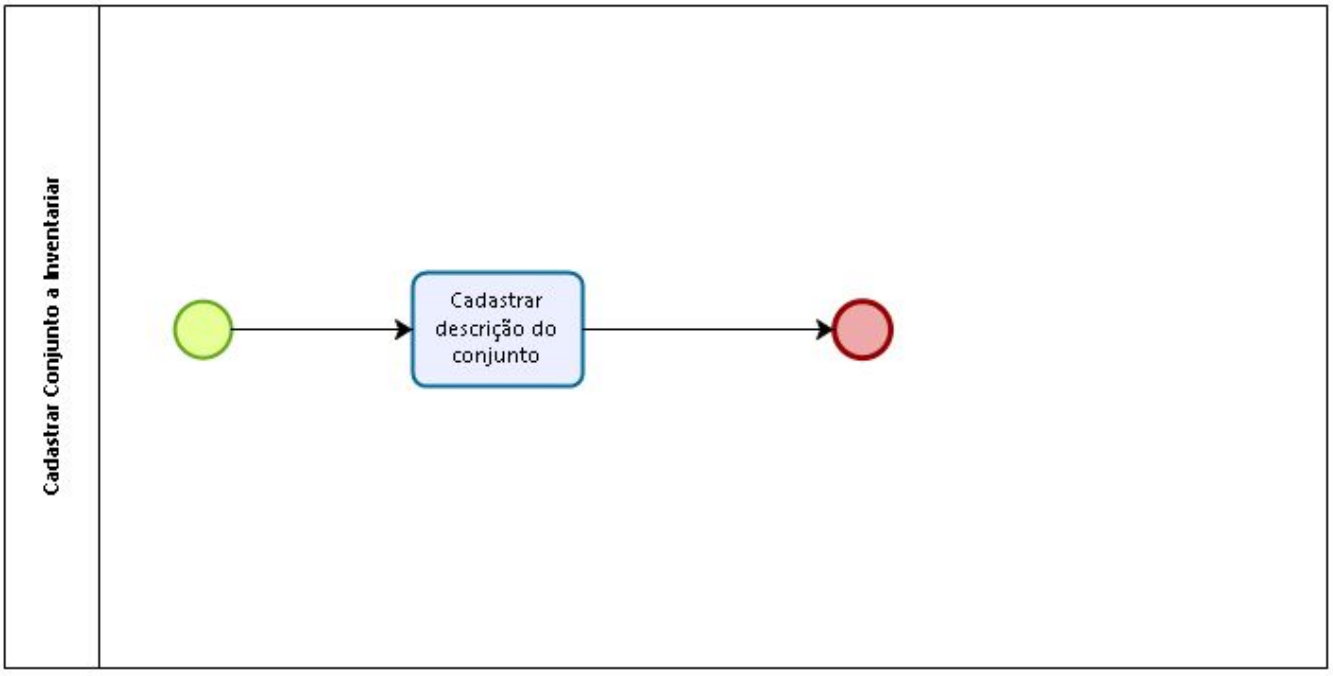
## Gerenciamento da Produção

No complemento de gerenciamento da produção temos o fluxo inicial que é um ciclo baseado nos status dos produtos e as tarefas que devem ser executadas para que um produto chegue ao seu status final.



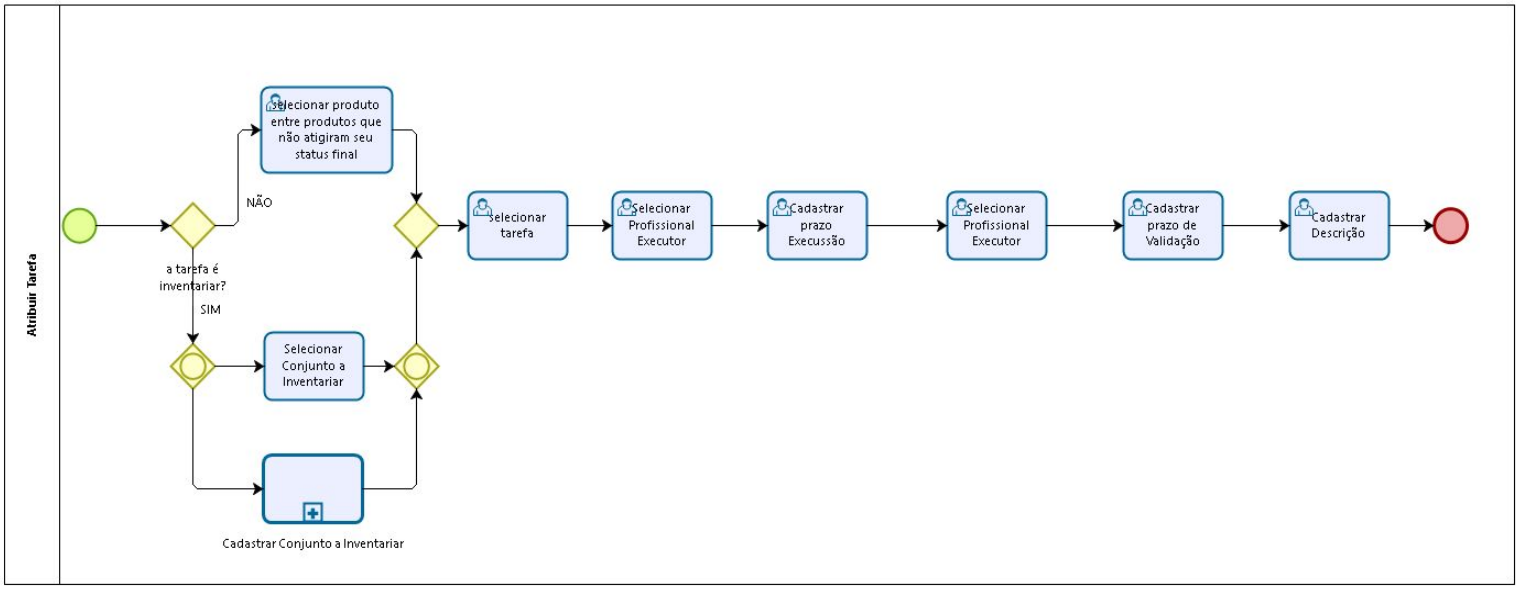
## Cadastrar Conjunto de dados a inventariar

Para se ter controle e poder estimar o esforço e atribuir tarefas é preciso cadastrar conjuntos de dados a inventariar. Esse cadastro é somente a descrição do CDG. Por exemplo, inventariar os 50 CDs que estão no armário branco.




## Atribuir tarefa

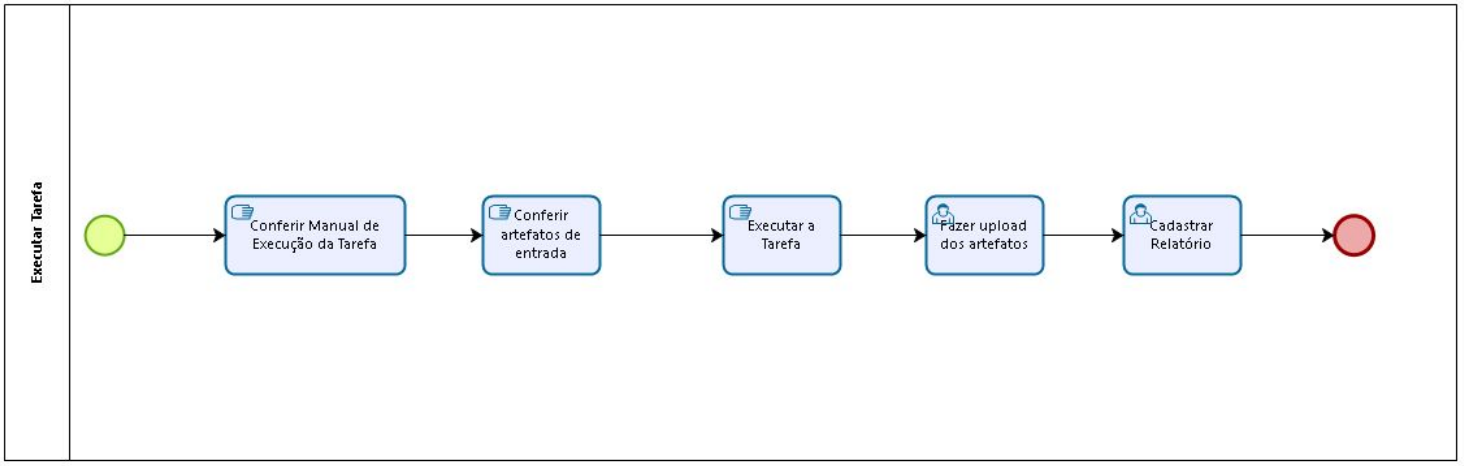
A atribuição da tarefa visa o controle das atividades bem como a possibilidade de avaliação de desempenho. Nessa etapa é selecionada a tarefa, o executor, o prazo para execução, o validador e o prazo para validação.



### Executar Tarefa

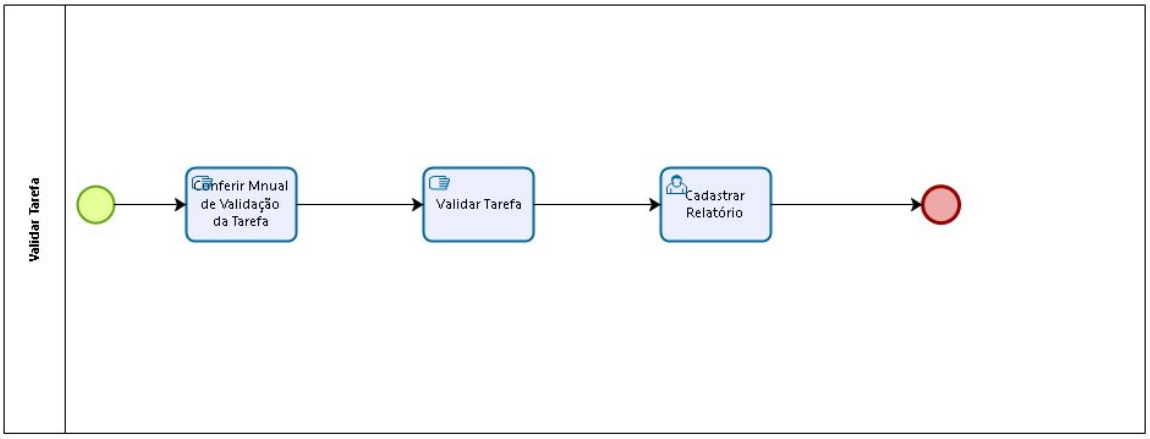
O fluxo de execução da tarefa possui apenas dois elementos que são suportados pelo GPC: O upload dos artefatos e o cadastro do relatório.

O *upload* dos artefatos é uma parte simples e essencial do GPC, pois organiza o armazenamento dos produtos digitais. Hoje os metadados são cadastrados mas os dados continuam nas URGEOs. Falaremos mais dessa parte no capítulo **Descrição das funcionalidades de importação, armazenamento e disponibilização de dados**




### Validar tarefa

Aqui só é expressado a validação manual da tarefa, uma vez que os processos de validação automática serão implementados em cada script. A validação manual termina com o cadastramento de um relatório sobre o processo de validação.



## Modelo de dados do GPC e Metadados

Um dos desafios da criação do modelo de dados do GPC é a integração dos dados da produção cartográfica com os metadados dos produtos cartográficos. Um produto cartográfico, dentro do GPC é pensado como um conjunto de artefatos.

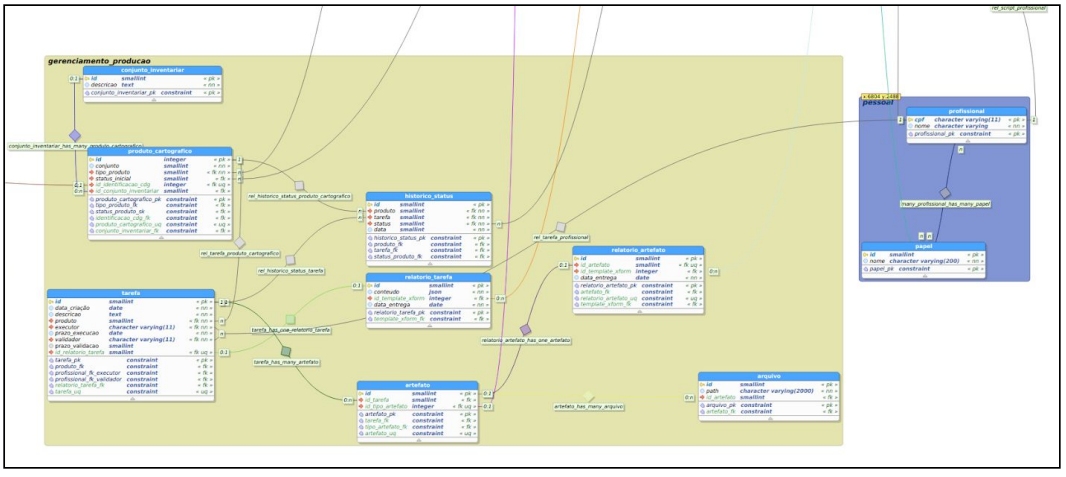
Para a CGCIG a cada novo status de um produto cartográfico se tem um novo produto cartográfico e dessa forma, deveria ser criado um novo metadado no Catálogo de Metadados.
Por exemplo, um carta topográfica analógica que deve passar pelas etapas de trabalho digitalização, georreferenciamento, vetorização e validação teria ao fim de cada processo um novo metadado. Assim seriam 5 metadados para um produto cartográfico. Esse processo onera os técnico e dificulta o processo de busca por um dado.

Essa consultoria entende que só existe um produto cartográfico e que ele é o produto cartográfico em seu status final, sendo assim, deve ter somente um metadado. Os status inicial e intermediários e as informações sobre os processos de transformação devem ser descritos na seção Qualidade em Linhagem. Esse preenchimento deve ser automatizado a partir das informações do produto inicial e das informações dos relatórios que são preenchidos em de cada etapa de trabalho.

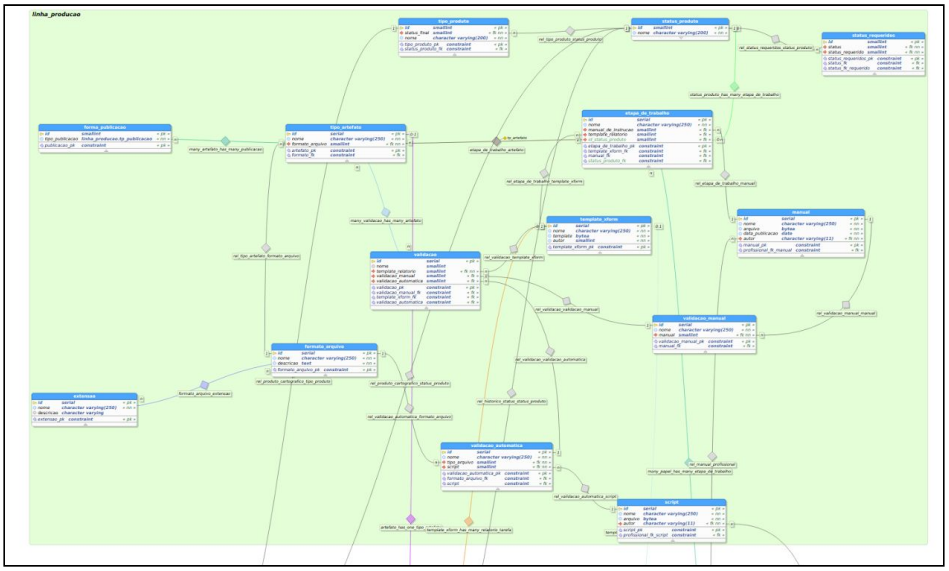
Essa opção integra o preenchimento dos metadados com as etapas de trabalho, facilitando o preenchimento e a manutenção dos metadados.

A seguir mostraremos o modelo de dados dos metadados e sua integração com o modelo de dados do GPC.

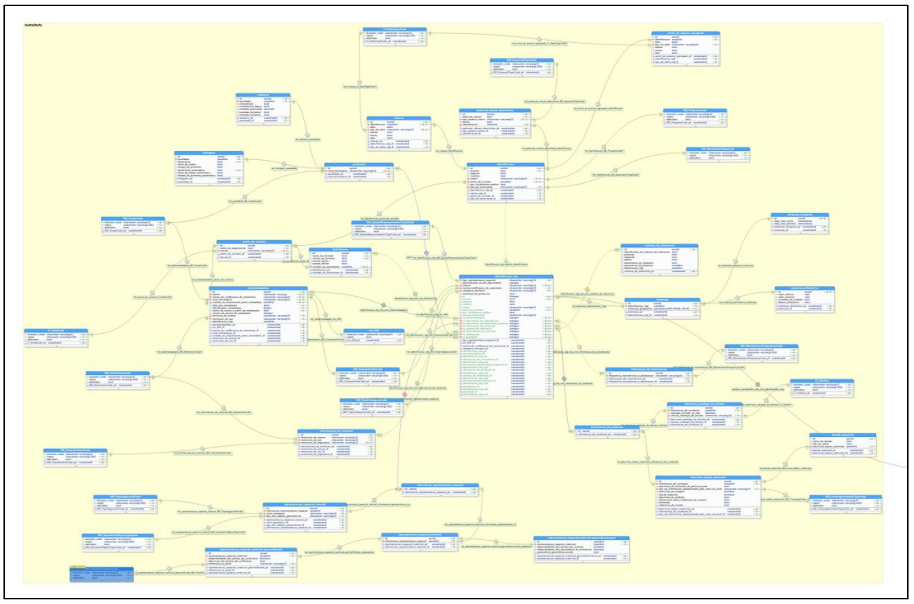
Aqui é representado os modelos dos componentes gerenciamento da produção e gerenciamento de pessoal.



Aqui é apresentado o modelo da estruturação da linha de produção.



Aqui é apresentado o modelo de dados do metadado



O modelo de metadados foi criado a partir do Perfil MGB e completamente aderente às proposições de automação.

A conexão entre o modelo de dados metadado e o modelo de dados do GPC é feito entre as classes identificacao_cdg e produto_cartografico. Sendo que um produto cartográfico pode ter somente uma identificação.

## Publicação dos Metadados

Como solução para a publicação dos metadados essa consultoria indica a seguinte estratégia:

1. Migração por meio de ETL dos metadados cadastrados no Catálogo de Metadados para o novo modelo de metadados
2. Criação de uma view materializada com o conteúdo dos metadados do CDG no formato XML.
    a. a view deve possuir os seguinte elementos:
        - **Identifier**: identificador único do CDG
        - **Typename**: Nome do tipo de metadata;normalmente o valor da tag do elemento raiz (ex. csw:Record, gmd:MD_Metadata)
        - **Schema**: esquema para os metadados; normalmente é o valor do namespace (ex. http://www.opengis.net/cat/csw/2.0.2, http://www.isotc211.org/2005/gmd)
        - **InsertDate**: Data de inclusão do metadado
        - **XML**: XML com a representação completa do metadado
        - **AnyText**: conjunto do valores de texto do XML. São os texto que serão utilizados para as pesquisas.
        - **BoundingBox**: utilizar o formato WKT ou EWKT geometry
        - **Keywords**: lista de tags separada por vírgula.
        - **Links**: links estruturados no formato “name,description,protocol,url[^,,,[^,,,]]”
3. Utilização da ferramenta pycsw, conectada a view criada para expor os metadados de acordo com os padrões da INDE.

# Importação, armazenamento e disponibilização de dados

Um dos requisitos mais importantes do GPC é a centralização do armazenamento dos produtos cartográficos. Uma vez que os produtos analógicos e mesmo os digitais estão espalhados pelo território facultando o descobrimento e o acesso a esses produtos somente a pessoas fisicamente próximas. Assim a centralização do armazenamento possibilitará o acesso imediato e “irrestrito” para todos os funcionários da SPU bem como para a população.

Quanto à natureza dos arquivos que constarão no GPC temos apenas 4 possibilidades: documentos matriciais; documentos vetoriais; documentos matriciais georreferenciados e documento vetoriais georreferenciados.
A natureza dos documentos condicionará o armazenamento e a disponibilização, ficando o processo de importação comum para todos

## Importação

A importação dos produtos cartográficos será realizada após a conclusão de uma tarefa do GPC. Ao fim de uma tarefa é necessário que se faça o upload de um artefato e a depender de seu tipo o sistema se responsabilizará pelo correto armazenamento do arquivo.
Como regra de importação o sistema deverá reescrever o nome do arquivo para corresponder ao nome do produto em minúsculo sem diacríticos e espaços acrescido dos prefixos: m_n para arquivos matriciais não georreferenciados; v_n para arquivos vetoriais não georreferenciados; m_g para arquivos matriciais georreferenciados e v_g para arquivos vetoriais georreferenciados.
Quando em uma linha de produção houverem etapas de trabalhos que gerem arquivos de mesma natureza, deverá ser acrescido o nome da etapa o nome do artefato ao nome do arquivo.
Assim, se um produto se chamar Carta topográfica Brasília, ela poderá assumir os seguintes nomes:

- m_n_carta_topografica_brasilia - Para o arquivo digitalizado
- m_g_carta_topografica_brasilia - Para o arquivo georreferenciado
- v_g_carta_topografica_brasilia_vetorizada - Para o arquivo vetorizado
- v_g_carta_topografica_brasilia_validada - Para o arquivo validado

## Armazenamento

O armazenamento tanto dos arquivos matriciais quanto vetoriais deverão ser feitos em disco, em pastas separadas por tipo de arquivo seguindo a nomenclatura, m_n, m_g, v_n e v_g.
No banco de dados, deverá ser armazenado somente o *path* do arquivo e a manutenção dos nomes dos arquivos e locais de salvamento deve ser mantido pelo GPC, assim se um produto for renomeado todos os artefatos devem ser renomeados automaticamente assim como o *path.*

## Disponibilização

A disponibilização dos arquivos será feita via serviço conforme a natureza dos tipos de artefato. Assim existiram 4 serviços de disponibilização:

- FTP - Para arquivos matriciais e vetoriais sem georreferenciamento
- WMS - Para arquivos matriciais e vetoriais georreferenciados
- WFS - Para arquivos vetoriais georreferenciados
- WCS - Para arquivos matriciais georreferenciados

O serviço de FTP deve ser implementado utilizando o que for mais adequado para a SPU e para a fábrica de software.
Para os serviços geo (wms, wfs e wcs) essa consultoria indica a utilização do software geoserver. O geoserver possui uma API REST onde é possível realizar a integração com o GPC de forma simples e eficiente.

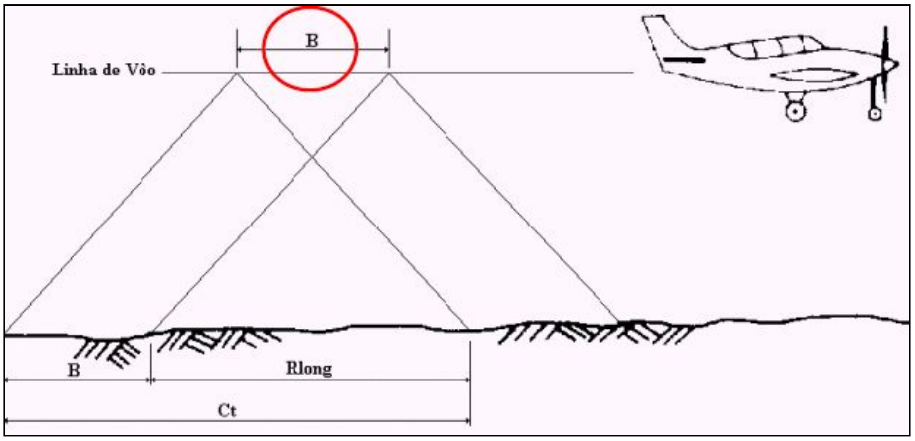
# Geração de Mosaicos de aerofotos

O objetivo deste item é a propor uma metodologia para geração de mosaicos não controlados de aerofotos tendo como objetivo a obtenção de um panorama geral da área imageada, considerando para tanto que o resultado do processamento será um mosaico georreferenciado não controlado.
Esse procedimento é adequado para conjuntos de aerofotos analógicas que foram georreferenciadas. Usualmente esse conjunto de aerofotos é produto de um vôo aerofotogramétrico que tinha por finalidade a produção cartográfico. Ao longo dos anos conjuntos e mais conjuntos dessas fotografias analógicas se acumularam gerando um acervo riquíssimo mas pouco ou não utilizado.
A fim de possibilitar o acesso dos técnicos, gestores e população a esses dados é necessário que eles sejam digitalizados, georreferenciados e mosaicados.

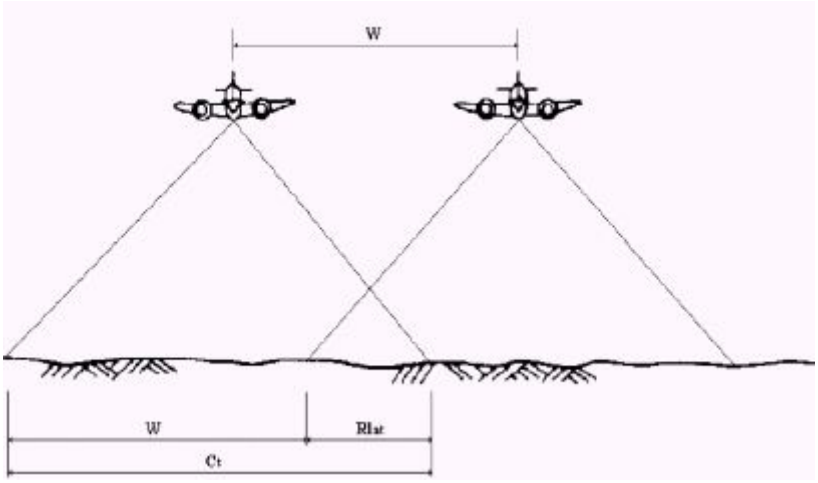
A nossa metodologia é focada no processo de mosaicagem. Para isso é preciso que entendamos algumas características dos levantamentos aerofotogramétricos. Dois pontos importantes para a cria para a nossa metodologia são:

- Superposição(recobrimento) longitudinal e lateral
- Perspectiva de ponto central

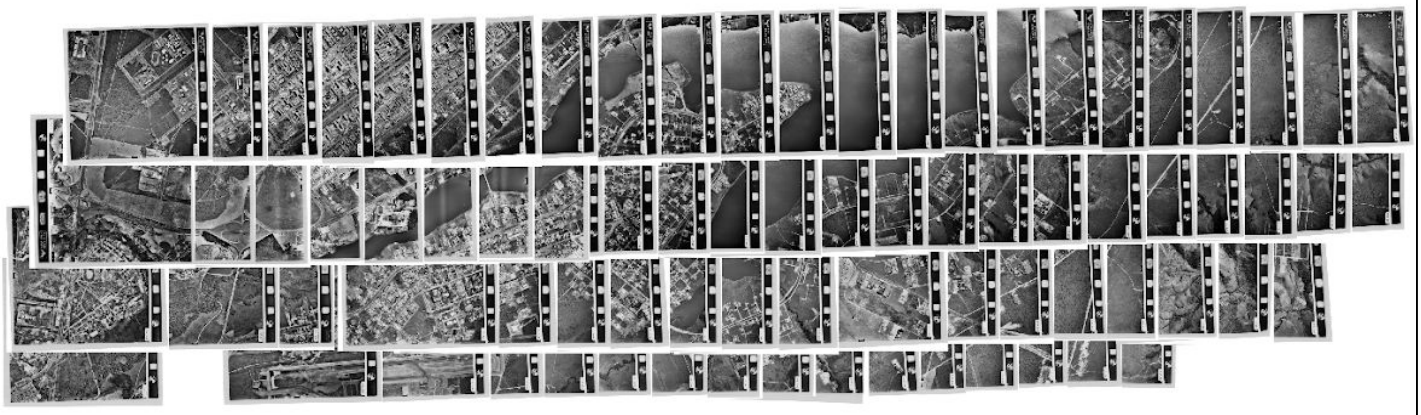
O recobrimento longitudinal é o recobrimento dentro de uma faixa de vôo



O recobrimento lateral é o recobrimento entre as faixas de vôo

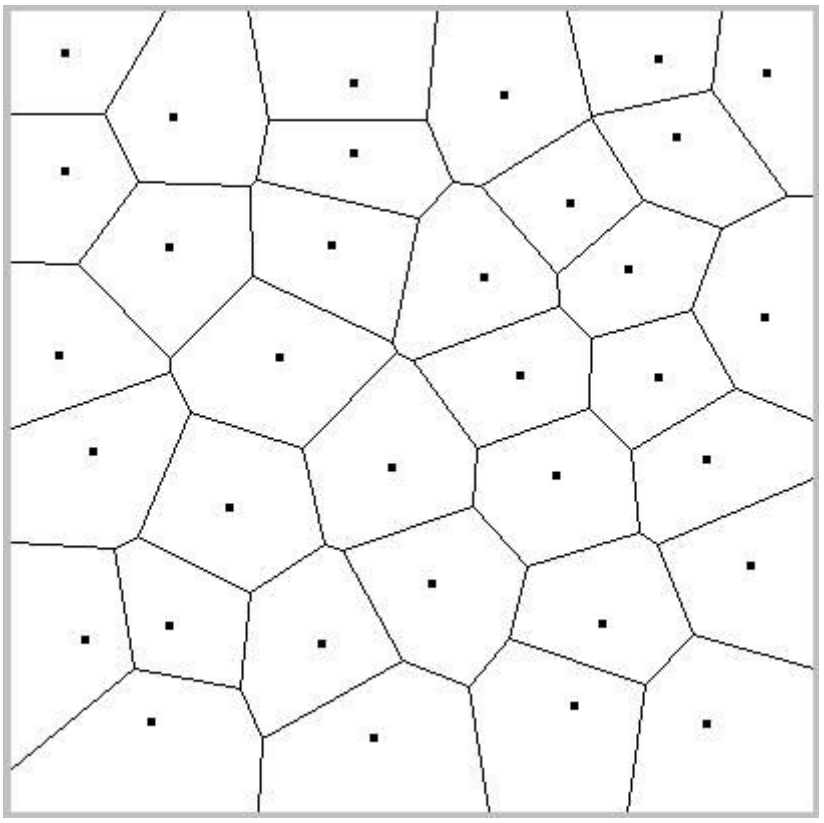



O problema do recobrimento é que quando as fotos são georreferenciadas as sobreposições das bordas impedem a visualização do cena e é necessário efetuar os cortes em todas as fotografias antes da mosaicagem.
A seguir temos como exemplo um conjunto de aerofotos da região de Brasília, do ano de 1975, nele é possível ver o “problema” causado pela recobrimento.



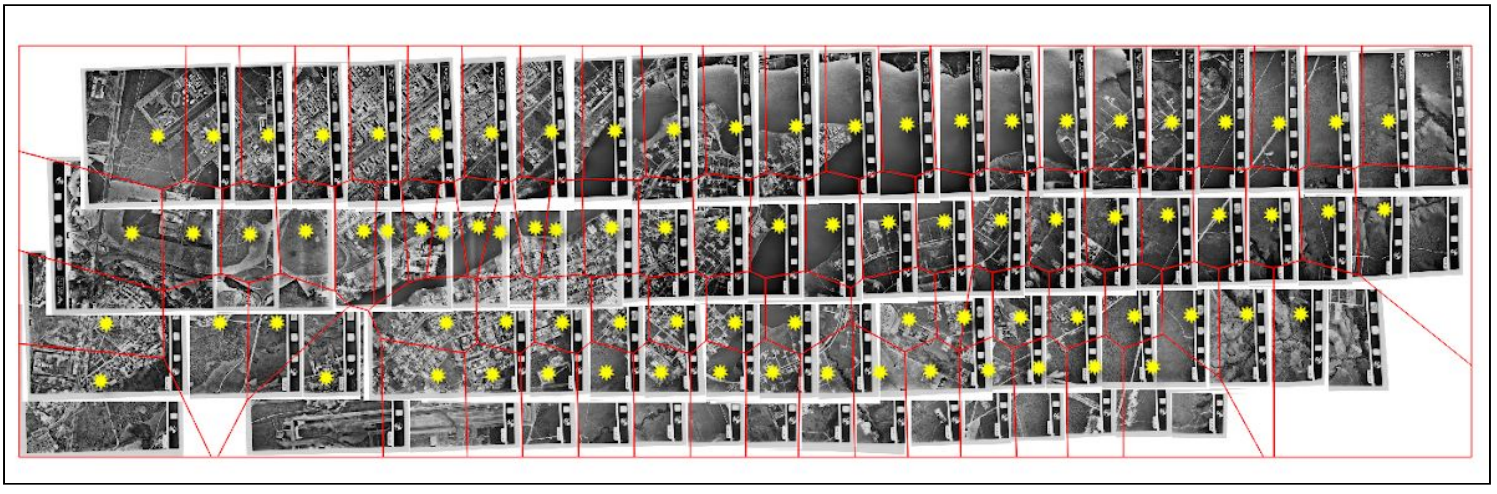
Para resolver o problema do corte e também qual parte de cada imagem utilizar no mosaico partimos de uma característica da aerofotos, as aerofotos possuem uma perspectiva de ponto central. A principal implicação disso é que no centro da uma foto a distorção é zero(ortogonalidade) e conforme se encaminha para as bordas a distorção aumenta.
Partindo desse princípio o ideal é que se corte as imagens de forma a utilizar as partes mais próximas ao centro de cada imagem, assim se usará no mosaico os pixels menos distorcidos.

O problema apresentado dessa forma pode ser resolvido pela utilização do Diagrama de Voronoi.

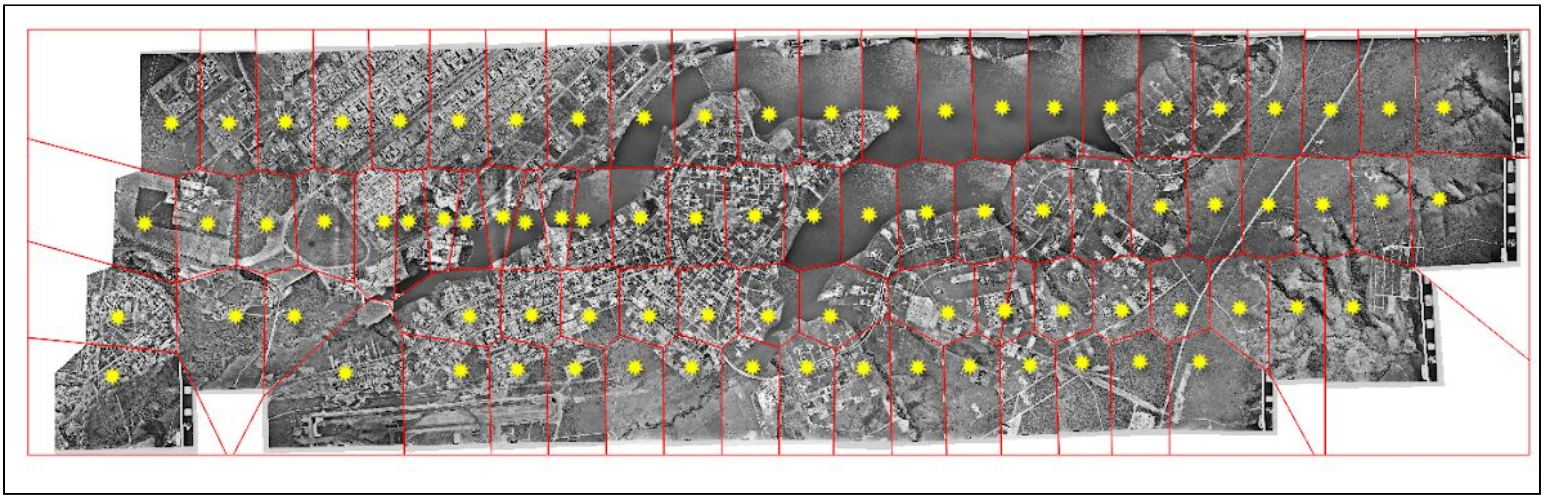


Informalmente o diagrama de voronoi é a subdivisão de um plano em regiões formadas pelos lugares mais próximos ao conjunto de pontos geradores. Para o nosso caso, esse conjunto de pontos geradores são os centros de cada fotografia. Como todas as fotografias estão georreferenciadas é possível automatizar todo o processo.

Abaixo é apresentado o diagrama de voronoi calculado para o conjunto de aerofotos de 1975.



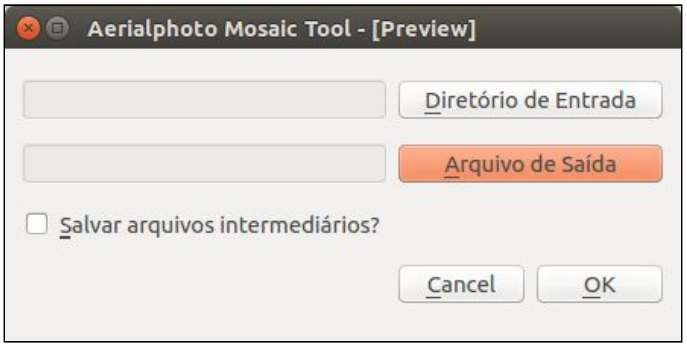
Uma vez que se tenha calculado os centros das imagens e os diagramas de voronoi é necessário cortar as imagens utilizando suas respectivas células de voronoi e depois executar a mosaicagem.



Para executar todo esse procedimento automatizado, foi desenvolvido um plugin para o software QGIS 3.0.

O código fonte do plugin está disponível no GitHub através do seguinte link:
https://github.com/dmcarvalho/aerialphoto_mosaic_tool
Para utilizar o plugin, basta baixar o código fonte no link acima e copiá-lo para o diretório:
UNIX/Mac: /home/(user)/.local/share/QGIS/QGIS3/profiles/default/python/plugins/

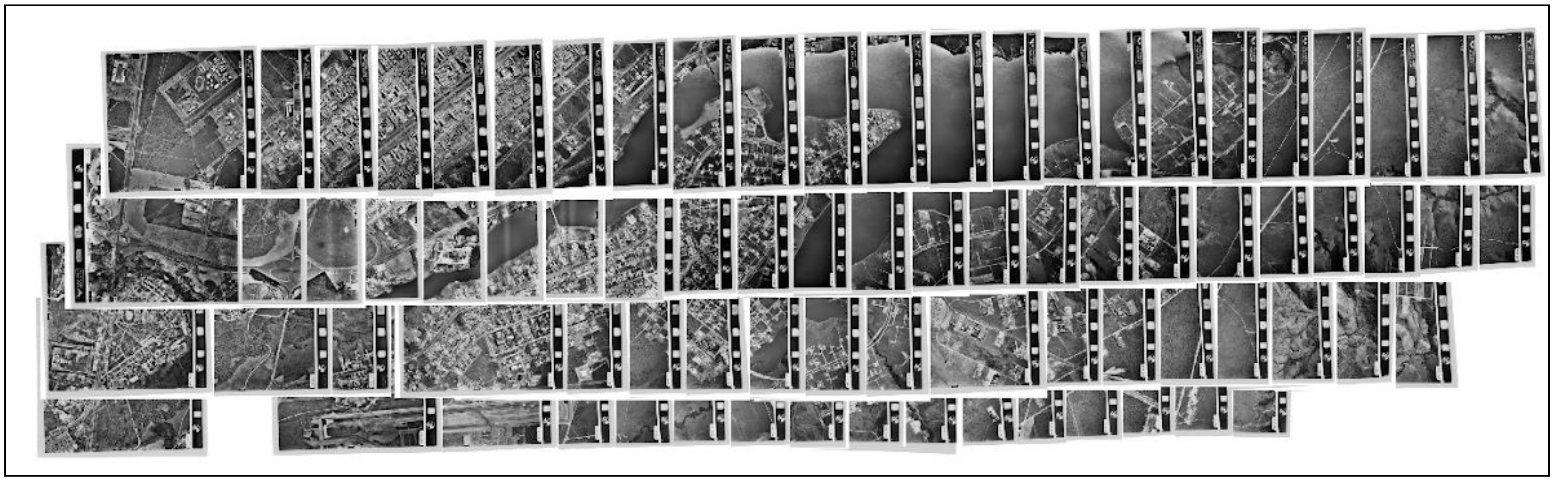
A tela do plugin é a seguinte:

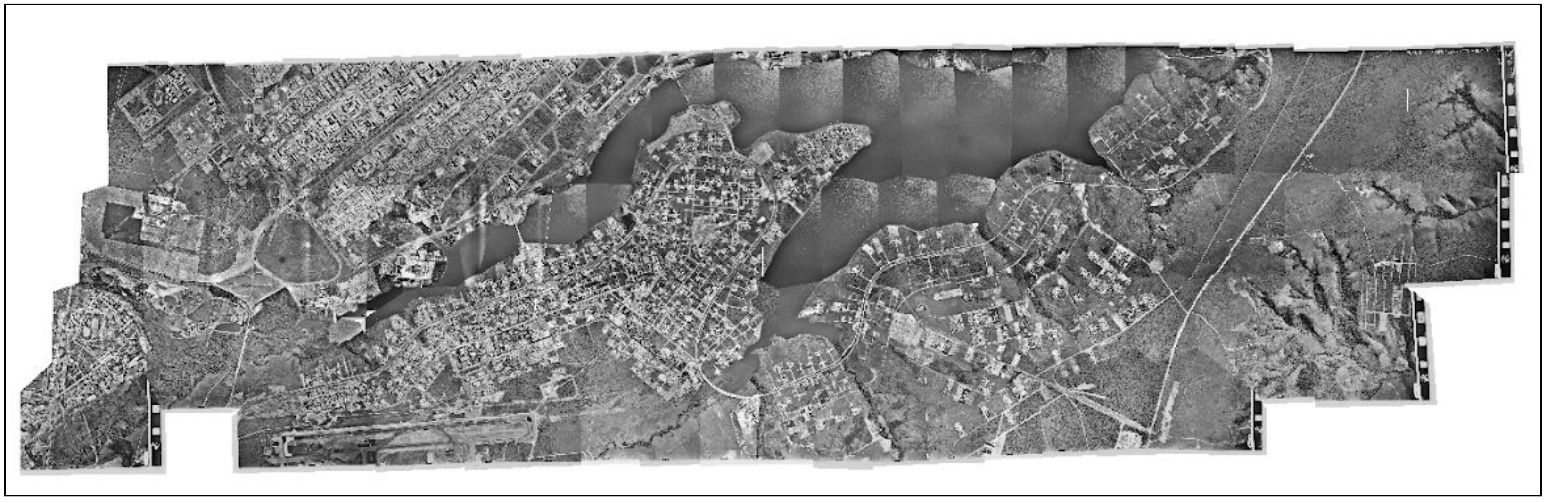



Os parâmetros utilizados na Tela do plugin são os seguintes:
Diretório de Entrada - Diretório onde está o conjunto de aerofotos.

Arquivo de Saída - Nome do arquivo de saída.
Salvar arquivos intermediários - Ao selecionar essa opção os arquivos intermediários (Centro das Fotografias, Diagrama de Voronoi e Fotografias recortadas) serão salvos na mesma local do arquivo de saída em uma pasta chamada arquivos_intermediarios.

Abaixo é possível comparar o resultado do processamento.





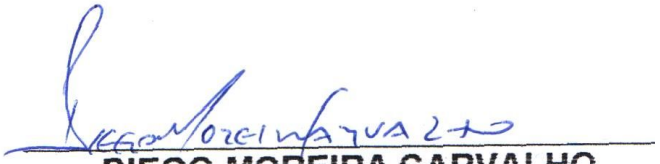
Esse método se mostra eficiente para a construção de mosaicos a partir de fotografias georreferenciadas, todavia apresenta a limitação de não se aplicar nenhuma correção de histograma e principalmente o fato de se haver a necessidade de ter todas as fotografias georreferenciadas.

# Conclusão

Essa consultoria considera que o trabalho conjunto com a equipe da CGCIG foi o principal fator de sucesso para o entendimento dos problemas e construção das soluções apresentadas nesse relatório.

Sugerimos como possibilidade de avanço, principalmente no que tange a construção de mosaicos de fotografias aéreas a construção de uma ferramenta que tenha como base a identificação de pontos homólogos nas fotografias isso apresentaria um grande avanço pelo motivo de tirar a carga de trabalho de se georreferenciar todas as fotografias. O processo de trabalho seria a criação do mosaico e depois seu georreferenciamento, diminuindo o tempo de execução e a quantidade de trabalho a ser realizado.

Brasília,13 de novembro de 2018



DIEGO MOREIRA CARVALHO

# ANEXO 1 - MODELO DE DADOS GPC

```
-- Database generated with pgModeler (PostgreSQL Database Modeler).
-- pgModeler  version: 0.9.0
-- PostgreSQL version: 9.6
-- Project Site: pgmodeler.com.br
-- Model Author: Diego Moreira Carvalho


-- Database creation must be done outside an multicommand file.
-- These commands were put in this file only for convenience.
-- -- object: gpc | type: DATABASE --
-- -- DROP DATABASE IF EXISTS gpc;
-- CREATE DATABASE gpc
-- ;
-- -- ddl-end --
--

-- object: metadado | type: SCHEMA --
-- DROP SCHEMA IF EXISTS metadado CASCADE;
CREATE SCHEMA metadado;
-- ddl-end --
ALTER SCHEMA metadado OWNER TO postgres;
-- ddl-end --

-- object: linha_producao | type: SCHEMA --
-- DROP SCHEMA IF EXISTS linha_producao CASCADE;
CREATE SCHEMA linha_producao;
-- ddl-end --
ALTER SCHEMA linha_producao OWNER TO postgres;
-- ddl-end --

-- object: gerenciamento_producao | type: SCHEMA --
-- DROP SCHEMA IF EXISTS gerenciamento_producao CASCADE;
CREATE SCHEMA gerenciamento_producao;
-- ddl-end --
ALTER SCHEMA gerenciamento_producao OWNER TO postgres;
-- ddl-end --

-- object: pessoal | type: SCHEMA --
-- DROP SCHEMA IF EXISTS pessoal CASCADE;
CREATE SCHEMA pessoal;
-- ddl-end --
ALTER SCHEMA pessoal OWNER TO postgres;
-- ddl-end --

SET search_path TO pg_catalog,public,metadado,linha_producao,gerenciamento_producao,pessoal;
-- ddl-end --

-- object: postgis | type: EXTENSION --
-- DROP EXTENSION IF EXISTS postgis CASCADE;
CREATE EXTENSION postgis
        WITH SCHEMA public;
-- ddl-end --

-- object: metadado.citacao | type: TABLE --
-- DROP TABLE IF EXISTS metadado.citacao CASCADE;
CREATE TABLE metadado.citacao(
    id serial NOT NULL,
    identificacao smallint NOT NULL,
    data date,
    tipo_de_data character varying(3) NOT NULL,
    edicao text,
    series text,
    isbn text,
    CONSTRAINT citacao_pk PRIMARY KEY (id)
```

```
);
-- ddl-end --
ALTER TABLE metadado.citacao OWNER TO postgres;
-- ddl-end --

-- object: linha_producao.etapa_de_trabalho | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.etapa_de_trabalho CASCADE;
CREATE TABLE linha_producao.etapa_de_trabalho(
    id serial NOT NULL,
    nome character varying(250) NOT NULL,
    manual_de_instrucao smallint,
    template_relatorio smallint,
    id_status_produto smallint,
    CONSTRAINT etapa_de_trabalho_pk PRIMARY KEY (id)

);
-- ddl-end --
COMMENT ON COLUMN linha_producao.etapa_de_trabalho.manual_de_instrucao IS 'caminho para o arquivo em disco';
-- ddl-end --
ALTER TABLE linha_producao.etapa_de_trabalho OWNER TO postgres;
-- ddl-end --

-- object: gerenciamento_producao.conjunto_inventariar | type: TABLE --
-- DROP TABLE IF EXISTS gerenciamento_producao.conjunto_inventariar CASCADE;
CREATE TABLE gerenciamento_producao.conjunto_inventariar(
    id smallint NOT NULL,
    descricao text NOT NULL,
    CONSTRAINT conjunto_inventariar_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE gerenciamento_producao.conjunto_inventariar OWNER TO postgres;
-- ddl-end --

-- object: linha_producao.tipo_artefato | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.tipo_artefato CASCADE;
CREATE TABLE linha_producao.tipo_artefato(
    id serial NOT NULL,
    nome character varying(250) NOT NULL,
    formato_arquivo smallint NOT NULL,
    CONSTRAINT artefato_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE linha_producao.tipo_artefato OWNER TO postgres;
-- ddl-end --

-- object: linha_producao.formato_arquivo | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.formato_arquivo CASCADE;
CREATE TABLE linha_producao.formato_arquivo(
    id serial NOT NULL,
    nome character varying(250) NOT NULL,
    descricao text NOT NULL,
    CONSTRAINT formato_arquivo_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE linha_producao.formato_arquivo OWNER TO postgres;
-- ddl-end --

-- object: linha_producao.extensao | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.extensao CASCADE;
CREATE TABLE linha_producao.extensao(
    id serial NOT NULL,
    nome character varying(250) NOT NULL,
    descricao character varying,
    CONSTRAINT extensao_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE linha_producao.extensao OWNER TO postgres;
```

```
-- ddl-end --

-- object: linha_producao.formato_arquivo_extensao | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.formato_arquivo_extensao CASCADE;
CREATE TABLE linha_producao.formato_arquivo_extensao(
    id_formato_arquivo integer NOT NULL,
    id_extensao integer NOT NULL,
    CONSTRAINT formato_arquivo_extensao_pk PRIMARY KEY (id_formato_arquivo,id_extensao)

);
-- ddl-end --

-- object: formato_arquivo_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.formato_arquivo_extensao DROP CONSTRAINT IF EXISTS formato_arquivo_fk CASCADE;
ALTER TABLE linha_producao.formato_arquivo_extensao ADD CONSTRAINT formato_arquivo_fk FOREIGN KEY (id_formato_arquivo)
REFERENCES linha_producao.formato_arquivo (id) MATCH FULL
ON DELETE RESTRICT ON UPDATE CASCADE;
-- ddl-end --

-- object: extensao_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.formato_arquivo_extensao DROP CONSTRAINT IF EXISTS extensao_fk CASCADE;
ALTER TABLE linha_producao.formato_arquivo_extensao ADD CONSTRAINT extensao_fk FOREIGN KEY (id_extensao)
REFERENCES linha_producao.extensao (id) MATCH FULL
ON DELETE RESTRICT ON UPDATE CASCADE;
-- ddl-end --

-- object: linha_producao.template_xform | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.template_xform CASCADE;
CREATE TABLE linha_producao.template_xform(
    id serial NOT NULL,
    nome character varying(250) NOT NULL,
    template bytea NOT NULL,
    autor smallint NOT NULL,
    CONSTRAINT template_xform_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE linha_producao.template_xform OWNER TO postgres;
-- ddl-end --

-- object: linha_producao.tp_artefato | type: TYPE --
-- DROP TYPE IF EXISTS linha_producao.tp_artefato CASCADE;
CREATE TYPE linha_producao.tp_artefato AS
 ENUM ('Artefato Inicial','Artefato Final','Artefato Auxiliar');
-- ddl-end --
ALTER TYPE linha_producao.tp_artefato OWNER TO postgres;
-- ddl-end --

-- object: linha_producao.validacao | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.validacao CASCADE;
CREATE TABLE linha_producao.validacao(
    id serial NOT NULL,
    nome smallint,
    template_relatorio smallint NOT NULL,
    validacao_manual smallint,
    validacao_automatica smallint,
    CONSTRAINT validacao_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE linha_producao.validacao OWNER TO postgres;
-- ddl-end --

-- object: linha_producao.validacao_manual | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.validacao_manual CASCADE;
CREATE TABLE linha_producao.validacao_manual(
    id serial NOT NULL,
    nome character varying(250) NOT NULL,
    manual smallint NOT NULL,
```

```
    CONSTRAINT validacao_manual_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE linha_producao.validacao_manual OWNER TO postgres;
-- ddl-end --

-- object: linha_producao.validacao_automatica | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.validacao_automatica CASCADE;
CREATE TABLE linha_producao.validacao_automatica(
    id serial NOT NULL,
    nome character varying(250) NOT NULL,
    tipo_arquivo smallint NOT NULL,
    script smallint NOT NULL,
    CONSTRAINT validacao_automatica_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE linha_producao.validacao_automatica OWNER TO postgres;
-- ddl-end --

-- object: linha_producao.manual | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.manual CASCADE;
CREATE TABLE linha_producao.manual(
    id serial NOT NULL,
    nome character varying(250) NOT NULL,
    arquivo bytea NOT NULL,
    data_publicacao date NOT NULL,
    autor character varying(11) NOT NULL,
    CONSTRAINT manual_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE linha_producao.manual OWNER TO postgres;
-- ddl-end --

-- object: linha_producao.script | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.script CASCADE;
CREATE TABLE linha_producao.script(
    id serial NOT NULL,
    nome character varying(250) NOT NULL,
    arquivo bytea NOT NULL,
    autor character varying(11) NOT NULL,
    CONSTRAINT script_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE linha_producao.script OWNER TO postgres;
-- ddl-end --

-- object: metadado."CI_DateTypeCode" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."CI_DateTypeCode" CASCADE;
CREATE TABLE metadado."CI_DateTypeCode"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "CI_DateTypeCode_pk" PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado."CI_DateTypeCode" OWNER TO postgres;
-- ddl-end --

-- object: metadado."MD_ProgressCode" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."MD_ProgressCode" CASCADE;
CREATE TABLE metadado."MD_ProgressCode"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "MD_ProgressCode_pk" PRIMARY KEY (domain_code)
```

```
);
-- ddl-end --
ALTER TABLE metadado."MD_ProgressCode" OWNER TO postgres;
-- ddl-end --

-- object: metadado.ponto_de_contato | type: TABLE --
-- DROP TABLE IF EXISTS metadado.ponto_de_contato CASCADE;
CREATE TABLE metadado.ponto_de_contato(
    id serial NOT NULL,
    nome_da_organizacao text NOT NULL,
    funcao character varying(3) NOT NULL,
    CONSTRAINT ponto_de_contato_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.ponto_de_contato OWNER TO postgres;
-- ddl-end --

-- object: metadado."CI_RoleCode" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."CI_RoleCode" CASCADE;
CREATE TABLE metadado."CI_RoleCode"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "CI_RoleCode_pk" PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado."CI_RoleCode" OWNER TO postgres;
-- ddl-end --

-- object: metadado.palavras_chaves_descritivas | type: TABLE --
-- DROP TABLE IF EXISTS metadado.palavras_chaves_descritivas CASCADE;
CREATE TABLE metadado.palavras_chaves_descritivas(
    id serial NOT NULL,
    palavras_chaves text NOT NULL,
    tipo_palavra_chave character varying(3) NOT NULL,
    lexico text,
    identificacao smallint,
    CONSTRAINT palavras_chaves_descritivas_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.palavras_chaves_descritivas OWNER TO postgres;
-- ddl-end --

-- object: metadado."MD_KeywordTypeCode" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."MD_KeywordTypeCode" CASCADE;
CREATE TABLE metadado."MD_KeywordTypeCode"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "MD_KeywordTypeCode_pk" PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado."MD_KeywordTypeCode" OWNER TO postgres;
-- ddl-end --

-- object: metadado.nome_do_arquivo_agregado | type: TABLE --
-- DROP TABLE IF EXISTS metadado.nome_do_arquivo_agregado CASCADE;
CREATE TABLE metadado.nome_do_arquivo_agregado(
    id serial NOT NULL,
    identificacao smallint,
    data date,
    tipo_de_data character varying(3),
    edicao text,
    series text,
    isbn text,
    CONSTRAINT nome_do_arquivo_agregado_pk PRIMARY KEY (id)
```

```
);
-- ddl-end --
ALTER TABLE metadado.nome_do_arquivo_agregado OWNER TO postgres;
-- ddl-end --

-- object: metadado."DS_AssociationTypeCode" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."DS_AssociationTypeCode" CASCADE;
CREATE TABLE metadado."DS_AssociationTypeCode"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "DS_AssociationTypeCode_pk" PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado."DS_AssociationTypeCode" OWNER TO postgres;
-- ddl-end --

-- object: metadado.identificacao | type: TABLE --
-- DROP TABLE IF EXISTS metadado.identificacao CASCADE;
CREATE TABLE metadado.identificacao(
    id serial NOT NULL,
    resumo text NOT NULL,
    objetivo text,
    creditos text,
    status character varying(3) NOT NULL,
    ponto_de_contato smallint NOT NULL,
    pre_visualizacao_grafica text,
    tipo_de_associacao character varying(3),
    CONSTRAINT identificaca_cdg_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.identificacao OWNER TO postgres;
-- ddl-end --

-- object: metadado."MD_SpatialRepresentationTypeCode" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."MD_SpatialRepresentationTypeCode" CASCADE;
CREATE TABLE metadado."MD_SpatialRepresentationTypeCode"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "MD_SpatialRepresentationTypeCode_pk" PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado."MD_SpatialRepresentationTypeCode" OWNER TO postgres;
-- ddl-end --

-- object: metadado.iso_639 | type: TABLE --
-- DROP TABLE IF EXISTS metadado.iso_639 CASCADE;
CREATE TABLE metadado.iso_639(
    domain_code character varying(3) NOT NULL,
    name text,
    CONSTRAINT iso_639_pk PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado.iso_639 OWNER TO postgres;
-- ddl-end --

-- object: metadado.metametadados | type: TABLE --
-- DROP TABLE IF EXISTS metadado.metametadados CASCADE;
CREATE TABLE metadado.metametadados(
    id serial NOT NULL,
    idioma character varying NOT NULL,
    norma_de_codificacao_de_caracteres character varying(3) NOT NULL,
    nivel_hierarquico character varying(3) NOT NULL,
    contato_do_responsavel_pelos_metadados smallint NOT NULL,
    data_dos_metadados date NOT NULL,
    identificador_metadados bigint NOT NULL,
```

```
    nome_da_norma_e_perfil_de_metadados text NOT NULL,
    versao_da_norma_de_metadados integer,
    restricao_de_acesso character varying(3),
    restricoes_de_uso character varying(3),
    identificacao_cdg smallint,
    CONSTRAINT metametadados_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.metametadados OWNER TO postgres;
-- ddl-end --

-- object: metadado."MD_CharacterSetCode" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."MD_CharacterSetCode" CASCADE;
CREATE TABLE metadado."MD_CharacterSetCode"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "MD_CharacterSetCode_pk" PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado."MD_CharacterSetCode" OWNER TO postgres;
-- ddl-end --

-- object: metadado."MD_TopicCategoryCode" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."MD_TopicCategoryCode" CASCADE;
CREATE TABLE metadado."MD_TopicCategoryCode"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "MD_TopicCategoryCode_pk" PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado."MD_TopicCategoryCode" OWNER TO postgres;
-- ddl-end --

-- object: metadado.extensao | type: TABLE --
-- DROP TABLE IF EXISTS metadado.extensao CASCADE;
CREATE TABLE metadado.extensao(
    id serial NOT NULL,
    identificacao_cdg smallint,
    extensao_geografica geometry(POLYGON, 4674) NOT NULL,
    CONSTRAINT extensao_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.extensao OWNER TO postgres;
-- ddl-end --

-- object: metadado.extensao_temporal | type: TABLE --
-- DROP TABLE IF EXISTS metadado.extensao_temporal CASCADE;
CREATE TABLE metadado.extensao_temporal(
    id serial NOT NULL,
    data_hora_inicio timestamp,
    data_hora_termino timestamp,
    CONSTRAINT extensao_temporal_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.extensao_temporal OWNER TO postgres;
-- ddl-end --

-- object: metadado.extensao_altimetrica | type: TABLE --
-- DROP TABLE IF EXISTS metadado.extensao_altimetrica CASCADE;
CREATE TABLE metadado.extensao_altimetrica(
    id serial NOT NULL,
    valor_minimo real,
    valor_maximo real,
    unidades_de_medida text,
```

```
    datum_altimetrico text,
    CONSTRAINT extensao_altimetrica_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.extensao_altimetrica OWNER TO postgres;
-- ddl-end --

-- object: metadado.informacoes_de_restricao | type: TABLE --
-- DROP TABLE IF EXISTS metadado.informacoes_de_restricao CASCADE;
CREATE TABLE metadado.informacoes_de_restricao(
    id serial NOT NULL,
    restricoes_de_acesso character varying(3),
    restricoes_de_uso character varying(3),
    restricoes_de_seguranca character varying(3) NOT NULL,
    CONSTRAINT informacoes_de_restricao_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.informacoes_de_restricao OWNER TO postgres;
-- ddl-end --

-- object: metadado."MD_RestrictionCode" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."MD_RestrictionCode" CASCADE;
CREATE TABLE metadado."MD_RestrictionCode"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "MD_RestrictionCode_pk" PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado."MD_RestrictionCode" OWNER TO postgres;
-- ddl-end --

-- object: metadado."MD_ClassificationCode" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."MD_ClassificationCode" CASCADE;
CREATE TABLE metadado."MD_ClassificationCode"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "MD_ClassificationCode_pk" PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado."MD_ClassificationCode" OWNER TO postgres;
-- ddl-end --

-- object: metadado.qualidade | type: TABLE --
-- DROP TABLE IF EXISTS metadado.qualidade CASCADE;
CREATE TABLE metadado.qualidade(
    id serial NOT NULL,
    nivel_hierarquico character varying(3) NOT NULL,
    CONSTRAINT qualidade_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.qualidade OWNER TO postgres;
-- ddl-end --

-- object: metadado."MD_ScopeCode" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."MD_ScopeCode" CASCADE;
CREATE TABLE metadado."MD_ScopeCode"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "MD_ScopeCode_pk" PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado."MD_ScopeCode" OWNER TO postgres;
```

```
-- ddl-end --

-- object: metadado.linhagem | type: TABLE --
-- DROP TABLE IF EXISTS metadado.linhagem CASCADE;
CREATE TABLE metadado.linhagem(
    id serial NOT NULL,
    qualidade smallint,
    declaracao text NOT NULL,
    fonte_de_dados text,
    etapas_do_processo text,
    declaracao_automatica text NOT NULL,
    fonte_de_dados_automatica text,
    etapas_do_processo_automatica text,
    CONSTRAINT linhagem_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.linhagem OWNER TO postgres;
-- ddl-end --

-- object: metadado.relatorio | type: TABLE --
-- DROP TABLE IF EXISTS metadado.relatorio CASCADE;
CREATE TABLE metadado.relatorio(
    id serial NOT NULL,
    qualidade smallint,
    completude text,
    consistencia_logica text,
    exatidao_posicional decimal,
    exatidao_temporal text,
    exatidao_tematica text,
    CONSTRAINT relatorio_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.relatorio OWNER TO postgres;
-- ddl-end --

-- object: metadado.informacao_de_manutencao | type: TABLE --
-- DROP TABLE IF EXISTS metadado.informacao_de_manutencao CASCADE;
CREATE TABLE metadado.informacao_de_manutencao(
    id serial NOT NULL,
    frequencia_manutencao_e_atualizacao character varying(3),
    CONSTRAINT informacao_de_manutencao_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.informacao_de_manutencao OWNER TO postgres;
-- ddl-end --

-- object: metadado."MD_MaintenanceFrequencyCode" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."MD_MaintenanceFrequencyCode" CASCADE;
CREATE TABLE metadado."MD_MaintenanceFrequencyCode"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "MD_MaintenanceFrequencyCode_pk" PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado."MD_MaintenanceFrequencyCode" OWNER TO postgres;
-- ddl-end --

-- object: metadado.informacao_representacao_espacial | type: TABLE --
-- DROP TABLE IF EXISTS metadado.informacao_representacao_espacial CASCADE;
CREATE TABLE metadado.informacao_representacao_espacial(
    id serial NOT NULL,
    CONSTRAINT informacao_representacao_espacial_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.informacao_representacao_espacial OWNER TO postgres;
```

```
-- ddl-end --

-- object: metadado.representacao_espacial_vetorial | type: TABLE --
-- DROP TABLE IF EXISTS metadado.representacao_espacial_vetorial CASCADE;
CREATE TABLE metadado.representacao_espacial_vetorial(
    id serial NOT NULL,
    informacao_representacao_espacial smallint,
    nivel_topologico character varying(3),
    tipo_dos_objetos_geometricos character varying(3),
    CONSTRAINT representacao_espacial_vetorial_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.representacao_espacial_vetorial OWNER TO postgres;
-- ddl-end --

-- object: metadado."MD_TopologyLevelCode" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."MD_TopologyLevelCode" CASCADE;
CREATE TABLE metadado."MD_TopologyLevelCode"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "MD_TopologyLevelCode_pk" PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado."MD_TopologyLevelCode" OWNER TO postgres;
-- ddl-end --

-- object: metadado."MD_GeometricObjectTypeCode" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."MD_GeometricObjectTypeCode" CASCADE;
CREATE TABLE metadado."MD_GeometricObjectTypeCode"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "MD_GeometricObjectTypeCode_pk" PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado."MD_GeometricObjectTypeCode" OWNER TO postgres;
-- ddl-end --

-- object: metadado.representacao_espacial_matricial | type: TABLE --
-- DROP TABLE IF EXISTS metadado.representacao_espacial_matricial CASCADE;
CREATE TABLE metadado.representacao_espacial_matricial(
    id serial NOT NULL,
    informacao_representacao_espacial smallint,
    CONSTRAINT representacao_espacial_matricial_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.representacao_espacial_matricial OWNER TO postgres;
-- ddl-end --

-- object: metadado.representacao_espacial_matricial_georretificada | type: TABLE --
-- DROP TABLE IF EXISTS metadado.representacao_espacial_matricial_georretificada CASCADE;
CREATE TABLE metadado.representacao_espacial_matricial_georretificada(
    id serial NOT NULL,
    representacao_espacial_matricial smallint,
    disponibilidade_dos_pontos_de_verificacao boolean,
    descricao_dos_pontos_de_verificacao text,
    referencia_no_pixel character varying(3),
    CONSTRAINT representacao_espacial_matricial_georretificada_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.representacao_espacial_matricial_georretificada OWNER TO postgres;
-- ddl-end --

-- object: metadado."MD_PixelOrientationCode" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."MD_PixelOrientationCode" CASCADE;
```

```
CREATE TABLE metadado."MD_PixelOrientationCode"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "MD_PixelOrientationCode_pk" PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado."MD_PixelOrientationCode" OWNER TO postgres;
-- ddl-end --

-- object: metadado.representacao_espacial_matricial_georreferenciavel | type: TABLE --
-- DROP TABLE IF EXISTS metadado.representacao_espacial_matricial_georreferenciavel CASCADE;
CREATE TABLE metadado.representacao_espacial_matricial_georreferenciavel(
    id smallint NOT NULL,
    representacao_espacial_matricial smallint,
    disponibilidade_dos_pontos_de_controle boolean,
    disponibilidade_dos_parametros_de_orientacao boolean,
    parametros_georreferenciaveis text,
    CONSTRAINT representacao_espacial_matricial_georreferenciavel_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.representacao_espacial_matricial_georreferenciavel OWNER TO postgres;
-- ddl-end --

-- object: metadado.sistema_de_referencia | type: TABLE --
-- DROP TABLE IF EXISTS metadado.sistema_de_referencia CASCADE;
CREATE TABLE metadado.sistema_de_referencia(
    id serial NOT NULL,
    identificador_do_sistema_de_referencia text,
    projecao text,
    elipsoide text,
    datum text,
    parametros_do_elipsoide text,
    parametros_da_projecao smallint,
    identicacao_cdg smallint,
    CONSTRAINT sistema_de_referencia_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.sistema_de_referencia OWNER TO postgres;
-- ddl-end --

-- object: metadado.informacao_de_conteudo | type: TABLE --
-- DROP TABLE IF EXISTS metadado.informacao_de_conteudo CASCADE;
CREATE TABLE metadado.informacao_de_conteudo(
    id serial NOT NULL,
    CONSTRAINT informacao_de_conteudo_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.informacao_de_conteudo OWNER TO postgres;
-- ddl-end --

-- object: metadado.descricao_catalogo_de_feicoes | type: TABLE --
-- DROP TABLE IF EXISTS metadado.descricao_catalogo_de_feicoes CASCADE;
CREATE TABLE metadado.descricao_catalogo_de_feicoes(
    id serial NOT NULL,
    informacao_de_conteudo smallint,
    catalogo_incluido_no_cdg boolean,
    citacao_catalogo_de_feicoes character varying(3),
    CONSTRAINT descricao_catalogo_de_feicoes_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.descricao_catalogo_de_feicoes OWNER TO postgres;
-- ddl-end --

-- object: metadado."CI_Citation" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."CI_Citation" CASCADE;
```

```
CREATE TABLE metadado."CI_Citation"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "CI_Citation_pk" PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado."CI_Citation" OWNER TO postgres;
-- ddl-end --

-- object: metadado.descricao_dados_matriciais | type: TABLE --
-- DROP TABLE IF EXISTS metadado.descricao_dados_matriciais CASCADE;
CREATE TABLE metadado.descricao_dados_matriciais(
    id serial NOT NULL,
    informacao_de_conteudo smallint,
    descricao_do_conteudo_da_particao_pixel text,
    tipo_da_informacao_representada_pelo_valor_do_pixel character varying(3),
    descricao_da_imagem boolean,
    banda_espectral boolean,
    descricao_do_atributo text,
    informacao_sobre_calibracao_da_camera text,
    dimensao text,
    cobertura_de_nuvem text,
    CONSTRAINT descricao_dados_matriciais_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.descricao_dados_matriciais OWNER TO postgres;
-- ddl-end --

-- object: metadado."MD_CoverageContentTypeCode" | type: TABLE --
-- DROP TABLE IF EXISTS metadado."MD_CoverageContentTypeCode" CASCADE;
CREATE TABLE metadado."MD_CoverageContentTypeCode"(
    domain_code character varying(3) NOT NULL,
    name character varying(100) NOT NULL,
    definition text NOT NULL,
    CONSTRAINT "MD_CoverageContentTypeCode_pk" PRIMARY KEY (domain_code)

);
-- ddl-end --
ALTER TABLE metadado."MD_CoverageContentTypeCode" OWNER TO postgres;
-- ddl-end --

-- object: metadado.banda_espectral | type: TABLE --
-- DROP TABLE IF EXISTS metadado.banda_espectral CASCADE;
CREATE TABLE metadado.banda_espectral(
    id serial NOT NULL,
    nome_da_banda text,
    bits_por_pixel text,
    descricao_dados_matriciais smallint,
    CONSTRAINT banda_espectral_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE metadado.banda_espectral OWNER TO postgres;
-- ddl-end --

-- object: metadado.distribuicao | type: TABLE --
-- DROP TABLE IF EXISTS metadado.distribuicao CASCADE;
CREATE TABLE metadado.distribuicao(
    id serial NOT NULL,
    nome_do_formato text NOT NULL,
    versao_do_formato text NOT NULL,
    acesso_online text,
    acesso_offline text,
    contato_do_distribuidor smallint NOT NULL,
    CONSTRAINT distribuicao_pk PRIMARY KEY (id)

);
-- ddl-end --
```

```
ALTER TABLE metadado.distribuicao OWNER TO postgres;
-- ddl-end --

-- object: metadado.identificacao_cdg | type: TABLE --
-- DROP TABLE IF EXISTS metadado.identificacao_cdg CASCADE;
CREATE TABLE metadado.identificacao_cdg(
    tipo_representacao_espacial character varying(3),
    denominador_escala_equivalente integer,
    idioma character varying(3) NOT NULL,
    norma_codificacao_de_caracteres character varying(3),
    categoria_tematica character varying(3),
    ambiente_de_producao text,
    id_metametadados integer,
    id_informacao_de_manutencao integer,
    id_informacao_representacao_espacial integer,
    id_informacao_de_conteudo integer,
    id_sistema_de_referencia integer,
    id_informacoes_de_restricao integer,
    id_distribuicao integer,
    id_qualidade integer,
--     id integer NOT NULL,
--     resumo text NOT NULL,
--     objetivo text,
--     creditos text,
--     status character varying(3) NOT NULL,
--     ponto_de_contato smallint NOT NULL,
--     pre_visualizacao_grafica text,
--     tipo_de_associacao character varying(3),
    CONSTRAINT identificacao_cdg_pk PRIMARY KEY (id)

) INHERITS(metadado.identificacao)
;
-- ddl-end --
ALTER TABLE metadado.identificacao_cdg OWNER TO postgres;
-- ddl-end --

-- object: metametadados_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS metametadados_fk CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT metametadados_fk FOREIGN KEY (id_metametadados)
REFERENCES metadado.metametadados (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: identificacao_cdg_uq | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS identificacao_cdg_uq CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT identificacao_cdg_uq UNIQUE (id_metametadados);
-- ddl-end --

-- object: informacao_de_manutencao_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS informacao_de_manutencao_fk CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT informacao_de_manutencao_fk FOREIGN KEY
(id_informacao_de_manutencao)
REFERENCES metadado.informacao_de_manutencao (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: identificacao_cdg_uq1 | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS identificacao_cdg_uq1 CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT identificacao_cdg_uq1 UNIQUE
(id_informacao_de_manutencao);
-- ddl-end --

-- object: informacao_representacao_espacial_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS informacao_representacao_espacial_fk
CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT informacao_representacao_espacial_fk FOREIGN KEY
(id_informacao_representacao_espacial)
REFERENCES metadado.informacao_representacao_espacial (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --
```

```
-- object: identificacao_cdg_uq2 | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS identificacao_cdg_uq2 CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT identificacao_cdg_uq2 UNIQUE (id_informacao_representacao_espacial);
-- ddl-end --

-- object: informacao_de_conteudo_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS informacao_de_conteudo_fk CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT informacao_de_conteudo_fk FOREIGN KEY (id_informacao_de_conteudo)
REFERENCES metadado.informacao_de_conteudo (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: identificacao_cdg_uq3 | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS identificacao_cdg_uq3 CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT identificacao_cdg_uq3 UNIQUE (id_informacao_de_conteudo);
-- ddl-end --

-- object: sistema_de_referencia_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS sistema_de_referencia_fk CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT sistema_de_referencia_fk FOREIGN KEY (id_sistema_de_referencia)
REFERENCES metadado.sistema_de_referencia (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: identificacao_cdg_uq4 | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS identificacao_cdg_uq4 CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT identificacao_cdg_uq4 UNIQUE (id_sistema_de_referencia);
-- ddl-end --

-- object: informacoes_de_restricao_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS informacoes_de_restricao_fk CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT informacoes_de_restricao_fk FOREIGN KEY (id_informacoes_de_restricao)
REFERENCES metadado.informacoes_de_restricao (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: identificacao_cdg_uq5 | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS identificacao_cdg_uq5 CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT identificacao_cdg_uq5 UNIQUE (id_informacoes_de_restricao);
-- ddl-end --

-- object: distribuicao_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS distribuicao_fk CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT distribuicao_fk FOREIGN KEY (id_distribuicao)
REFERENCES metadado.distribuicao (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: identificacao_cdg_uq6 | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS identificacao_cdg_uq6 CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT identificacao_cdg_uq6 UNIQUE (id_distribuicao);
-- ddl-end --

-- object: qualidade_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS qualidade_fk CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT qualidade_fk FOREIGN KEY (id_qualidade)
REFERENCES metadado.qualidade (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: identificacao_cdg_uq7 | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS identificacao_cdg_uq7 CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT identificacao_cdg_uq7 UNIQUE (id_qualidade);
-- ddl-end --
```

```
-- object: pessoal.profissional | type: TABLE --
-- DROP TABLE IF EXISTS pessoal.profissional CASCADE;
CREATE TABLE pessoal.profissional(
    cpf character varying(11) NOT NULL,
    nome character varying NOT NULL,
    CONSTRAINT profissional_pk PRIMARY KEY (cpf)

);
-- ddl-end --
ALTER TABLE pessoal.profissional OWNER TO postgres;
-- ddl-end --

-- object: pessoal.papel | type: TABLE --
-- DROP TABLE IF EXISTS pessoal.papel CASCADE;
CREATE TABLE pessoal.papel(
    id smallint NOT NULL,
    nome character varying(200) NOT NULL,
    CONSTRAINT papel_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE pessoal.papel OWNER TO postgres;
-- ddl-end --

-- object: linha_producao.tipo_produto | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.tipo_produto CASCADE;
CREATE TABLE linha_producao.tipo_produto(
    id smallint NOT NULL,
    status_final smallint NOT NULL,
    nome character varying(200) NOT NULL,
    CONSTRAINT tipo_produto_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE linha_producao.tipo_produto OWNER TO postgres;
-- ddl-end --

-- object: linha_producao.status_produto | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.status_produto CASCADE;
CREATE TABLE linha_producao.status_produto(
    id smallint NOT NULL,
    nome character varying(200) NOT NULL,
    CONSTRAINT status_produto_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE linha_producao.status_produto OWNER TO postgres;
-- ddl-end --

-- object: linha_producao.status_requeridos | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.status_requeridos CASCADE;
CREATE TABLE linha_producao.status_requeridos(
    id smallint NOT NULL,
    status smallint NOT NULL,
    status_requerido smallint NOT NULL,
    CONSTRAINT status_requeridos_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE linha_producao.status_requeridos OWNER TO postgres;
-- ddl-end --

-- object: status_produto_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.etapa_de_trabalho DROP CONSTRAINT IF EXISTS status_produto_fk CASCADE;
ALTER TABLE linha_producao.etapa_de_trabalho ADD CONSTRAINT status_produto_fk FOREIGN KEY (id_status_produto)
REFERENCES linha_producao.status_produto (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: linha_producao.etapa_de_trabalho_artefato | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.etapa_de_trabalho_artefato CASCADE;
```

```
CREATE TABLE linha_producao.etapa_de_trabalho_artefato(
    id_etapa_de_trabalho integer NOT NULL,
    id_tipo_artefato integer NOT NULL,
    tp_artefato linha_producao.tp_artefato NOT NULL,
    CONSTRAINT etapa_de_trabalho_artefato_pk PRIMARY KEY (id_etapa_de_trabalho,id_tipo_artefato)

);
-- ddl-end --

-- object: etapa_de_trabalho_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.etapa_de_trabalho_artefato DROP CONSTRAINT IF EXISTS etapa_de_trabalho_fk CASCADE;
ALTER TABLE linha_producao.etapa_de_trabalho_artefato ADD CONSTRAINT etapa_de_trabalho_fk FOREIGN KEY (id_etapa_de_trabalho)
REFERENCES linha_producao.etapa_de_trabalho (id) MATCH FULL
ON DELETE RESTRICT ON UPDATE CASCADE;
-- ddl-end --

-- object: tipo_artefato_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.etapa_de_trabalho_artefato DROP CONSTRAINT IF EXISTS tipo_artefato_fk CASCADE;
ALTER TABLE linha_producao.etapa_de_trabalho_artefato ADD CONSTRAINT tipo_artefato_fk FOREIGN KEY (id_tipo_artefato)
REFERENCES linha_producao.tipo_artefato (id) MATCH FULL
ON DELETE RESTRICT ON UPDATE CASCADE;
-- ddl-end --

-- object: linha_producao.many_validacao_has_many_artefato | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.many_validacao_has_many_artefato CASCADE;
CREATE TABLE linha_producao.many_validacao_has_many_artefato(
    id_validacao integer NOT NULL,
    id_tipo_artefato integer NOT NULL,
    CONSTRAINT many_validacao_has_many_artefato_pk PRIMARY KEY (id_validacao,id_tipo_artefato)

);
-- ddl-end --

-- object: validacao_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.many_validacao_has_many_artefato DROP CONSTRAINT IF EXISTS validacao_fk CASCADE;
ALTER TABLE linha_producao.many_validacao_has_many_artefato ADD CONSTRAINT validacao_fk FOREIGN KEY (id_validacao)
REFERENCES linha_producao.validacao (id) MATCH FULL
ON DELETE RESTRICT ON UPDATE CASCADE;
-- ddl-end --

-- object: tipo_artefato_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.many_validacao_has_many_artefato DROP CONSTRAINT IF EXISTS tipo_artefato_fk CASCADE;
ALTER TABLE linha_producao.many_validacao_has_many_artefato ADD CONSTRAINT tipo_artefato_fk FOREIGN KEY (id_tipo_artefato)
REFERENCES linha_producao.tipo_artefato (id) MATCH FULL
ON DELETE RESTRICT ON UPDATE CASCADE;
-- ddl-end --

-- object: pessoal.many_profissional_has_many_papel | type: TABLE --
-- DROP TABLE IF EXISTS pessoal.many_profissional_has_many_papel CASCADE;
CREATE TABLE pessoal.many_profissional_has_many_papel(
    cpf_profissional character varying(11) NOT NULL,
    id_papel smallint NOT NULL,
    CONSTRAINT many_profissional_has_many_papel_pk PRIMARY KEY (cpf_profissional,id_papel)

);
-- ddl-end --

-- object: profissional_fk | type: CONSTRAINT --
-- ALTER TABLE pessoal.many_profissional_has_many_papel DROP CONSTRAINT IF EXISTS profissional_fk CASCADE;
ALTER TABLE pessoal.many_profissional_has_many_papel ADD CONSTRAINT profissional_fk FOREIGN KEY (cpf_profissional)
REFERENCES pessoal.profissional (cpf) MATCH FULL
ON DELETE RESTRICT ON UPDATE CASCADE;
-- ddl-end --
```

```
-- object: papel_fk | type: CONSTRAINT --
-- ALTER TABLE pessoal.many_profissional_has_many_papel DROP CONSTRAINT IF EXISTS papel_fk CASCADE;
ALTER TABLE pessoal.many_profissional_has_many_papel ADD CONSTRAINT papel_fk FOREIGN KEY (id_papel)
REFERENCES pessoal.papel (id) MATCH FULL
ON DELETE RESTRICT ON UPDATE CASCADE;
-- ddl-end --

-- object: pessoal.many_papel_has_many_etapa_de_trabalho | type: TABLE --
-- DROP TABLE IF EXISTS pessoal.many_papel_has_many_etapa_de_trabalho CASCADE;
CREATE TABLE pessoal.many_papel_has_many_etapa_de_trabalho(
    id_papel smallint NOT NULL,
    id_etapa_de_trabalho integer NOT NULL,
    CONSTRAINT many_papel_has_many_etapa_de_trabalho_pk PRIMARY KEY (id_papel,id_etapa_de_trabalho)

);
-- ddl-end --

-- object: papel_fk | type: CONSTRAINT --
-- ALTER TABLE pessoal.many_papel_has_many_etapa_de_trabalho DROP CONSTRAINT IF EXISTS papel_fk CASCADE;
ALTER TABLE pessoal.many_papel_has_many_etapa_de_trabalho ADD CONSTRAINT papel_fk FOREIGN KEY (id_papel)
REFERENCES pessoal.papel (id) MATCH FULL
ON DELETE RESTRICT ON UPDATE CASCADE;
-- ddl-end --

-- object: etapa_de_trabalho_fk | type: CONSTRAINT --
-- ALTER TABLE pessoal.many_papel_has_many_etapa_de_trabalho DROP CONSTRAINT IF EXISTS
etapa_de_trabalho_fk CASCADE;
ALTER TABLE pessoal.many_papel_has_many_etapa_de_trabalho ADD CONSTRAINT etapa_de_trabalho_fk FOREIGN
KEY (id_etapa_de_trabalho)
REFERENCES linha_producao.etapa_de_trabalho (id) MATCH FULL
ON DELETE RESTRICT ON UPDATE CASCADE;
-- ddl-end --

-- object: gerenciamento_producao.produto_cartografico | type: TABLE --
-- DROP TABLE IF EXISTS gerenciamento_producao.produto_cartografico CASCADE;
CREATE TABLE gerenciamento_producao.produto_cartografico(
    id integer NOT NULL,
    conjunto smallint NOT NULL,
    tipo_produto smallint NOT NULL,
    status_inicial smallint,
    id_identificacao_cdg integer,
    id_conjunto_inventariar smallint,
    CONSTRAINT produto_cartografico_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE gerenciamento_producao.produto_cartografico OWNER TO postgres;
-- ddl-end --

-- object: gerenciamento_producao.tarefa | type: TABLE --
-- DROP TABLE IF EXISTS gerenciamento_producao.tarefa CASCADE;
CREATE TABLE gerenciamento_producao.tarefa(
    id smallint NOT NULL,
    "data_criação" date NOT NULL,
    descricao text NOT NULL,
    produto smallint NOT NULL,
    executor character varying(11) NOT NULL,
    prazo_execucao date NOT NULL,
    validador character varying(11) NOT NULL,
    prazo_validacao smallint,
    id_relatorio_tarefa smallint,
    CONSTRAINT tarefa_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE gerenciamento_producao.tarefa OWNER TO postgres;
-- ddl-end --

-- object: gerenciamento_producao.historico_status | type: TABLE --
-- DROP TABLE IF EXISTS gerenciamento_producao.historico_status CASCADE;
```

```
CREATE TABLE gerenciamento_producao.historico_status(
    id smallint NOT NULL,
    produto smallint NOT NULL,
    tarefa smallint NOT NULL,
    status smallint NOT NULL,
    data smallint NOT NULL,
    CONSTRAINT historico_status_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE gerenciamento_producao.historico_status OWNER TO postgres;
-- ddl-end --

-- object: gerenciamento_producao.relatorio_tarefa | type: TABLE --
-- DROP TABLE IF EXISTS gerenciamento_producao.relatorio_tarefa CASCADE;
CREATE TABLE gerenciamento_producao.relatorio_tarefa(
    id smallint NOT NULL,
    conteudo json NOT NULL,
    id_template_xform integer,
    data_entrega date NOT NULL,
    CONSTRAINT relatorio_tarefa_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE gerenciamento_producao.relatorio_tarefa OWNER TO postgres;
-- ddl-end --

-- object: relatorio_tarefa_fk | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.tarefa DROP CONSTRAINT IF EXISTS relatorio_tarefa_fk CASCADE;
ALTER TABLE gerenciamento_producao.tarefa ADD CONSTRAINT relatorio_tarefa_fk FOREIGN KEY (id_relatorio_tarefa)
REFERENCES gerenciamento_producao.relatorio_tarefa (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: tarefa_uq | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.tarefa DROP CONSTRAINT IF EXISTS tarefa_uq CASCADE;
ALTER TABLE gerenciamento_producao.tarefa ADD CONSTRAINT tarefa_uq UNIQUE (id_relatorio_tarefa);
-- ddl-end --

-- object: gerenciamento_producao.artefato | type: TABLE --
-- DROP TABLE IF EXISTS gerenciamento_producao.artefato CASCADE;
CREATE TABLE gerenciamento_producao.artefato(
    id smallint NOT NULL,
    id_tarefa smallint,
    id_tipo_artefato integer,
    CONSTRAINT artefato_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE gerenciamento_producao.artefato OWNER TO postgres;
-- ddl-end --

-- object: linha_producao.tp_publicacao | type: TYPE --
-- DROP TYPE IF EXISTS linha_producao.tp_publicacao CASCADE;
CREATE TYPE linha_producao.tp_publicacao AS
 ENUM ('ftp','wms','wfs','wcs');
-- ddl-end --
ALTER TYPE linha_producao.tp_publicacao OWNER TO postgres;
-- ddl-end --

-- object: linha_producao.forma_publicacao | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.forma_publicacao CASCADE;
CREATE TABLE linha_producao.forma_publicacao(
    id smallint NOT NULL,
    tipo_publicacao linha_producao.tp_publicacao NOT NULL,
    CONSTRAINT publicacao_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE linha_producao.forma_publicacao OWNER TO postgres;
-- ddl-end --
```

```
-- object: linha_producao.many_artefato_has_many_publicacao | type: TABLE --
-- DROP TABLE IF EXISTS linha_producao.many_artefato_has_many_publicacao CASCADE;
CREATE TABLE linha_producao.many_artefato_has_many_publicacao(
    id_tipo_artefato integer NOT NULL,
    id_forma_publicacao smallint NOT NULL,
    CONSTRAINT many_artefato_has_many_publicacao_pk PRIMARY KEY (id_tipo_artefato,id_forma_publicacao)

);
-- ddl-end --

-- object: tipo_artefato_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.many_artefato_has_many_publicacao DROP CONSTRAINT IF EXISTS tipo_artefato_fk CASCADE;
ALTER TABLE linha_producao.many_artefato_has_many_publicacao ADD CONSTRAINT tipo_artefato_fk FOREIGN KEY (id_tipo_artefato)
REFERENCES linha_producao.tipo_artefato (id) MATCH FULL
ON DELETE RESTRICT ON UPDATE CASCADE;
-- ddl-end --

-- object: forma_publicacao_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.many_artefato_has_many_publicacao DROP CONSTRAINT IF EXISTS forma_publicacao_fk CASCADE;
ALTER TABLE linha_producao.many_artefato_has_many_publicacao ADD CONSTRAINT forma_publicacao_fk FOREIGN KEY (id_forma_publicacao)
REFERENCES linha_producao.forma_publicacao (id) MATCH FULL
ON DELETE RESTRICT ON UPDATE CASCADE;
-- ddl-end --

-- object: gerenciamento_producao.relatorio_artefato | type: TABLE --
-- DROP TABLE IF EXISTS gerenciamento_producao.relatorio_artefato CASCADE;
CREATE TABLE gerenciamento_producao.relatorio_artefato(
    id smallint NOT NULL,
    id_artefato smallint,
    id_template_xform integer,
    data_entrega date NOT NULL,
    CONSTRAINT relatorio_artefato_pk PRIMARY KEY (id)

);
-- ddl-end --
ALTER TABLE gerenciamento_producao.relatorio_artefato OWNER TO postgres;
-- ddl-end --

-- object: artefato_fk | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.relatorio_artefato DROP CONSTRAINT IF EXISTS artefato_fk CASCADE;
ALTER TABLE gerenciamento_producao.relatorio_artefato ADD CONSTRAINT artefato_fk FOREIGN KEY (id_artefato)
REFERENCES gerenciamento_producao.artefato (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: relatorio_artefato_uq | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.relatorio_artefato DROP CONSTRAINT IF EXISTS relatorio_artefato_uq CASCADE;
ALTER TABLE gerenciamento_producao.relatorio_artefato ADD CONSTRAINT relatorio_artefato_uq UNIQUE (id_artefato);
-- ddl-end --

-- object: tarefa_fk | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.artefato DROP CONSTRAINT IF EXISTS tarefa_fk CASCADE;
ALTER TABLE gerenciamento_producao.artefato ADD CONSTRAINT tarefa_fk FOREIGN KEY (id_tarefa)
REFERENCES gerenciamento_producao.tarefa (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: gerenciamento_producao.arquivo | type: TABLE --
-- DROP TABLE IF EXISTS gerenciamento_producao.arquivo CASCADE;
CREATE TABLE gerenciamento_producao.arquivo(
    id smallint NOT NULL,
    path character varying(2000) NOT NULL,
    id_artefato smallint,
    CONSTRAINT arquivo_pk PRIMARY KEY (id)
```

```
);
-- ddl-end --
ALTER TABLE gerenciamento_producao.arquivo OWNER TO postgres;
-- ddl-end --

-- object: tipo_artefato_fk | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.artefato DROP CONSTRAINT IF EXISTS tipo_artefato_fk CASCADE;
ALTER TABLE gerenciamento_producao.artefato ADD CONSTRAINT tipo_artefato_fk FOREIGN KEY (id_tipo_artefato)
REFERENCES linha_producao.tipo_artefato (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: artefato_uq | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.artefato DROP CONSTRAINT IF EXISTS artefato_uq CASCADE;
ALTER TABLE gerenciamento_producao.artefato ADD CONSTRAINT artefato_uq UNIQUE (id_tipo_artefato);
-- ddl-end --

-- object: artefato_fk | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.arquivo DROP CONSTRAINT IF EXISTS artefato_fk CASCADE;
ALTER TABLE gerenciamento_producao.arquivo ADD CONSTRAINT artefato_fk FOREIGN KEY (id_artefato)
REFERENCES gerenciamento_producao.artefato (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: template_xform_fk | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.relatorio_artefato DROP CONSTRAINT IF EXISTS template_xform_fk
CASCADE;
ALTER TABLE gerenciamento_producao.relatorio_artefato ADD CONSTRAINT template_xform_fk FOREIGN KEY
(id_template_xform)
REFERENCES linha_producao.template_xform (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: template_xform_fk | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.relatorio_tarefa DROP CONSTRAINT IF EXISTS template_xform_fk CASCADE;
ALTER TABLE gerenciamento_producao.relatorio_tarefa ADD CONSTRAINT template_xform_fk FOREIGN KEY
(id_template_xform)
REFERENCES linha_producao.template_xform (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: identificacao_cdg_fk | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.produto_cartografico DROP CONSTRAINT IF EXISTS identificacao_cdg_fk
CASCADE;
ALTER TABLE gerenciamento_producao.produto_cartografico ADD CONSTRAINT identificacao_cdg_fk FOREIGN KEY
(id_identificacao_cdg)
REFERENCES metadado.identificacao_cdg (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: produto_cartografico_uq | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.produto_cartografico DROP CONSTRAINT IF EXISTS produto_cartografico_uq
CASCADE;
ALTER TABLE gerenciamento_producao.produto_cartografico ADD CONSTRAINT produto_cartografico_uq UNIQUE
(id_identificacao_cdg);
-- ddl-end --

-- object: conjunto_inventariar_fk | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.produto_cartografico DROP CONSTRAINT IF EXISTS conjunto_inventariar_fk
CASCADE;
ALTER TABLE gerenciamento_producao.produto_cartografico ADD CONSTRAINT conjunto_inventariar_fk FOREIGN KEY
(id_conjunto_inventariar)
REFERENCES gerenciamento_producao.conjunto_inventariar (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end --

-- object: identificacao_cdg_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.citacao DROP CONSTRAINT IF EXISTS identificacao_cdg_fk CASCADE;
ALTER TABLE metadado.citacao ADD CONSTRAINT identificacao_cdg_fk FOREIGN KEY (identificacao)
REFERENCES metadado.identificacao (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
```

```
-- ddl-end --

-- object: tipo_de_data_cdg_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.citacao DROP CONSTRAINT IF EXISTS tipo_de_data_cdg_fk CASCADE;
ALTER TABLE metadado.citacao ADD CONSTRAINT tipo_de_data_cdg_fk FOREIGN KEY (tipo_de_data)
REFERENCES metadado."CI_DateTypeCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: template_xform_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.etapa_de_trabalho DROP CONSTRAINT IF EXISTS template_xform_fk CASCADE;
ALTER TABLE linha_producao.etapa_de_trabalho ADD CONSTRAINT template_xform_fk FOREIGN KEY
(template_relatorio)
REFERENCES linha_producao.template_xform (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: manual_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.etapa_de_trabalho DROP CONSTRAINT IF EXISTS manual_fk CASCADE;
ALTER TABLE linha_producao.etapa_de_trabalho ADD CONSTRAINT manual_fk FOREIGN KEY (manual_de_instrucao)
REFERENCES linha_producao.manual (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: formato_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.tipo_artefato DROP CONSTRAINT IF EXISTS formato_fk CASCADE;
ALTER TABLE linha_producao.tipo_artefato ADD CONSTRAINT formato_fk FOREIGN KEY (formato_arquivo)
REFERENCES linha_producao.formato_arquivo (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: validacao_manual_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.validacao DROP CONSTRAINT IF EXISTS validacao_manual_fk CASCADE;
ALTER TABLE linha_producao.validacao ADD CONSTRAINT validacao_manual_fk FOREIGN KEY (validacao_manual)
REFERENCES linha_producao.validacao_manual (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: template_xform_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.validacao DROP CONSTRAINT IF EXISTS template_xform_fk CASCADE;
ALTER TABLE linha_producao.validacao ADD CONSTRAINT template_xform_fk FOREIGN KEY (template_relatorio)
REFERENCES linha_producao.template_xform (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: validacao_automatica | type: CONSTRAINT --
-- ALTER TABLE linha_producao.validacao DROP CONSTRAINT IF EXISTS validacao_automatica CASCADE;
ALTER TABLE linha_producao.validacao ADD CONSTRAINT validacao_automatica FOREIGN KEY (validacao_automatica)
REFERENCES linha_producao.validacao_automatica (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: manual_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.validacao_manual DROP CONSTRAINT IF EXISTS manual_fk CASCADE;
ALTER TABLE linha_producao.validacao_manual ADD CONSTRAINT manual_fk FOREIGN KEY (manual)
REFERENCES linha_producao.manual (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: formato_arquivo_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.validacao_automatica DROP CONSTRAINT IF EXISTS formato_arquivo_fk CASCADE;
ALTER TABLE linha_producao.validacao_automatica ADD CONSTRAINT formato_arquivo_fk FOREIGN KEY (tipo_arquivo)
REFERENCES linha_producao.formato_arquivo (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: script | type: CONSTRAINT --
-- ALTER TABLE linha_producao.validacao_automatica DROP CONSTRAINT IF EXISTS script CASCADE;
ALTER TABLE linha_producao.validacao_automatica ADD CONSTRAINT script FOREIGN KEY (script)
REFERENCES linha_producao.script (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
```

```
-- ddl-end --

-- object: profissional_fk_manual | type: CONSTRAINT --
-- ALTER TABLE linha_producao.manual DROP CONSTRAINT IF EXISTS profissional_fk_manual CASCADE;
ALTER TABLE linha_producao.manual ADD CONSTRAINT profissional_fk_manual FOREIGN KEY (autor)
REFERENCES pessoal.profissional (cpf) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: profissional_fk_script | type: CONSTRAINT --
-- ALTER TABLE linha_producao.script DROP CONSTRAINT IF EXISTS profissional_fk_script CASCADE;
ALTER TABLE linha_producao.script ADD CONSTRAINT profissional_fk_script FOREIGN KEY (autor)
REFERENCES pessoal.profissional (cpf) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: funcao_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.ponto_de_contato DROP CONSTRAINT IF EXISTS funcao_fk CASCADE;
ALTER TABLE metadado.ponto_de_contato ADD CONSTRAINT funcao_fk FOREIGN KEY (funcao)
REFERENCES metadado."CI_RoleCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: tipo_palavra_chave_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.palavras_chaves_descritivas DROP CONSTRAINT IF EXISTS tipo_palavra_chave_fk
CASCADE;
ALTER TABLE metadado.palavras_chaves_descritivas ADD CONSTRAINT tipo_palavra_chave_fk FOREIGN KEY
(tipo_palavra_chave)
REFERENCES metadado."MD_KeywordTypeCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: identificacao_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.palavras_chaves_descritivas DROP CONSTRAINT IF EXISTS identificacao_fk CASCADE;
ALTER TABLE metadado.palavras_chaves_descritivas ADD CONSTRAINT identificacao_fk FOREIGN KEY (identificacao)
REFERENCES metadado.identificacao (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: identificacao_cdg | type: CONSTRAINT --
-- ALTER TABLE metadado.nome_do_arquivo_agregado DROP CONSTRAINT IF EXISTS identificacao_cdg CASCADE;
ALTER TABLE metadado.nome_do_arquivo_agregado ADD CONSTRAINT identificacao_cdg FOREIGN KEY (identificacao)
REFERENCES metadado.identificacao (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: tipo_de_data_cdg_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.nome_do_arquivo_agregado DROP CONSTRAINT IF EXISTS tipo_de_data_cdg_fk CASCADE;
ALTER TABLE metadado.nome_do_arquivo_agregado ADD CONSTRAINT tipo_de_data_cdg_fk FOREIGN KEY
(tipo_de_data)
REFERENCES metadado."CI_DateTypeCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: status_cdg_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao DROP CONSTRAINT IF EXISTS status_cdg_fk CASCADE;
ALTER TABLE metadado.identificacao ADD CONSTRAINT status_cdg_fk FOREIGN KEY (status)
REFERENCES metadado."MD_ProgressCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: ponto_de_contato_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao DROP CONSTRAINT IF EXISTS ponto_de_contato_fk CASCADE;
ALTER TABLE metadado.identificacao ADD CONSTRAINT ponto_de_contato_fk FOREIGN KEY (ponto_de_contato)
REFERENCES metadado.ponto_de_contato (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: tipo_de_associacao_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao DROP CONSTRAINT IF EXISTS tipo_de_associacao_fk CASCADE;
ALTER TABLE metadado.identificacao ADD CONSTRAINT tipo_de_associacao_fk FOREIGN KEY (tipo_de_associacao)
```

```
REFERENCES metadado."DS_AssociationTypeCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: iso_639_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.metametadados DROP CONSTRAINT IF EXISTS iso_639_fk CASCADE;
ALTER TABLE metadado.metametadados ADD CONSTRAINT iso_639_fk FOREIGN KEY (idioma)
REFERENCES metadado.iso_639 (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: norma_de_codificacao_de_caracteres_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.metametadados DROP CONSTRAINT IF EXISTS norma_de_codificacao_de_caracteres_fk CASCADE;
ALTER TABLE metadado.metametadados ADD CONSTRAINT norma_de_codificacao_de_caracteres_fk FOREIGN KEY (norma_de_codificacao_de_caracteres)
REFERENCES metadado."MD_CharacterSetCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: nivel_hierarquico_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.metametadados DROP CONSTRAINT IF EXISTS nivel_hierarquico_fk CASCADE;
ALTER TABLE metadado.metametadados ADD CONSTRAINT nivel_hierarquico_fk FOREIGN KEY (nivel_hierarquico)
REFERENCES metadado."MD_ScopeCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: contato_do_responsavel_pelos_metadados_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.metametadados DROP CONSTRAINT IF EXISTS contato_do_responsavel_pelos_metadados_fk CASCADE;
ALTER TABLE metadado.metametadados ADD CONSTRAINT contato_do_responsavel_pelos_metadados_fk FOREIGN KEY (contato_do_responsavel_pelos_metadados)
REFERENCES metadado.ponto_de_contato (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: restricoes_de_acesso_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.metametadados DROP CONSTRAINT IF EXISTS restricoes_de_acesso_fk CASCADE;
ALTER TABLE metadado.metametadados ADD CONSTRAINT restricoes_de_acesso_fk FOREIGN KEY (restricao_de_acesso)
REFERENCES metadado."MD_RestrictionCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: restricoes_de_uso_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.metametadados DROP CONSTRAINT IF EXISTS restricoes_de_uso_fk CASCADE;
ALTER TABLE metadado.metametadados ADD CONSTRAINT restricoes_de_uso_fk FOREIGN KEY (restricoes_de_uso)
REFERENCES metadado."MD_RestrictionCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: identificacao_cdg_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.extensao DROP CONSTRAINT IF EXISTS identificacao_cdg_fk CASCADE;
ALTER TABLE metadado.extensao ADD CONSTRAINT identificacao_cdg_fk FOREIGN KEY (identificacao_cdg)
REFERENCES metadado.identificacao_cdg (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: extensao_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.extensao_temporal DROP CONSTRAINT IF EXISTS extensao_fk CASCADE;
ALTER TABLE metadado.extensao_temporal ADD CONSTRAINT extensao_fk FOREIGN KEY (id)
REFERENCES metadado.extensao (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: extensao_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.extensao_altimetrica DROP CONSTRAINT IF EXISTS extensao_fk CASCADE;
ALTER TABLE metadado.extensao_altimetrica ADD CONSTRAINT extensao_fk FOREIGN KEY (id)
REFERENCES metadado.extensao (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --
```

```
-- object: restricoes_de_acesso_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.informacoes_de_restricao DROP CONSTRAINT IF EXISTS restricoes_de_acesso_fk CASCADE;
ALTER TABLE metadado.informacoes_de_restricao ADD CONSTRAINT restricoes_de_acesso_fk FOREIGN KEY
(restricoes_de_acesso)
REFERENCES metadado."MD_RestrictionCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: restricoes_de_uso_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.informacoes_de_restricao DROP CONSTRAINT IF EXISTS restricoes_de_uso_fk CASCADE;
ALTER TABLE metadado.informacoes_de_restricao ADD CONSTRAINT restricoes_de_uso_fk FOREIGN KEY
(restricoes_de_uso)
REFERENCES metadado."MD_RestrictionCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: restricoes_de_seguranca_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.informacoes_de_restricao DROP CONSTRAINT IF EXISTS restricoes_de_seguranca_fk
CASCADE;
ALTER TABLE metadado.informacoes_de_restricao ADD CONSTRAINT restricoes_de_seguranca_fk FOREIGN KEY
(restricoes_de_seguranca)
REFERENCES metadado."MD_ClassificationCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: nivel_hierarquico_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.qualidade DROP CONSTRAINT IF EXISTS nivel_hierarquico_fk CASCADE;
ALTER TABLE metadado.qualidade ADD CONSTRAINT nivel_hierarquico_fk FOREIGN KEY (nivel_hierarquico)
REFERENCES metadado."MD_ScopeCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: qualidade_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.linhagem DROP CONSTRAINT IF EXISTS qualidade_fk CASCADE;
ALTER TABLE metadado.linhagem ADD CONSTRAINT qualidade_fk FOREIGN KEY (qualidade)
REFERENCES metadado.qualidade (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: qualidade_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.relatorio DROP CONSTRAINT IF EXISTS qualidade_fk CASCADE;
ALTER TABLE metadado.relatorio ADD CONSTRAINT qualidade_fk FOREIGN KEY (qualidade)
REFERENCES metadado.qualidade (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: frequencia_manutencao_e_atualizacao_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.informacao_de_manutencao DROP CONSTRAINT IF EXISTS
frequencia_manutencao_e_atualizacao_fk CASCADE;
ALTER TABLE metadado.informacao_de_manutencao ADD CONSTRAINT frequencia_manutencao_e_atualizacao_fk
FOREIGN KEY (frequencia_manutencao_e_atualizacao)
REFERENCES metadado."MD_MaintenanceFrequencyCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: nivel_topologico_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.representacao_espacial_vetorial DROP CONSTRAINT IF EXISTS nivel_topologico_fk
CASCADE;
ALTER TABLE metadado.representacao_espacial_vetorial ADD CONSTRAINT nivel_topologico_fk FOREIGN KEY
(nivel_topologico)
REFERENCES metadado."MD_TopologyLevelCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: tipo_dos_objetos_geometricos_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.representacao_espacial_vetorial DROP CONSTRAINT IF EXISTS
tipo_dos_objetos_geometricos_fk CASCADE;
ALTER TABLE metadado.representacao_espacial_vetorial ADD CONSTRAINT tipo_dos_objetos_geometricos_fk FOREIGN
KEY (tipo_dos_objetos_geometricos)
REFERENCES metadado."MD_GeometricObjectTypeCode" (domain_code) MATCH FULL
```

```
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: informacao_representacao_espacial_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.representacao_espacial_vetorial DROP CONSTRAINT IF EXISTS informacao_representacao_espacial_fk CASCADE;
ALTER TABLE metadado.representacao_espacial_vetorial ADD CONSTRAINT informacao_representacao_espacial_fk FOREIGN KEY (informacao_representacao_espacial)
REFERENCES metadado.informacao_representacao_espacial (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: informacao_representacao_espacial_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.representacao_espacial_matricial DROP CONSTRAINT IF EXISTS informacao_representacao_espacial_fk CASCADE;
ALTER TABLE metadado.representacao_espacial_matricial ADD CONSTRAINT informacao_representacao_espacial_fk FOREIGN KEY (informacao_representacao_espacial)
REFERENCES metadado.informacao_representacao_espacial (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: referencia_no_pixel_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.representacao_espacial_matricial_georretificada DROP CONSTRAINT IF EXISTS referencia_no_pixel_fk CASCADE;
ALTER TABLE metadado.representacao_espacial_matricial_georretificada ADD CONSTRAINT referencia_no_pixel_fk FOREIGN KEY (referencia_no_pixel)
REFERENCES metadado."MD_PixelOrientationCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: representacao_espacial_matricial_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.representacao_espacial_matricial_georretificada DROP CONSTRAINT IF EXISTS representacao_espacial_matricial_fk CASCADE;
ALTER TABLE metadado.representacao_espacial_matricial_georretificada ADD CONSTRAINT representacao_espacial_matricial_fk FOREIGN KEY (representacao_espacial_matricial)
REFERENCES metadado.representacao_espacial_matricial (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: representacao_espacial_matricial_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.representacao_espacial_matricial_georreferenciavel DROP CONSTRAINT IF EXISTS representacao_espacial_matricial_fk CASCADE;
ALTER TABLE metadado.representacao_espacial_matricial_georreferenciavel ADD CONSTRAINT representacao_espacial_matricial_fk FOREIGN KEY (representacao_espacial_matricial)
REFERENCES metadado.representacao_espacial_matricial (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: citacao_catalogo_de_feicoes_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.descricao_catalogo_de_feicoes DROP CONSTRAINT IF EXISTS citacao_catalogo_de_feicoes_fk CASCADE;
ALTER TABLE metadado.descricao_catalogo_de_feicoes ADD CONSTRAINT citacao_catalogo_de_feicoes_fk FOREIGN KEY (citacao_catalogo_de_feicoes)
REFERENCES metadado."CI_Citation" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: informacao_de_conteudo_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.descricao_catalogo_de_feicoes DROP CONSTRAINT IF EXISTS informacao_de_conteudo_fk CASCADE;
ALTER TABLE metadado.descricao_catalogo_de_feicoes ADD CONSTRAINT informacao_de_conteudo_fk FOREIGN KEY (informacao_de_conteudo)
REFERENCES metadado.informacao_de_conteudo (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: informacao_de_conteudo_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.descricao_dados_matriciais DROP CONSTRAINT IF EXISTS informacao_de_conteudo_fk CASCADE;
ALTER TABLE metadado.descricao_dados_matriciais ADD CONSTRAINT informacao_de_conteudo_fk FOREIGN KEY (informacao_de_conteudo)
```

```
REFERENCES metadado.informacao_de_conteudo (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: tipo_da_informacao_representada_pelo_valor_do_pixel_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.descricao_dados_matriciais DROP CONSTRAINT IF EXISTS tipo_da_informacao_representada_pelo_valor_do_pixel_fk CASCADE;
ALTER TABLE metadado.descricao_dados_matriciais ADD CONSTRAINT tipo_da_informacao_representada_pelo_valor_do_pixel_fk FOREIGN KEY (tipo_da_informacao_representada_pelo_valor_do_pixel)
REFERENCES metadado."MD_CoverageContentTypeCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: descricao_dados_matriciais_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.banda_espectral DROP CONSTRAINT IF EXISTS descricao_dados_matriciais_fk CASCADE;
ALTER TABLE metadado.banda_espectral ADD CONSTRAINT descricao_dados_matriciais_fk FOREIGN KEY (descricao_dados_matriciais)
REFERENCES metadado.descricao_dados_matriciais (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: contato_do_distribuidor_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.distribuicao DROP CONSTRAINT IF EXISTS contato_do_distribuidor_fk CASCADE;
ALTER TABLE metadado.distribuicao ADD CONSTRAINT contato_do_distribuidor_fk FOREIGN KEY (contato_do_distribuidor)
REFERENCES metadado.ponto_de_contato (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: tipo_representacao_espacial_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS tipo_representacao_espacial_fk CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT tipo_representacao_espacial_fk FOREIGN KEY (tipo_representacao_espacial)
REFERENCES metadado."MD_SpatialRepresentationTypeCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: iso_639_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS iso_639_fk CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT iso_639_fk FOREIGN KEY (idioma)
REFERENCES metadado.iso_639 (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: norma_de_codificacao_de_caracteres_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS norma_de_codificacao_de_caracteres_fk CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT norma_de_codificacao_de_caracteres_fk FOREIGN KEY (norma_codificacao_de_caracteres)
REFERENCES metadado."MD_CharacterSetCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: categoria_tematica_fk | type: CONSTRAINT --
-- ALTER TABLE metadado.identificacao_cdg DROP CONSTRAINT IF EXISTS categoria_tematica_fk CASCADE;
ALTER TABLE metadado.identificacao_cdg ADD CONSTRAINT categoria_tematica_fk FOREIGN KEY (categoria_tematica)
REFERENCES metadado."MD_TopicCategoryCode" (domain_code) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: status_produto_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.tipo_produto DROP CONSTRAINT IF EXISTS status_produto_fk CASCADE;
ALTER TABLE linha_producao.tipo_produto ADD CONSTRAINT status_produto_fk FOREIGN KEY (status_final)
REFERENCES linha_producao.status_produto (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: status_fk | type: CONSTRAINT --
-- ALTER TABLE linha_producao.status_requeridos DROP CONSTRAINT IF EXISTS status_fk CASCADE;
ALTER TABLE linha_producao.status_requeridos ADD CONSTRAINT status_fk FOREIGN KEY (status)
```

```
REFERENCES linha_producao.status_produto (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: status_fk_requerido | type: CONSTRAINT --
-- ALTER TABLE linha_producao.status_requeridos DROP CONSTRAINT IF EXISTS status_fk_requerido CASCADE;
ALTER TABLE linha_producao.status_requeridos ADD CONSTRAINT status_fk_requerido FOREIGN KEY (status_requerido)
REFERENCES linha_producao.status_produto (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: tipo_produto_fk | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.produto_cartografico DROP CONSTRAINT IF EXISTS tipo_produto_fk CASCADE;
ALTER TABLE gerenciamento_producao.produto_cartografico ADD CONSTRAINT tipo_produto_fk FOREIGN KEY (tipo_produto)
REFERENCES linha_producao.tipo_produto (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: status_produto_sk | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.produto_cartografico DROP CONSTRAINT IF EXISTS status_produto_sk CASCADE;
ALTER TABLE gerenciamento_producao.produto_cartografico ADD CONSTRAINT status_produto_sk FOREIGN KEY (status_inicial)
REFERENCES linha_producao.status_produto (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: produto_fk | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.tarefa DROP CONSTRAINT IF EXISTS produto_fk CASCADE;
ALTER TABLE gerenciamento_producao.tarefa ADD CONSTRAINT produto_fk FOREIGN KEY (produto)
REFERENCES gerenciamento_producao.produto_cartografico (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: profissional_fk_executor | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.tarefa DROP CONSTRAINT IF EXISTS profissional_fk_executor CASCADE;
ALTER TABLE gerenciamento_producao.tarefa ADD CONSTRAINT profissional_fk_executor FOREIGN KEY (executor)
REFERENCES pessoal.profissional (cpf) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: profissional_fk_validador | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.tarefa DROP CONSTRAINT IF EXISTS profissional_fk_validador CASCADE;
ALTER TABLE gerenciamento_producao.tarefa ADD CONSTRAINT profissional_fk_validador FOREIGN KEY (validador)
REFERENCES pessoal.profissional (cpf) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: produto_fk | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.historico_status DROP CONSTRAINT IF EXISTS produto_fk CASCADE;
ALTER TABLE gerenciamento_producao.historico_status ADD CONSTRAINT produto_fk FOREIGN KEY (produto)
REFERENCES gerenciamento_producao.produto_cartografico (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: tarefa_fk | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.historico_status DROP CONSTRAINT IF EXISTS tarefa_fk CASCADE;
ALTER TABLE gerenciamento_producao.historico_status ADD CONSTRAINT tarefa_fk FOREIGN KEY (tarefa)
REFERENCES gerenciamento_producao.tarefa (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --

-- object: status_produto_fk | type: CONSTRAINT --
-- ALTER TABLE gerenciamento_producao.historico_status DROP CONSTRAINT IF EXISTS status_produto_fk CASCADE;
ALTER TABLE gerenciamento_producao.historico_status ADD CONSTRAINT status_produto_fk FOREIGN KEY (status)
REFERENCES linha_producao.status_produto (id) MATCH FULL
ON DELETE NO ACTION ON UPDATE NO ACTION;
-- ddl-end --
```