**DATOS**

*nota: aquí job no se refiere a la instancia de corrida del modelo en la plataforma, sino a la orden de trabajo*

**INPUT**

Aún resta que Claudio dé el OK a la planilla de datos de input. Pero avanzamos con lo que tenemos.

Por un lado, en el **Excel**, se detallan los siguientes datos para cada OT (hoja OT):

|  |  |  |
| --- | --- | --- |
|  | **Campo** | **Dato** |
| 1 | OT (nro) | int |
| 2 | Tipo Producto | string |
| 3 | Color | String |
| 4 | Peso | Int |
| 5 | ID producto | Int |
| 6 | Máquina (nombre) | String |
| 7 | Máquina (nro en Gantt) | Int |
| 8 | Cantidad planificada | Int |
| 9 | Horas de trabajo estimadas | Int |
| 10 | Fecha vencimiento | Date |
| 11 | Cadencia de maquina | int |
| 12 | Requerimiento de Operarios | Int |
| 13 | Prioridad | int |

En hoja Setup:

|  |  |  |
| --- | --- | --- |
|  | **Campo** | **Dato** |
| 1 | Tiempo Set Up (horas) | float |

En hoja FueraServicioMaquina:

|  |  |  |
| --- | --- | --- |
|  | **Campo** | **Dato** |
| 1 | Máquina | int |
| 2 | Hora Inicio | Date-Time |
| 3 | Hora Fin | Date-Time |

A agregar: Hoja Operarios

Se puede definir en una celda, un entero que represente el número de operarios disponibles.

|  |  |  |
| --- | --- | --- |
|  | **Campo** | **Dato** |
| 1 | Cant. operarios disponibles | int |

Captura de pantalla de planilla en Hoja OT (a modo ilustrativo):



Por otra parte, en el **modelo .py** tenemos:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Ejemplo** | **Dato tipo** | **Descrip.** |
| jobs | ['j1', 'j2', 'j3'] | List | Conjunto de OTs |
| ope | ['o1', 'o2', 'o3'] | List | Conjunto de operarios |
| maq | ['m1', 'm2'] | List | Conjunto de máquinas |
| asignacion | {'j1': 'm1', 'j2': 'm1', 'j3': 'm2'} | Dict | Máquina (en “maq”) asociada/asignada a cada OT (en “jobs”) |
| operarios | {'j1': 2, 'j2': 1, 'j3': 2} | Dict | Entero que representa cantidad de operarios requeridos por la OT. |
| prioridad | {'j1': 1, 'j2': 2, 'j3': 3} | Dict | Entero que representa el orden de prioridad de cada OT. |
| proc | {'j1': 10, 'j2': 9, 'j3': 8} | Dict | Entero que representa el tpo de procesamiento de cada OT. |
| dd | {'j1': 9, 'j2': 57, 'j3': 27} | Dict | Entero que representa las hora de vencimiento/entrega de una OT. |
| changeover | 0.5 | Float | Tiempo de changeover para todo par de OTs. |
| fueraservicio | ['m5'] | List | Máquinas fuera de servicio |
| tiinactiva | {'m5': 60} | Dict | Tpo inicio de fuera de servicio, en hs (int) desde inicio de agenda. |
| tfinactiva | {'m5': 70} | Dict | Tpo fin de fuera de servicio, en hs (int) desde inicio de agenda. |
| bigM | 300 | Int | Bib M, parámetro del modelo |

El modelo está importando Pandas, pero si fuera necesario, se pueden emplear otras librerías también.

**MATCHEO**

Esquemitas de origen (Excel) y destino (modelo .py) de datos, con aclaraciones cuando amerita:

Datos desde Hoja OT:



OTs: Los campos Tipo Prod., Id Prod, Color, Peso, pueden tomarse como atributos de cada OT. Opción: que en “Jobs”, cada elemento sea una tupla (o set) con todos estos atributos (cada elemento representa a cada OT). Más allá de los datos descriptivos, es relevante que cada OT represente un elemento del conjunto Jobs.

Si bien la OT está representada con un número en planilla, puede tomarse como string en el modelo.

Máquinas: Sería posible generar la asignación desde el campo “máquina nro Gantt” directamente, dejando el campo con la descripción del nombre del equipo como algo informativo, no como parte relevante del modelo. Similar al caso de OTs y sus atributos.

La lista “maq” puede generarse a partir del campo “máquina nro Gantt”, sin duplicados. Si bien la maq está representada con un número en planilla, puede tomarse como string en el modelo.

Procesamiento: El tiempo de procesamiento está calculado en la misma planilla, a partir de otros dos campos (Cadencia y Cant Planific.).

Duedates: en el modelo, las duedates (dd) están representadas en horas desde el inicio de la agenda, mientras que en la planilla es una fecha. En primera instancia, para que tenga un sentido, a la fecha debería agregarse una hora. Entonces, habría que realizar la diferencia entre el nro real que representa la fecha-hora (dd) y el nro real que representa la fecha-hora en el que inicia la agenda. Esto se puede hacer en excel oculto y tomar el dato desde allí (o hacerlo en script, como sea mejor). Ese dato, un entero que representa horas, es el que se requiere en “dd” en modelo. De todas maneras, no es un dato que se esté usando, por lo que lo dejaría para el último (no lo consideraría en una primer instancia del demo).

BigM: Es un dato interno del modelo, que no debería ser accesible al usuario, pero que de todas maneras es susceptible de ser modificado y ajustado por equipo de I+D.

Hoja SetUp:



Hoja FueraServicioMaquina:



Horas inicio y fin: Dato en excel en fecha-hora, y en int en modelo. Hay que realizar la diferencia entre el real que representa la fecha-hora (inicio y fin) y el real que representa la fecha-hora en el que inicia la agenda. Esto se puede hacer en excel oculto y tomar el dato desde allí (o hacerlo en script, como sea más fácil/mejor).

Hoja Operarios:



Hoja a agregar. La lista “ope” surgiría de generar una lista de elementos con tope en el dato numérico ingresado en el campo “cant. operarios disponibles” en el Excel (rango y de rango a lista?).

**PROCESO (respeto a datos)**

Esto es una idea desde una lógica “convencional”. Replantear en caso de mejor alternativa.

Inicialización de datos:

Los datos se inicializarían desde dentro del mismo modelo .py.

Opción 1: El llamado de inicialización referenciaría a datos en el archivo de Excel que se esté empleando para la corrida que se va a realizar.

Opción 2: El llamado de inicialización referenciaría a datos en un archivo temporal, posiblemente con estructura JSON, generado a partir de la planilla de Excel.

Conformación de la corrida:

1. Configurar los datos para la corrida en la planilla
2. Una vez definidos los datos, botón en plataforma inicia corrida.
3. Ese botón, dispara una instancia de preprocesamiento de datos, previo a la corrida del modelo en sí mismo.
4. Genera los datos a partir de la planilla, según se elija opción 1 u opción 2.
5. Deja disponible esos datos para ser llamados desde las sentencias de inicialización dentro del modelo.
6. Inicia ejecución del modelo, el cual en las primeras líneas tiene las sentencias de inicialización de listas y demás.

Hoy el modelo tiene los datos embebidos. Para generar en el modelo las sentencias de inicialización que tomarán datos desde una fuente externa, primero hay que definir cuál será esa fuente (planilla? temporal file?) y su estructura.