**Manual de Operação da Ferramenta de Otimização de Rotas**

Capítulo 1: Large Office

1. Dados Necessários para entrada no modelo:
   1. “Dados.xlsx”: Este arquivo é responsável por determinar quais os patrimônios que serão roteirizados, assim como suas informações necessárias. Ele é composto por três abas¹.
      1. “parceiros”

As colunas presentes são:

- FILIAL: Sigla da filial, como CPS, RJ, SP, BA...

- PARCEIRO: Nome do parceiro².

- LAT: Latitude do parceiro.

- LON: Longitude do parceiro.

- TEMPO\_DE\_ENTRADA\_MIN: Tempo de entrada, em minutos, no parceiro.

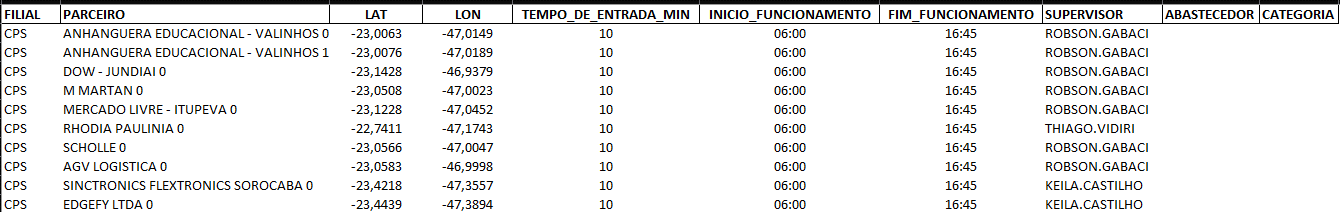
- INICIO\_FUNCIONAMENTO: Horário inicial desejado para chegada no local.

- FIM\_FUNCIONAMENTO: Horário final desejado para chegada no local.

- SUPERVISOR: Supervisor associado ao parceiro.

- ABASTECEDOR: caso deseje fixar um abastecedor no parceiro específico, basta escrever. Entende-se que este artificio dificulta obter uma solução no modelo e por isso deve ser usado com cautela.

- CATEGORIA: Flag somente, que associa uma categoria (Hospital, Laboratório...). Útil para identificar horários de acesso especiais.



* + 1. “patrimonios”

As colunas presentes são:

- FILIAL: Sigla da filial, como CPS, RJ, SP, BA...

- PARCEIRO: Nome do parceiro².

- PATRIMONIO: Nome do patrimônio³.

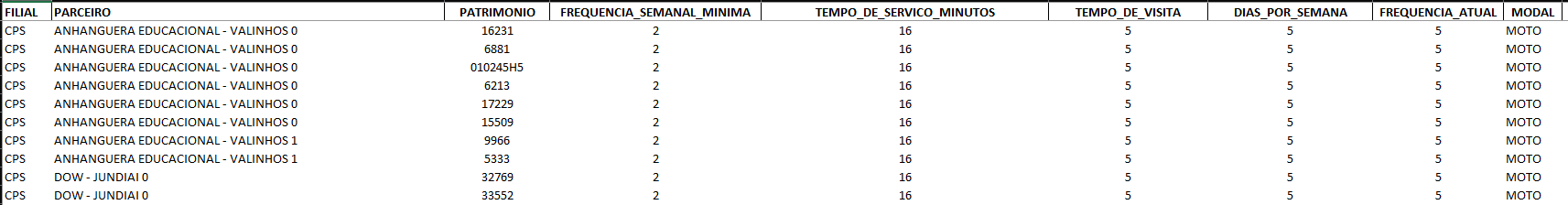
- FREQUENCIA\_SEMANAL\_MINIMA: Frequência semanal mínima de vista ao patrimônio.

- TEMPO\_DE\_SERVICO\_MINUTOS: Tempo de abastecimento⁴ no patrimônio.

- TEMPO\_DE\_VISITA: Tempo de visita**⁵** no patrimônio.

- DIAS\_POR\_SEMANA: Quantidade de dias possíveis de abastecimento (5, para segunda a sexta e 6, para segunda a sábado)

- FREQUENCIA\_ATUAL: Frequência atual de abastecimento no patrimônio.



* + 1. “insumos”

As colunas presentes são:

- FILIAL: Sigla da filial, como CPS, RJ, SP, BA...

- PARCEIRO: Nome do parceiro².

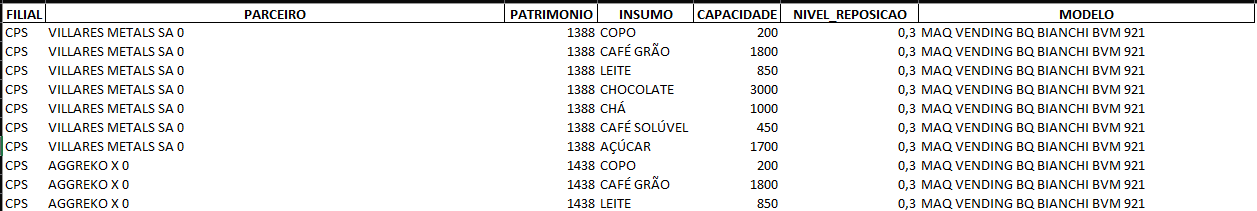
- PATRIMONIO: Nome do patrimônio³.

- INSUMO: Nome do insumo que tem capacidade na respectiva máquina.

- CAPACIDADE: Capacidade em gramas por insumo.

-NÍVEL\_DE\_REPOSIÇÃO: Nível de reposição para identificar que necessita abastecimento. Padrão é 0,3 (30%) da capacidade da máquina.

- MODELO: Flag somente, que associa qual a máquina do parceiro.



* 1. “Dados de Consumo.csv”: Este arquivo é responsável por indicar o consumo de insumos por patrimônio⁶. Ele é composto por somente uma aba.

As colunas presentes são:

- FILIAL: Sigla da filial, como CPS, RJ, SP, BA...

- PARCEIRO: Nome do parceiro².

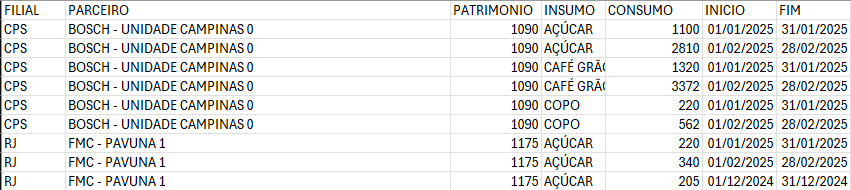
- PATRIMONIO: Nome do patrimônio³.

- INSUMO: Nome do insumo consumido.

- CONSUMO: Consumo⁷ do insumo durante o mês identificado.

- INICIO: Data do início do mês.

- FIM: Data do fim do mês.



* 1. Aba de configurações do arquivo Final

Na Aba de “Configurações Gerais” do arquivo final de resultado, são possíveis inserir configurações que vão se aplicar ao modelo. Elas são:

- Dias considerados por semana: Total de dias considerados na semana para realizar a roteirização.

- Escala: Definição de horas diárias para Part-Time, RPA 3h e RPA 2h.

- Tempo Máximo de Otimização por período e Filial (segundos): tempo limite máximo em que o modelo pode estar calculando a solução para cada rota por período e filial.

- Modal a ser considerado: Opção de Carro/Moto ou a pé.

- Tolerância no tempo total: Tolerância no tempo total da rota, considerando a escala do abastecedor.

- Adicionar repasses: Recurso para visita de um patrimônio, mais de uma vez no dia (opção “Sim).

- Roteirização 1 abastecedor por máquina:

- Definição da função 1:1, opção “Sim” (somente um abastecedor responsável pelo mesmo patrimônio)

- 1:n (é possível mais de um abastecedor ser responsável pelo mesmo patrimônio)

- Visitar todas as máquinas no parceiro visitado:

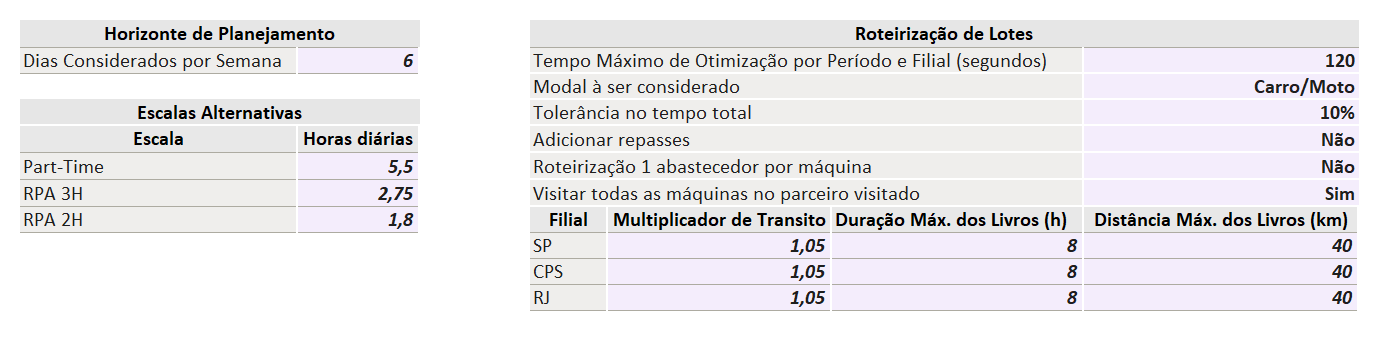
- Opção “Sim”: opção de visita acionada

- Opção “Não”: opção de visita desabilitada

- Multiplicador de Trânsito: fator multiplicativo de trânsito para cada cidade

- Duração Máx. dos Livros (h): Duração máxima de uma rota para um abastecedor.

- Distância Máx. dos Livros (km): Distância máxima de uma rota para um abastecedor.



**Pontos de atenção:**

¹é obrigatório que as informações entre as três abas sejam coerentes. Isto é, é necessário que a quantidade de patrimônios e parceiros, por filial, seja a mesma nas três abas. Por exemplo, caso um patrimônio esteja na aba de “insumos”, porém não esteja na aba de “patrimônios”, haverá erro no código.

² para cada filial, um parceiro deve ter somente uma Latitude e Longitude. Foram identificados casos em que um mesmo parceiro tinha dois patrimônios, sendo cada um em de endereços diferentes. Logo como são coordenadas diferentes, os parceiros devem ser identificados como diferentes. A diferenciação foi feita através de “0”, “1”, “2” ... ao final do nome do parceiro.

³quando um patrimônio possui seu nome composto somente por números, sendo seu primeiro digito “0”, o Excel identifica isso como texto e pede para transformar a célula em número. Ao transformar em número, o “0” é eliminado no início nome. É importante salientar-se que se essa transformação for feita deve ocorrer em todos os locais (Dados.xlsx e Dados de Consumo.csv).

⁴abastecimento e o ato de fazer a reposição e limpeza completa de insumos, como recomendado nos treinamentos.

⁵visita é o ato de limpeza superficial na máquina, sem necessidade de abastecimento de insumos.

⁶insumos especiais como 'CAFÉ COM LEITE CARAMELO', 'CAPPUCCINO COM CANELA CAFÉ DO CENTRO', 'CAPPUCCINO SEM CANELA CAFÉ DO CENTRO' são consumidos em doses, porém não são especificados na capacidade das máquinas. Dessa forma, como especificado pela equipe técnica da Gran Coffee, esses insumos são colocados nos compartimentos vazios de CHOCOLATE, CHÁ ou CAFÉ SOLÚVEL. Assim, quando um patrimônio possui consumo desse insumo especial é necessário colocar seu consumo com nome do compartimento que ele foi inserido, para assim ter sinergia com os dados inseridos em “insumos” do arquivo “Dados.xlsx”.

⁷considerando a realocação de máquinas e o baixo histórico de consumo, é possível que alguns patrimônios não tenham sido incluídos na planilha "Dados de Consumo.csv". Portanto, não há problema se um patrimônio estiver presente em "Dados.xlsx" mas ausente em "Dados de Consumo.csv"; nesse caso, sua frequência será baseada na frequência semanal mínima imputada.

1. Entendimento dos arquivos .py

Os códigos devem ser operados de forma modular (na sequência indicada), para obter o resultado. A explicação de cada um encontra-se abaixo.

* 1. distance\_matrix.py

Este código Python utiliza a biblioteca “openrouteservice” para calcular matrizes de distância e duração entre todos os parceiros. O objetivo principal é obter informações distância e tempo de viagem entre odos os locais identificados na aba “parceiros” do arquivo “Dados.xlsx”. Esse código Python não é necessário de ser rodado sempre, somente, quando houver alguma alteração (mudança de coordenadas, ou adição de parceiros) na própria aba mencionada.

**Em resumo, o código realiza as seguintes etapas:**

1. **Carrega Dados:** Lê dados do arquivo Excel ("Dados.xlsx"), na aba “parceiros” contendo informações sobre os parceiros (incluindo suas coordenadas de latitude e longitude).
2. **Calcula Proximidade:** Utiliza a distância euclidiana para identificar os parceiros mais próximos dentro de um raio definido.
3. **Realiza Chamadas à API:**
   * Utiliza a chave de API fornecida para se conectar ao openrouteservice.
   * Para cada "chamada" gerada, envia uma requisição para obter a matriz de distâncias e durações entre os pontos de origem e destino especificados, considerando um perfil de transporte (carro ou a pé).
4. **Salva Resultados Finais:** Salva a matriz de distâncias e durações tratada em um arquivo Parquet ("distance\_matrix.parquet") para cada modal de transporte.
   1. frequency.py

Este código Python calcula quantas vezes por semana cada patrimônio precisa ser visitado para repor insumos. Ele deve ser rodado sempre quando houver alterações nas informações de consumo, de capacidade ou adição/remoção de patrimônios

**Ele faz isso seguindo estes passos:**

1. **Lê os dados:** Pega informações dos patrimônios do arquivo Excel e de consumo do CSV.
2. **Calcula o consumo:** Vê a média de quanto cada máquina gasta de cada material por semana.
3. **Estima a frequência:** Com base no consumo, na capacidade da máquina e em quanto se quer deixar de material de reserva, calcula quantas visitas semanais são necessárias para evitar que falte nenhum insumo.
4. **Mostra o resultado:** Escreve na aba "Frequências" do arquivo Excel a frequência de visitas calculada para cada máquina.
   1. cronograma.py

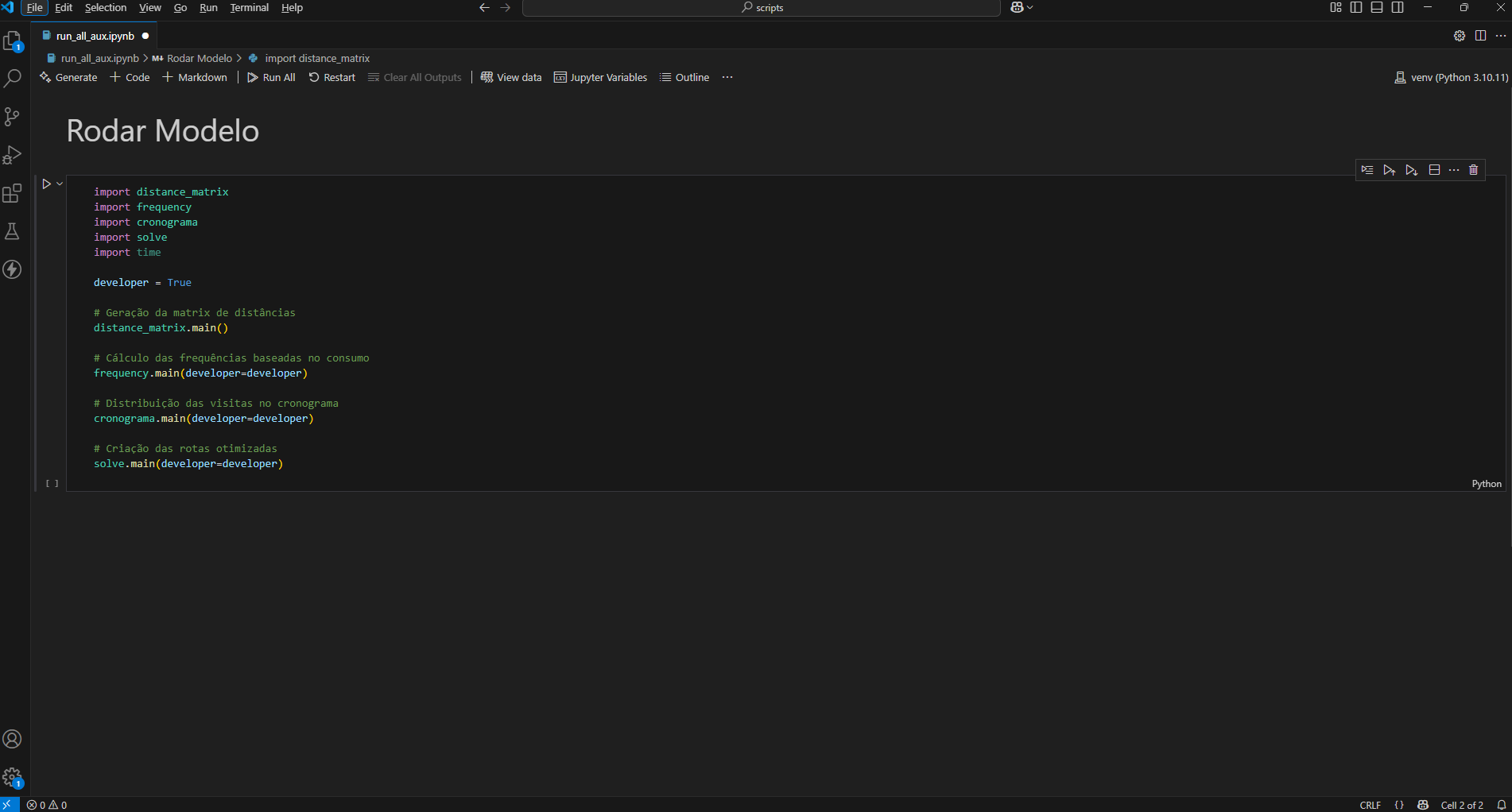
Este código Python cria um cronograma de visitas semanais para cada cliente (parceiro) e suas máquinas (patrimônios), respeitando a frequência de visitas definida anteriormente.

1. **Lê as frequências:** Pega da aba "Frequências" do arquivo Excel quantas vezes por semana cada máquina precisa ser visitada.
2. **Gera padrões de visita:** Para cada frequência (por exemplo, 3 vezes por semana), ele cria diferentes maneiras de distribuir essas visitas ao longo dos dias da semana (segunda, quarta e sexta; terça, quinta e sábado; etc.).
3. **Otimiza o cronograma:** Ele tenta escolher os padrões de visita de forma que o número de visitas em cada dia da semana seja o mais equilibrado possível entre todos os clientes e máquinas de uma mesma filial. Para isso, ele usa uma ferramenta de otimização matemática (ortools).
4. **Cria o cronograma:** Monta uma tabela mostrando para cada máquina em quais dias da semana a visita deve acontecer.
5. **Mostra o cronograma:** Escreve esse cronograma na aba "Cronograma" do arquivo Excel, formatando a tabela para facilitar a leitura.
   1. solve.py

Este código Python tem como objetivo otimizar a criação de rotas de visita para técnicos, minimizando o tempo e a distância total percorrida, respeitando janelas de tempo dos clientes e capacidade dos veículos.

1. **Lê os dados:** Importa informações de arquivos Excel (configurações, cronograma de visitas, dados de patrimônios e parceiros) e arquivos Parquet (matrizes de distância e IDs de pontos).
2. **Prepara os dados:** Organiza e transforma os dados lidos em um formato adequado para o modelo de otimização. Isso inclui definir janelas de tempo para cada cliente (horário de funcionamento), tempo de serviço em cada local, e calcular as distâncias e tempos de deslocamento entre os pontos.
3. **Agrupa as visitas:** Agrupa as visitas que devem ser feitas no mesmo período (dia da semana) para o mesmo cliente, considerando um limite máximo de tempo para cada "lote" de visitas.
4. **Otimiza as rotas:** Para cada período e para cada técnico (supervisor/abastecedor), o código utiliza um solver de otimização (ortools) para encontrar a melhor sequência de visitas dentro de um lote. O objetivo é minimizar a distância ou o tempo total da rota, respeitando as janelas de tempo dos clientes e a capacidade máxima de tempo de trabalho do técnico.
5. **Cria os "livros" de rota:** Os resultados da otimização são organizados em "livros" de rota, onde cada livro representa a sequência de visitas de um técnico em um determinado período.
6. **Salva os resultados:** As rotas otimizadas (os "livros") são salvas na aba "Livros" do arquivo Excel.
7. Operação do modelo

A operação do modelo baseia-se no arquivo run\_all\_aux.ipynb, no qual concatena todos os arquivos necessários explicados na seção anterior. Caso queira rodar somente distance\_matrix, ou algum outro arquivo, basta comentar as linhas dos outros arquivos.



Caso seja necessário reduzir a quantidade de patrimônios para roteirização ou ajustar algum parâmetro de configuração, basta incluir no arquivo “Dados.xlsx” apenas os patrimônios relevantes para esse escopo.

Por exemplo, se a equipe de Operações de Large Office quiser realizar a roteirização apenas para os patrimônios atualmente atendidos a pé, deve-se criar um “Dados.xlsx” contendo somente esses patrimônios e selecionar o modal “A pé” nas configurações. Portanto, dessa maneira podem ser criados “grupos de roteirização" para que rotas específicas não se misture com outras.

Capítulo 2: On the GO

* 1. Dados Necessários para entrada no modelo:
  2. “Dados.xlsx”: Este arquivo é responsável por determinar quais os patrimônios que serão roteirizados, assim como suas informações necessárias. Ele é composto por quatro abas¹.
     1. “parceiros”

As colunas presentes são:

- FILIAL: Sigla da filial, como CPS, RJ, SP, BA...

- PARCEIRO: Nome do parceiro².

- LAT: Latitude do parceiro.

- LON: Longitude do parceiro.

- TEMPO\_DE\_ENTRADA\_MIN: Tempo de entrada no parceiro.

- INICIO\_FUNCIONAMENTO: Horário inicial desejado para chegada no local.

- FIM\_FUNCIONAMENTO: Horário final desejado para chegada no local.

- SUPERVISOR: Supervisor associado ao parceiro.



* + 1. “patrimonios”

As colunas presentes são:

- FILIAL: Sigla da filial, como CPS, RJ, SP, BA...

- PARCEIRO: Nome do parceiro².

- PATRIMONIO: Nome do patrimônio³.

- FREQUENCIA\_SEMANAL\_ATUAL: Frequência atual de abastecimento no patrimônio.

- FREQUENCIA\_SEMANAL\_MINIMA: Frequência semanal mínima de vista ao patrimônio.

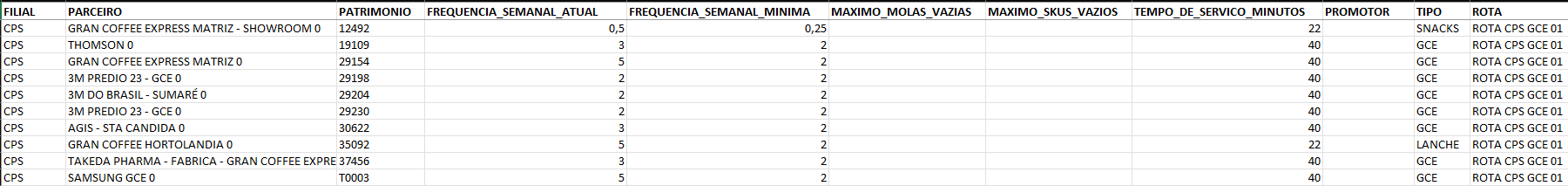
- MÁXIMO\_MOLAS\_VAZIAS: Máximo de molas vazias possíveis para o patrimônio, para solicitar abastecimento..

- MÁXIMO\_SKUS\_VAZIOS: Máximo de SKUs vazios para solicitar abastecimento,

- TEMPO\_DE\_SERVICO\_MINUTOS: Tempo de abastecimento no patrimônio.

- PROMOTOR: caso deseje fixar um promotor no parceiro específico, basta escrever. Entende-se que este artificio dificulta obter uma solução no modelo e por isso deve ser usado com cautela.

- TIPO: Flag somente, que associa o tipo de máquina (Snacks, GCE, Lanche...). Útil para identificar tempos de serviço e frequência semanal mínima especiais.



* + 1. “sku”

As colunas presentes são:

- FILIAL: Sigla da filial, como CPS, RJ, SP, BA...

- PARCEIRO: Nome do parceiro².

- PATRIMONIO: Nome do patrimônio³.

- SKU: Nome do SKU que tem capacidade na respectiva máquina.

- DESCRIÇÃO: Descrição do SKU.

- MOLAS: Quantidade de molas que possuem esse SKU.

- NIVEL PAR: Capacidade do SKU na mola.

- MARGEM: Margem⁴ calculada para o SKU



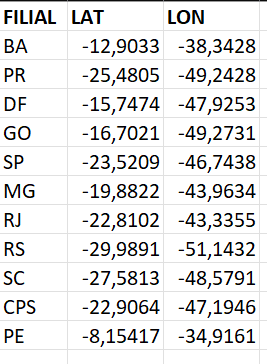
* + 1. “bases”

As colunas presentes são:

- FILIAL: Sigla da filial, como CPS, RJ, SP, BA...

- LAT: Latitude da filial.

- LON: Longitude da Filial



* 1. “Dados de Consumo.csv”: Este arquivo é responsável por indicar o consumo de insumos por patrimônio⁶. Ele é composto por somente uma aba.

As colunas presentes são:

- FILIAL: Sigla da filial, como CPS, RJ, SP, BA...

- PARCEIRO: Nome do parceiro².

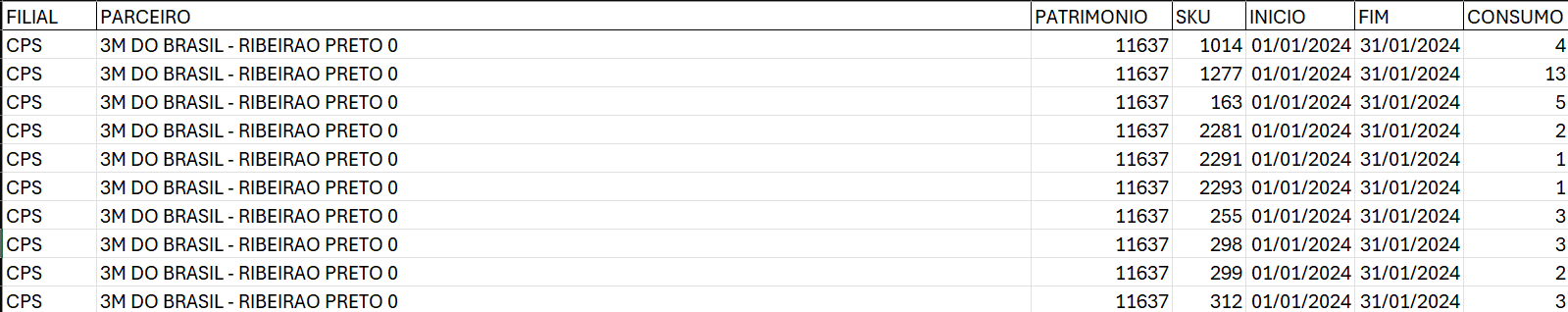
- PATRIMONIO: Nome do patrimônio³.

- SKU: Nome do insumo consumido.

- INICIO: Data do início do mês.

- FIM: Data do fim do mês.

- CONSUMO: Consumo do SKU durante o mês identificado.



* 1. Aba de configurações do arquivo Final

Na Aba de “Configurações Gerais” do arquivo final de resultado, são possíveis inserir configurações que vão se aplicar ao modelo. Elas são:

- Semanas: Total de semanas no mês para realizar a roteirização.

- Dias Considerados por semana: Total de dias na semana para realizar a roteirização.

- Flexibilidade em relação a Frequência Semanal Atual: O valor correspondente à variação possível da frequência final a partir da frequência atual.

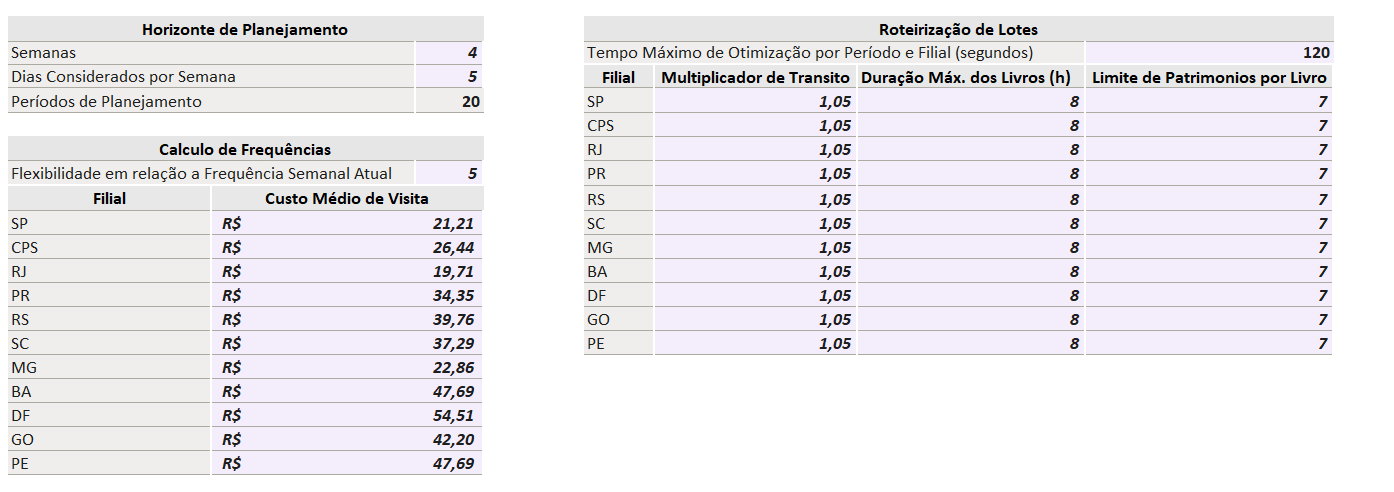
- Tempo Máximo de Otimização por período e Filial (segundos): tempo limite máximo em que o modelo pode estar calculando a solução para cada rota por período e filial.

- Tolerância no tempo total: Tolerância no tempo total da rota, considerando a escala

- Multiplicador de Trânsito: fator multiplicativo de trânsito para cada cidade

- Duração Máx. dos Livros (h): Duração máxima de uma rota para um abastecedor.

- Distância Máx. dos Livros (km): Distância máxima de uma rota para um abastecedor.



**Pontos de atenção:**

¹é obrigatório que as informações entre as quatro abas sejam coerentes. Isto é, é necessário que a quantidade de patrimônios, parceiros e filiais, seja a mesma nas três abas. Por exemplo, caso um patrimônio esteja na aba de “insumos”, porém não esteja na aba de “patrimônios”, haverá erro no código.

² para cada filial, um parceiro deve ter somente uma Latitude e Longitude. Foram identificados casos em que um mesmo parceiro tinha dois patrimônios, sendo cada um em de endereços diferentes. Logo como são coordenadas diferentes, os parceiros devem ser identificados como diferentes. A diferenciação foi feita através de “0”, “1”, “2” ... ao final do nome do parceiro.

³quando um patrimônio possui seu nome composto somente por números, sendo seu primeiro digito “0”, o Excel identifica isso como texto e pede para transformar a célula em número. Ao transformar em número, o “0” é eliminado no início nome. É importante salientar-se que se essa transformação for feita deve ocorrer em todos os locais (Dados.xlsx e Dados de Consumo.csv).

⁴a margem é calculada a partir de dados de Remessa (que ontem o consumo). Dessa forma, se o patrimônio no parceiro atual, não está neste dado, não terá dado de margem. Assim,para conseguir colocar o patrimônio dentro das rotas, é necessário inserir o patrimônio no formato abaixo, como exemplo.



Neste caso, o patrimônio PR01496, no parceiro AZUL LINHAS AÉREAS BRASIELRIAS – CAFÉ 1 não estava presente em Remessas, então foi necessário inserir MARGEM = 0.

⁵considerando a realocação de máquinas e o baixo histórico de consumo, é possível que alguns patrimônios não tenham sido incluídos na planilha "Dados de Consumo.csv". Portanto, não há problema se um patrimônio estiver presente em "Dados.xlsx" mas ausente em "Dados de Consumo.csv"; nesse caso, sua frequência será baseada na frequência semanal mínima imputada.

* 1. Entendimento dos arquivos .py

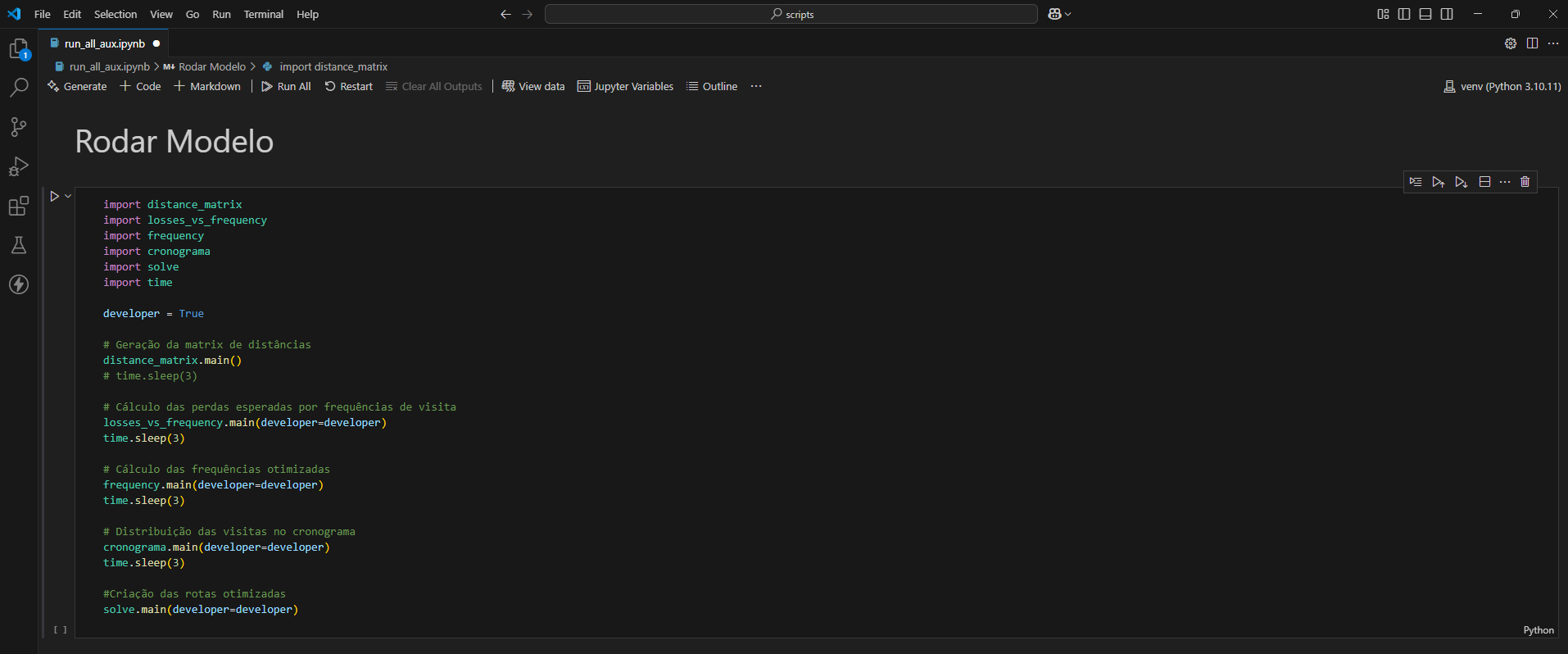
Os códigos devem ser operados de forma modular (na sequência indicada), para obter o resultado. A explicação de cada um encontra-se abaixo.

* + 1. distance\_matrix.py

Este código Python utiliza a biblioteca “openrouteservice” para calcular matrizes de distância e duração entre todos os parceiros. O objetivo principal é obter informações distância e tempo de viagem entre odos os locais identificados na aba “parceiros” do arquivo “Dados.xlsx”. Esse código Python não é necessário de ser rodado sempre, somente, quando houver alguma alteração (mudança de coordenadas, ou adição de parceiros) na própria aba mencionada.

* + - 1. **Carrega Dados:** Lê dados do arquivo Excel ("Dados.xlsx"), na aba “parceiros” contendo informações sobre os parceiros (incluindo suas coordenadas de latitude e longitude).
      2. **Calcula Proximidade:** Utiliza a distância euclidiana para identificar os parceiros mais próximos dentro de um raio definido.
      3. **Realiza Chamadas à API:**
         * Utiliza a chave de API fornecida para se conectar ao openrouteservice.
         * Para cada "chamada" gerada, envia uma requisição para obter a matriz de distâncias e durações entre os pontos de origem e destino especificados, considerando um perfil de transporte (carro ou a pé).
      4. **Salva Resultados Finais:** Salva a matriz de distâncias e durações tratada em um arquivo Parquet ("distance\_matrix.parquet") para cada modal de transporte.
    1. losses\_vs\_frequency.py
       1. **Lê dados:** Configurações, SKUs (Dados.xlsx), consumo (Dados de Consumo.csv).
       2. **Calcula consumo semanal:** Processa datas, agrupa consumo por semana.
       3. **Estima perdas esperadas:** Calcula média, desvio padrão do consumo, aplica distribuição normal para prever perdas em diferentes períodos de reposição.
       4. **Salva resultados:** Consumo histórico, perdas por frequência, perdas por SKU em arquivos .parquet.
       5. **Finaliza:** Imprime tempos de execução e (opcionalmente) fecha o Excel.
    2. frequency.py
       1. **Lê configurações e dados:** Custos de visita, informações de locais, perdas esperadas.
       2. **Calcula perdas e custos:** Para várias frequências de visita por local.
       3. **Considera restrições:** Frequência mínima/máxima, limites de itens vazios.
       4. **Otimiza frequência:** Escolhe a frequência com menor custo total (perdas + visitas) por local.
       5. **Salva resultados:** Frequência ideal e dados relacionados em aba do Excel.
    3. cronograma.py,
       1. **Lê dados de frequência:** Obtém as frequências de visita calculadas previamente para cada local.
       2. **Gera padrões de visita:** Cria diferentes maneiras de distribuir as visitas ao longo das semanas para cada frequência.
       3. **Otimiza o cronograma:** Utiliza um solver de programação linear (ortools) para definir um cronograma de visitas que minimize o máximo de visitas em um único dia, garantindo que cada local seja visitado na frequência correta.
       4. **Salva o cronograma:** Escreve o cronograma otimizado em uma aba específica ("Cronograma") do arquivo Excel, indicando em quais dias cada local deve ser visitado.
    4. solve.py
       1. **Importa dados e configurações:** Pontos de referência, parâmetros de otimização (tempo, capacidade), frequências de visita, cronograma existente e matriz de distâncias/tempos.
       2. **Prepara dados para o modelo:** Cria um objeto Lote para cada filial, definindo clientes, base (depósito), capacidades, janelas de tempo, tempos de serviço, demandas e matrizes de distância/tempo.
       3. **Remove pontos inviáveis:** Identifica e remove clientes cuja visita excederia o tempo máximo permitido.
       4. **Resolve o problema de roteamento:** Utiliza um solver de otimização (ortools) para encontrar rotas eficientes para um único veículo, respeitando as janelas de tempo e minimizando o custo (penalidade por usar rotas).
       5. **Organiza e retorna os resultados:** Formata as rotas encontradas para cada filial, indicando a sequência de visitas.
  1. Operação do código

A operação do modelo baseia-se no arquivo run\_all\_aux.ipynb, no qual concatena todos os arquivos necessários explicados na seção anterior. Caso queira rodar somente distance\_matrix, ou algum outro arquivo, basta comentar as linhas dos outros arquivos.



Caso seja necessário reduzir a quantidade de patrimônios para roteirização ou ajustar algum parâmetro de configuração, basta incluir no arquivo “Dados.xlsx” apenas os patrimônios relevantes para esse escopo.

Por exemplo, se a equipe de Operações de OTG quiser realizar a roteirização apenas para as rotas noturnas, deve-se incluir no arquivo “Dados.xlsx” somente os patrimônios que se enquadram nisso. Portanto, dessa maneira podem ser criados “grupos de roteirização" para que rotas específicas não se misture com outras.