Data for
Development Impact

Manejo y Limpieza de Datos

Rony Rodriguez-Ramírez

January 17, 2021

LAMBDA

Manejo de Datos

Pensemos en replicabilidad

El fin de esta sesión es asegurar que nuestras investigación sea reproducible.

- Publicar un artículo por sí solo ya no es suficiente:
  - Al igual que las tablas y las figuras, el código ahora es un resultado igualmente importante para compartir. En este contexto, publicar código no tiene sentido si es:
    1. no reproducible.
    2. nadie puede entender como se corre.
  - Para que el código sea útil, requiere transparencia, responsabilidad y un flujo de trabajo fácil de entender.
  - Básicamente, el código debe ser organizado y legible.

Esto es mucho más fácil decirlo que hacerlo

- En DIME, tenemos grandes equipos que colaboran en los mismos códigos y conjuntos de datos.
- Los proyectos grandes se vuelven fácilmente complejos ya que tienen múltiples rondas / fuentes de datos que deben organizarse.
- La estandarización de la organización de documentos y códigos previene errores y reduce el costo de la transición entre proyectos y equipos.

¿A qué nos referimos con gestión de datos?

En esta sesión entenderemos e implementaremos las mejores prácticas para gestionar el trabajo de datos a través de:

- Configurar una buena estructura de carpetas
- Crear un script maestro que ejecute todo el código
- Establecer un sistema de control de versiones

Cuando los contenidos de esta sesión se aplican a un proyecto, cualquier persona con acceso completo a sus archivos y carpetas podrá replicar la investigación y comprender la estructura del trabajo de datos.

Estructura de las carpetas

¿Por qué nos debería importar la estructura de nuestros folderes?

- La carpeta de su proyecto probablemente tenga muchas subcarpetas para literatura, presentaciones, notas conceptuales y otros documentos.
- En esta sesión, nos centraremos en las carpetas relacionadas con el trabajo de datos. Llamaremos al conjunto de carpetas relacionadas con datos la carpeta DataWork.

¿Por qué nos debería importar la estructura de nuestros folderes?

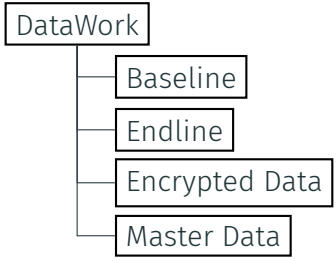
- El paquete `ietoolkit` Stata ofrece un comando llamado `iefolder` que ayuda a configurar la estructura de carpetas y sus interacciones con los archivos de código.
- `Iefolder` proporciona una plantilla para la estructura de carpetas para un proyecto DIME típico utilizando datos primarios.
- No entraremos en detalles sobre cómo usar el comando `iefolder` aquí, sino que nos centraremos en los principios detrás de él.
- Para obtener documentación y detalles sobre cómo usar el comando, escriba `help iefolder` en Stata.

¿Por qué nos debería importar la estructura de nuestros folderes?

- La motivación detrás de iefolder es crear una estructura estandarizada que sea fácil de navegar para los miembros del equipo.
- Los diferentes proyectos pueden tener necesidades específicas, pero las plantillas utilizadas en iefolder son un buen punto de partida para pensar en la estructura de carpetas de cualquier proyecto.
- Sin embargo, sea cual sea la estructura que esté utilizando, aplicarla a todos sus proyectos hará que sea más fácil moverse entre proyectos.

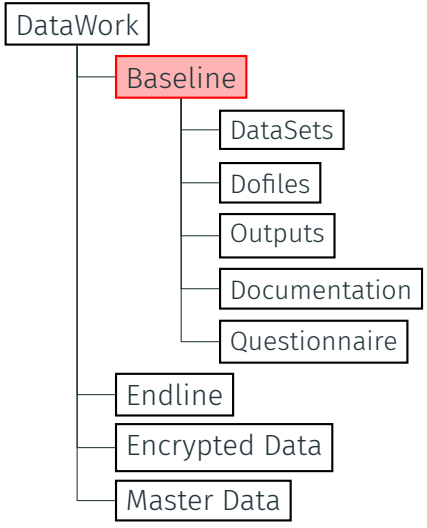
DataWork Folder: Visión General

- Así es como se ve la carpeta `DataWork` creada por `iefolder`:



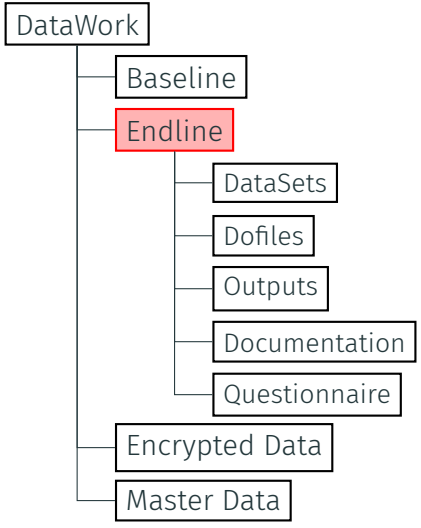
DataWork Folder: Visión General

- La carpeta de línea de base (baseline) almacenará todos los datos de línea de base, así como los archivos do y las salidas que se refieren exclusivamente a esta ronda de recopilación de datos.



DataWork Folder: Visión General

- La carpeta de la línea final almacenará todos los datos de la línea final, así como el código y las salidas que se refieren exclusivamente a esta ronda de recopilación de datos.
- Tenga en cuenta que su estructura es exactamente la misma que la estructura de la carpeta de línea de base.



DataWork: Round Folders

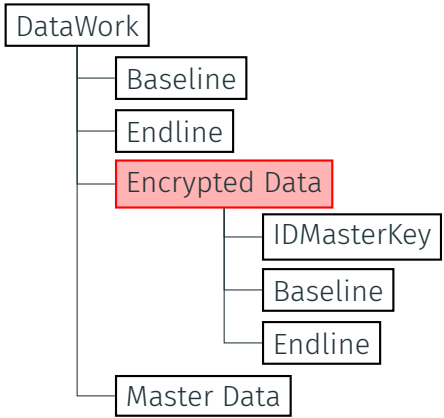
- Si bien una ronda de recopilación de datos solo puede parecer aplicable a los datos primarios, puede pensar en una “ronda” como una fuente de recopilación de datos.
- Otra forma de decirlo es pensar en una “ronda” como un conjunto de datos que se procesarán con el mismo código

DataWork: Round Folders

- Un ejemplo de esto podría ser una recopilación de datos primarios que incluye dos niveles de observación (hogar y comunidad, estudiante y escuela, paciente y médico, etc.): cada nivel probablemente tendrá su propio cuestionario y se limpiará por separado.
- Por lo tanto, creará diferentes carpetas, como `HouseholdBaseline` y `CommunityBaseline`.
- Otro ejemplo es cuando el mismo cuestionario se aplica dos veces, pero los nombres de las variables y las etiquetas de valor son ligeramente diferentes. Entonces también necesitarás dos carpetas separadas.

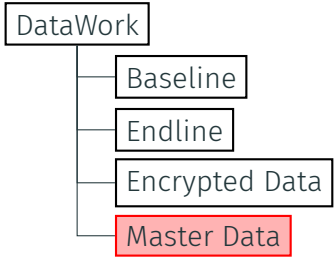
DataWork: Carpeta encryptada (datos cifrados)

- La carpeta de datos cifrados contendrá todos los datos de identificación personal para cada ronda de recopilación de datos, y una carpeta con claves maestras de identificación que vincula cada identificación no identificada a las observaciones identificadas.
- Como su nombre indica, esta carpeta debe estar encriptada.



DataWork: Master Data

- La carpeta de datos maestros (master data) almacenará los conjuntos de datos maestros para cada unidad de observación en su proyecto.



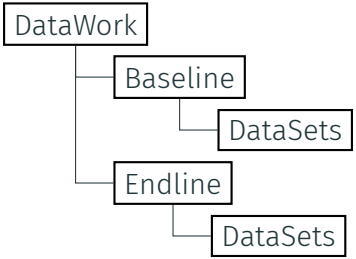
¿Qué es el master data?

- Una lista completa o lista completa de todas las observaciones potenciales que uno puede encontrar durante el curso de un proyecto.
- Forma una documentación exhaustiva de las acciones tomadas para cada unidad de observación en el alcance del proyecto.
- Un ejemplo: un conjunto de datos maestros del hogar incluirá:
  - hogares enumerados en el censo.
  - hogares muestreados para las encuestas.
  - hogares incluidos en el monitoreo (incluso si no son parte del proyecto).
  - hogares incluidos en el análisis.
  - una identificación única para cada uno de estos hogares.

Usando la carpeta de DataWork

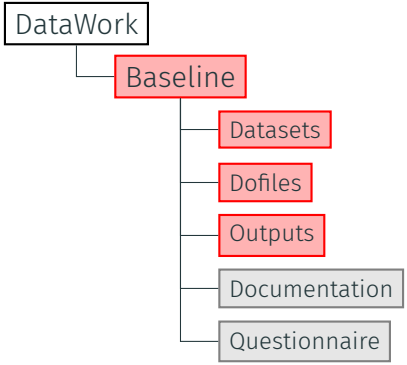
Base de datos

- Todas las bases de datos a ocupar deben de estar de-idenficadas.



Base de datos

- Las secuencias de comandos en cada carpeta cargarán datos de la carpeta de bases de datos de esa ronda y almacenarán cualquier salida en la carpeta de salidas de la ronda.



Master Script

¿Por qué es necesario un script maestro?

- Como habrás notado, mencionamos la creación de muchos scripts de código en las diapositivas anteriores.
- Un gran proyecto puede volverse muy complejo, y las secuencias de comandos deben ejecutarse en un cierto orden para crear la salida correcta.

¿Por qué es necesario un script maestro?

- Eso podría significar que necesitaría escribir una secuencia de comandos extremadamente larga o un documento diferente con instrucciones sobre en qué orden ejecutar todas las secuencias de comandos.
- Sin embargo, puede crear un script que ejecute otros scripts.
- Esto facilita que cualquiera que reproduzca su código lo haga con facilidad.

¿Qué es un script maestro?

- El script maestro es el mapa sobre todo el trabajo de datos en su carpeta de datos
- Es la tabla de contenido para las instrucciones que codifica
- Debería ser posible seguir todo el trabajo de datos en la carpeta de datos, desde datos sin procesar hasta resultados de análisis, leyendo el script maestro.

Script maestro: permite una colaboración fácil

- Si compartimos un proyecto a través de DropBox o GitHub, todos los miembros del equipo tienen la misma estructura de carpetas.
- Un script maestro permite que varias personas establezcan su propio global en la carpeta del proyecto.
- De esta manera, cualquiera que comparta la carpeta del proyecto puede ejecutar fácilmente sus scripts.

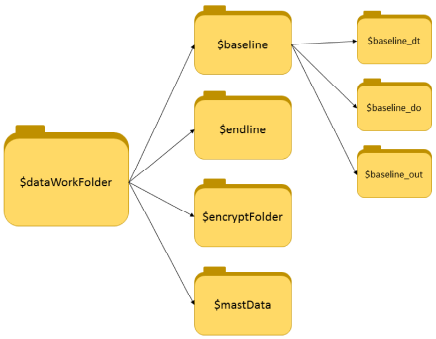
Script maestro: Connexión entre código y estructura de carpetas

- El script maestro contiene `globals` u objetos que hacen referencia a la subcarpeta en la carpeta `DataWork`, por lo que tiene accesos directos fáciles para ellos.
- Cualquier cambio en la estructura de carpetas puede explicarse fácilmente cambiando el global de la carpeta.

Script maestro: Connexión entre código y estructura de carpetas

```
    * Project folder globals
    * ----------------------

    global dataWorkFolder          "$projectfolder/DataWork"

*iefolder*1*FolderGlobals*master************************************************
*iefolder will not work properly if the line above is edited

    global mastData                "$dataWorkFolder/MasterData"

*iefolder*1*FolderGlobals*rawData************************************************
*iefolder will not work properly if the line above is edited

    global encryptFolder           "$dataWorkFolder/EncryptedData"
    global masterIdDataSets        "$encryptFolder/IDMasterKey"


*iefolder*1*RoundGlobals*rounds*baseline*****************************************
*iefolder will not work properly if the line above is edited

    *baseline folder globals
    global baseline                  "$dataWorkFolder/baseline"
    global baseline_dt               "$baseline/DataSets"
    global baseline_do               "$baseline/Dofiles"
    global baseline_out              "$baseline/Output"
```

(a) Master do-file



(b) Folder structure

Script maestro: Permite actualizaciones fáciles

- Las entradas del usuario y la configuración global deben definirse en el script maestro.
- Ejemplos de esto incluyen rutas de carpeta, tasas de conversión, control y variables de resultado, e incluso colores de gráficos
- Esto le permite realizar cambios en una sola línea de código a través de un objeto global o cuando quiera aplicar una actualización a todos sus códigos.
- Si está utilizando Stata, los globales solo deben definirse en el do file maestro.

Script maestro: Permite replicaciones fáciles

- Cualquiera debería poder seguir y reproducir todo su trabajo desde los datos en bruto a todas las salidas con un clic en este script, después de agregar solo la ruta de la carpeta raíz.
- En general, siempre debe ejecutar códigos a través del script maestro Esto evita el flujo de trabajo muy común de ejecutar este script, luego este y finalmente el otro.
- También le ayuda a asegurarse de que los cambios que realice en un código no rompan otros códigos.

Script maestro: El mapa a todo el trabajo de datos

- Al leer el script maestro, alguien externo al proyecto debe tener una comprensión general de lo que se está haciendo en cada paso
- Si desea ver cómo se creó o creó una tabla o conjunto de datos en particular, leer el script maestro debería ser suficiente para decir qué script mirar.
- El uso de locales y objetos para crear interruptores para seleccionar qué partes del proyecto ejecutar o no facilitar el uso del código cuando los proyectos son muy largos y complejos.

Prácticas recomendadas para rutas de archivos

```
*** Dinamico, Paths absolutos
    // Dinamico y absoluto = GOOD
    global myDocs       "C:/users/username/Documents"
    global myProject    "${MyDocs}/MyProject"

    use "${myProject}/MyDataset.dta", clear


*** Relativo y absoluto
    // Relativo = BAD
    cd "C:/users/username/Documents/MyProject"
    use MyDataset.dta

    // Absoluto pero no dinamico = BAD
    use "C:/users/username/Documents/MyProject/MyDataset.dta"
```