

Practical session: Data Preparation



UNU
WIDER

Break out groups:

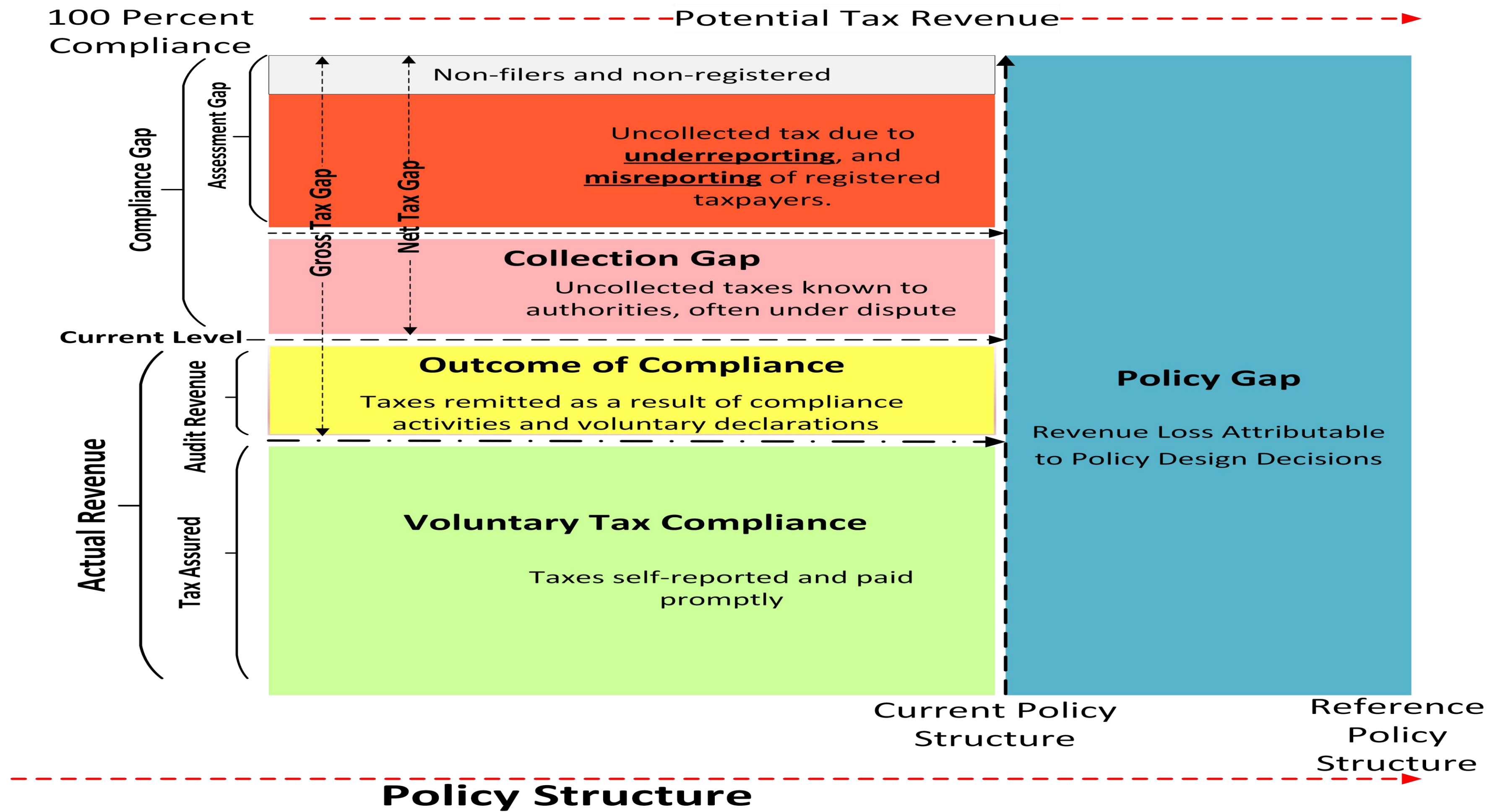
Maria Jouste + Gerald Agaba

Amina Ebrahim + Mostafa Bahbah

Kwabena Adu-Ababio + Hilja-Maria

2024 ATI Tax Gap Estimation Toolkit Training | October 7-8, 2024

Level of compliance



Summary

Key Learnings So Far:

- ❖ What is a tax gap
- ❖ Why estimate the tax gap
- ❖ Benefits and limitations of the Bottom-up approach
- ❖ Importance of Random Audits
- ❖ What are the data requirements for the tax gap estimation
- ❖ How to prepare data for tax gap estimation

Aim of this session

1. How to prepare data for tax gap estimation
2. Show the steps taken to prepare the data in Stata before the estimation (Toolkit Part 1)

We are not teaching how to use Stata, merely guiding the steps to follow.

Own implementation requires contextualizing.

Exercise 1 – Questions - Acquiring the correct data

In this exercise, you will try to implement the first stage of the data cleaning part of the toolkit.

- a) Run the code lines 18 to 23 to prepare the Stata environment.
- b) Select a location on your computer and create a folder named “Cleaning”, then create six subfolders within the “Cleaning” folder with the following names: Excel, Data, Graphs, Dofile, Results, and Log. Once you have created these folders, set the directory paths.
- c) How would you modify the following line of code?

`global dir "<YOUR PATH>"`

Exercise 1 – Questions continued

d) Place the file named “train_ex1” in the Excel folder you just created and then implement the following:

- Adjust code line 39 to import the “train_ex1” data.

Hint: This file is delimited text in CSV format, so you have to make the extension as “csv” instead of “dsv”.

- Format the tax_id_variable, rename variables (region, isci, year) to (province, isic_code, rtn_year) respectively.
- Save this data in Stata format and name it “cit”.

Exercise 1 – Solutions

c) How would you modify the following line of code?

```
24  
25 global dir "D:\work\ATI toolkit\Training\Training\Training\Online training\Testing Data"  
26
```

d) Place the file named “train_ex1” in the Excel folder you just created and then implement the following:

- Adjust code line 39 to import the “train_ex1” data.

```
37 import delimited "$xlsout\train_ex1.csv",clear
```

Exercise 1 – Solutions continued

d) Place the file named “train_ex1” in the Excel folder you just created and then implement the following:

- Format the tax_id_variable, rename variables (region, isci, year) to (province, isic_code, rtn_year) respectively.

```
39  
40 format tpin %15.0g  
41  
42  
43 * Hint:  
44 *rename old_name new_name  
45 rename region province  
46 rename isci isic_code  
47 rename year rtn_year
```

- Save this data in Stata format and name it “cit”.

```
50 save "$data\cit.dta", replace
```

Exercise 2 – Questions - Dealing with Duplicates

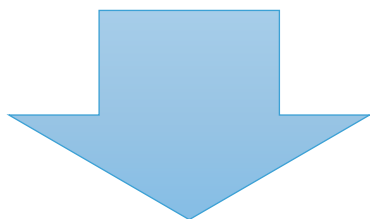
In this exercise, you will try to implement the second stage of the data cleaning part of the toolkit.

- a) How many duplicates exist in the CIT return data, identified by the return year and tax identification number? Please provide a report summarizing the number of observations with duplicates, categorized by the number of duplicate copies.
- b) Create a variable (dup0) that tags all multiple/duplicated return_year and tax_id pairs.
 - How many duplicated pairs can be found?
- c) Check if the variables that you have including the new variables are duplicated.
- d) Note the comment in line 79 and run from lines 80 to 86 to drop all duplicates.
- e) Sort and order by duplicated return_year and tax_id variable, then save the Data

Exercise 2 - Solution

a) How many duplicates exist in the CIT return data, identified by the return year and tax identification number? Please provide a report summarizing the number of observations with duplicates, categorized by the number of duplicate copies.

```
58
59 duplicates report rtn_year tpin
60
```



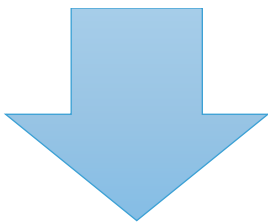
Duplicates in terms of rtn_year tpin		
Copies	Observations	Surplus
12	387228	354959
13	269100	248400
14	81970	76115
15	15810	14756
16	1808	1695
17	136	128
18	18	17

Exercise 2 – Solutions continued

b) Create a variable (dup0) that tags all multiple/duplicated return_year and tax_id pairs.

- How many duplicated pairs can be found?

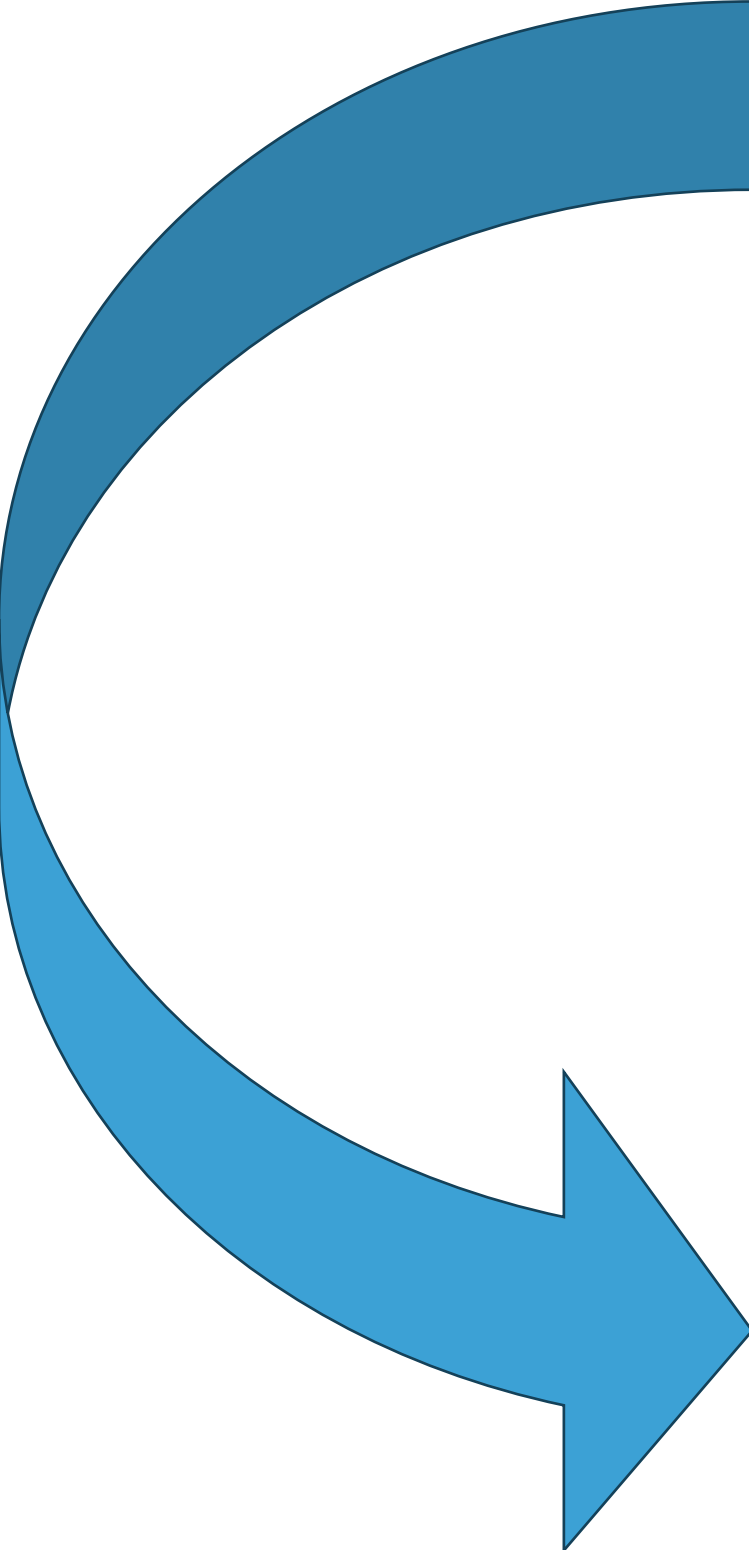
```
64  
65 duplicates tag rtn_year tpin , gen(dup0)  
66  
67 tabulate dup0
```



dup0	Freq.	Percent	Cum.
11	387,228	51.22	51.22
12	269,100	35.59	86.81
13	81,970	10.84	97.65
14	15,810	2.09	99.74
15	1,808	0.24	99.98
16	136	0.02	100.00
17	18	0.00	100.00
Total	756,070	100.00	

Exercise 2 – Solutions continued

c) Check if the variables that you have including the new variables are duplicated.



69
70
71

`browse rtn_year tpin accumulateddepreciation provisions totalfixedassets dup0 if dup0 != 0`

Data Editor (Browse) - [cit]

File Edit View Data Tools

rtn_year[1]

2001

	rtn_year	tpin	accumulateddepreciation	provisions	totalfixedassets	dup0
1	2001	1251651	491	230	964	14
2	2001	1251651	369	42	963	14
3	2001	1251651	382	1	479	14
4	2002	1251651	166	337	851	13
5	2002	1251651	468	82	556	13
6	2003	1251651	457	441	242	12
7	2007	1251651	195	50	692	12
8	2008	1251651	135	495	370	12
9	2009	1251651	297	153	446	12
10	2010	1251651	200	12	778	12
11	2001	185442	255	311	440	12
12	2005	185442	64	228	467	13
13	2005	185442	244	413	550	13
14	2008	185442	188	123	354	12
15	2005	39183	97	120	336	12
16	2007	39183	496	146	262	12
17	2009	39183	276	488	624	12
18	2003	2191704	417	6	197	12
19	2008	2191704	318	144	497	13
20	2008	2191704	54	91	566	13
21	2009	2191704	185	284	278	12
22	2010	2191704	226	216	725	13
23	2010	2191704	249	345	770	13
24	2002	2841035	384	275	769	14

Variables

Filter variables here

☒ Name

Label

☒ rtn_year

☒ tpin

☒ accumulateddepreciation

☒ provisions

☒ totalfixedassets

☒ dup0

☐ month

Variables

Snapshots

Properties

Variables

Name

rtn_year

Label

Type

int

Format

%8.0g

Value label

Notes

Data

Frame

default

Filename

cit.dta

Label

Notes

Ready

Vars: 6 of 28 Order: Dataset Obs: 756,070 of 756,070 Filter: On Mode: Browse CAP NUM

Exercise 2 – Solutions continued

d) Note the comment in line 79 and run from lines 80 to 86 to drop all duplicates.

```
74 // Note: Tax authorities sometimes make multiple reviews or assessments. In this context, a duplicate refers to multiple records for the same taxpayer (tpin) and year  
75 (rtn_year). We want to keep the most updated pre-audit return and discard older versions or reviewed/assessed returns.  
76 bys tpin: egen latest0 = max(rtn_year) if dup0 != 0  
77 drop if rtn_year != latest0 & dup0 != 0  
78  
79 duplicates drop rtn_year tpin, force  
80 duplicates tag rtn_year tpin, gen(dup1)  
81 assert dup1 == 0  
82 drop dup0 dup1 latest0
```

e) Sort and order by duplicated return_year and tax_id variable, then save the Data

```
84  
85 sort rtn_year tpin  
86 order rtn_year tpin  
87
```

Exercise 3 – Questions - Preparing Audit data

In this exercise, you will try to implement the third stage of the data cleaning part of the toolkit.

a) Here similar to what has been done for the tax return data, you have to import and rename some variables and save audit data in a Stata format file by doing the following:

a1. Place the audit_data file in the Excel folder, and then import this data by adjusting the code line number 106.

Hint: This file is delimited text in CSV format, so you have to make the extension as “csv” instead of “dsv”.

a2. Rename the variables totaltax to totalassessments and audittype to AuditType, and then format the tpin variable to have a fixed width of 15 characters with no decimal places.

a3. Save audit_data in the Data folder in Stata format and name it “audit”.

Exercise 3 – Questions continued

b) Adjusting Total Assessments for Annual Audit Periods

b1. Set any negative values in the total assessments variable to zero.

b2. Why are negative values in total assessments set to zero when analyzing tax evasion?

b3. Allocate total assessments across the audit periods by counting non-missing period variables and dividing total assessments by this count to create total assessments_annual.

Hint: Only edit line 130.

Exercise 3 – Questions continued

c) Sorting and Dropping Duplicates Based on Assessment Date

In managing audit return data, duplicates can distort analysis by inflating or skewing results. A duplicate in this context refers to multiple records for the same taxpayer ID and return year that may represent different audit assessments. To ensure data integrity, it's crucial to identify and retain only the most relevant records. This exercise is about handling duplicates by leveraging the `assessment_date`, which indicates the final assessment made by the Revenue Authority for a firm in a particular tax category.

c1. Use the `gsort` command to sort the dataset by `tax_id_variable`, `return_year_variable`, `tax_head_variable`, `initial_period`, `final_period`, `inter_period_1`, `inter_period_2`, and `assessment_date_variable` in descending order of `assessment_date`.

Hint: Edit line 142 with the actual variables.

Exercise 3 – Questions continued

c2. Remove duplicate records based on tpin, rtn_year, tax_head, initial_period, final_period, inter_period_1, and inter_period_2, keeping only the latest assessment_date.

c3. Sort the dataset by tax_id_variable, return_year_variable, tax_head_variable, initial_period, final_period, 1st intermediate period, and 2nd intermediate period, then order these variables accordingly.

c4. Check duplicates, tag any remaining duplicates, and browse for identifications

c5. Remove all identified duplicates, ensuring only unique records remain.

Exercise 3 – Questions continued

- d) Encode the AuditType variable into auditcode, recode auditcode into audittype with the specified mappings, assign descriptive labels to audittype, generate a frequency table of audittype, order the variables tpin, initial_period, final_period, inter_period_1, inter_period_2, audittype, and totalassessments_annual, and save the cleaned dataset as audit.dta in the Data folder.

Exercise 3 – Solutions

a1. Place the audit_data file in the Excel folder, and then import this data by adjusting the code line number 106.

```
98  
99 import delimited "$xlsout\audit_data.csv", clear  
100
```

a2. Rename the variables totaltax to totalassessments and audittype to AuditType, and then format the "tax_id_variable" to have a fixed width of 15 characters with no decimal places.

```
102  
103 * rename old_name new_name // Hint  
104 rename totaltax totalassessments  
105 rename audittype AuditType  
106  
107 format tpin %15.0g  
108
```

a3. Save audit_data in the Data folder in Stata format and name it “audit”.

```
110  
111 save "$data\audit.dta", replace  
112
```

Exercise 3 – Solutions continued

b) Adjusting Total Assessments for Annual Audit Periods

b1. Set any negative values in the total assessments variable to zero.

```
117  
118 replace totalassessments = 0 if totalassessments < 0  
119
```

b2. Why are negative values in total assessments set to zero when analyzing tax evasion?

Negative values in tax assessments usually represent overpayments, refunds, or adjustments. They are unimportant in tax evasion analysis because tax evasion focuses on underreported income or inflated deductions, not excess payments or credits.

Exercise 3 – Solutions continued

b3. Allocate total assessments across the audit periods by counting non-missing period variables and dividing total assessments by this count to create total assessments_annual.

```
121  
122 egen total_audit_period = rownonmiss(initial_period final_period  
   ↵ inter_period_1 inter_period_2)  
123 gen totalassessments_annual = totalassessments / total_audit_period  
124
```

Exercise 3 – Solutions continued

c1. Use the gsort command to sort the dataset by tax_id_variable, return_year_variable, tax_head_variable, initial_period, final_period, inter_period_1, inter_period_2, and assessment_date_variable in descending order of assessment_date.

```
134  
135 gsort +tpin +rtn_year +tax_head +initial_period +final_period +  
   ↵ inter_period_1 +inter_period_2 -assessment_date  
136
```

c2. Remove duplicate records based on tax_id_variable, , return_year_variable, tax_head_variable, initial_period, final_period, inter_period_1, and inter_period_2, keeping only the latest assessment_date.

```
138  
139 duplicates drop tpin rtn_year tax_head initial_period final_period  
   ↵ inter_period_1 inter_period_2, force  
140
```

Exercise 3 – Solutions continued

c3. Sort the dataset by tax_id_variable, return_year_variable, tax_head_variable, initial_period, final_period, 1st intermediate period, and 2nd intermediate period, then order these variables accordingly.

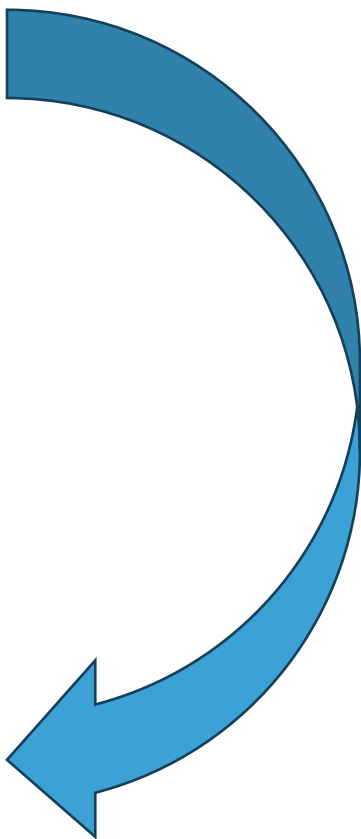
```
145 sort rtn_year tpin initial_period final_period inter_period_1 inter_period_2
146 order rtn_year tpin initial_period final_period inter_period_1
    inter_period_2
147
```

c4. Check duplicates, tag any remaining duplicates, and browse for identifications

```
151 duplicates report rtn_year tpin initial_period final_period inter_period_1
    inter period 2
```

Duplicates in terms of rtn_year tpin initial_period final_period inter_period_1 inter_period_2

Copies	Observations	Surplus
1	640572	0
2	57302	28651
3	6810	4540



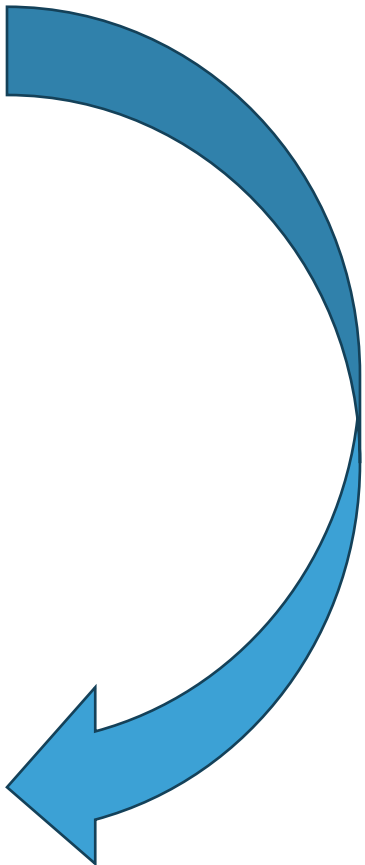
c5. Remove all identified duplicates, ensuring only unique records remain.

Exercise 3 – Solutions continued

c4. Check duplicates, tag any remaining duplicates, and browse for identifications

```
152 duplicates tag rtn_year tpin initial_period final_period inter_period_1
    ↵ inter_period_2, gen(dup1)
153 tabulate dup1
```

dup1	Freq.	Percent	Cum.
0	640,572	90.90	90.90
1	57,302	8.13	99.03
2	6,810	0.97	100.00
Total	704,684	100.00	



Exercise 3 – Solutions continued

c5. Remove all identified duplicates, ensuring only unique records remain.

```
158 duplicates drop rtn_year tpin initial_period final_period inter_period_1
    inter_period_2, force
```



Duplicates in terms of rtn_year tpin initial_period final_period inter_period_1 inter_period_2
(33,191 observations deleted)

```
159 duplicates report rtn_year tpin initial_period final_period inter_period_1
    inter_period_2
160 drop dup1
```



Duplicates in terms of rtn_year tpin initial_period final_period inter_period_1 inter_period_2

Copies	Observations	Surplus
1	671493	0

Exercise 3 – Solutions continued

- d) Encode the AuditType variable into auditcode, recode auditcode into auditttype with the specified mappings, assign descriptive labels to auditttype, generate a frequency table of auditttype, order the variables tpin, initial_period, final_period, inter_period_1, inter_period_2, auditttype, and totalassessments_annual, and save the cleaned dataset as audit.dta in the Data folder.

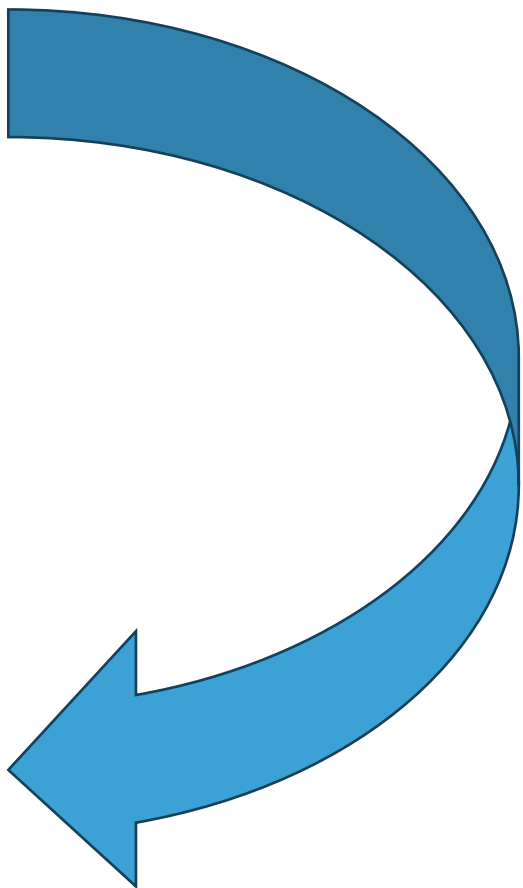
```
164 encode AuditType, gen(auditcode) //Create variable for audit types
165 recode auditcode (1=4) (2=1) (3=3) (4=7) (5=2) (6=5) (7=6), gen(auditttype)
166 drop auditcode
167 lab define type_lbl 1"1.Comprehensive audit" 2"2.Issue audit" 3
    "3.Credibility check" 4"4.CIT Audit" 5"5.PIT Audit" 6"6.VAT audit" 7
    "7.Desk audit"
168 label val auditttype type_lbl
```

Exercise 3 – Solutions continued

d) Encode the AuditType variable into auditcode, recode auditcode into audittype with the specified mappings, assign descriptive labels to audittype, generate a frequency table of audittype, order the variables tpin, initial_period, final_period, inter_period_1, inter_period_2, audittype, and totalassessments_annual, and save the cleaned dataset as audit.dta in the Data folder.

```
169 tabulate audittype
```

RECODE of auditcode (AuditType)	Freq.	Percent	Cum.
1.Comprehensive audit	5,039	0.75	0.75
2.Issue audit	5,045	0.75	1.50
3.Credibility check	5,083	0.76	2.26
4.CIT Audit	217,168	32.34	34.60
5.PIT Audit	216,950	32.31	66.91
6.VAT audit	217,123	32.33	99.24
7.Desk audit	5,085	0.76	100.00
Total	671,493	100.00	



Exercise 3 – Solutions continued

- d) Encode the AuditType variable into auditcode, recode auditcode into auditttype with the specified mappings, assign descriptive labels to auditttype, generate a frequency table of auditttype, order the variables tpin, initial_period, final_period, inter_period_1, inter_period_2, auditttype, and totalassessments_annual, and save the cleaned dataset as audit.dta in the Data folder.

```
170 order tpin initial_period final_period inter_period_1 inter_period_2  
    ↳ auditttype totalassessments_annual  
171  
172 *save as STATA data  
173 save "$data\audit.dta", replace
```

Exercise 4 – Questions – Date formatting

In audit data, dates often appear in inconsistent formats, in this exercise we will show an example of these date problems and how to convert them to STATA formats. In this example, you are given “[date_problem](#)” data with one date variable with eight observations and you have to find the incorrect observation and after correcting them convert them to a Stata date format by doing the following:

- a) Import this data to STATA after placing it in your Excel folder.
- b) List the variable observations to have an initial look.
- c) Browse records where the date variable cannot be converted into a valid date format in any of the specified formats: "DMY", "MDY", "YMD", or "MYD".
- d) Correct the records that could not be converted following the example of code line 202.
- e) Run the code from lines 211 to 234 to convert all the date variable records to STATA format after the correction.

Exercise 4 – Solutions

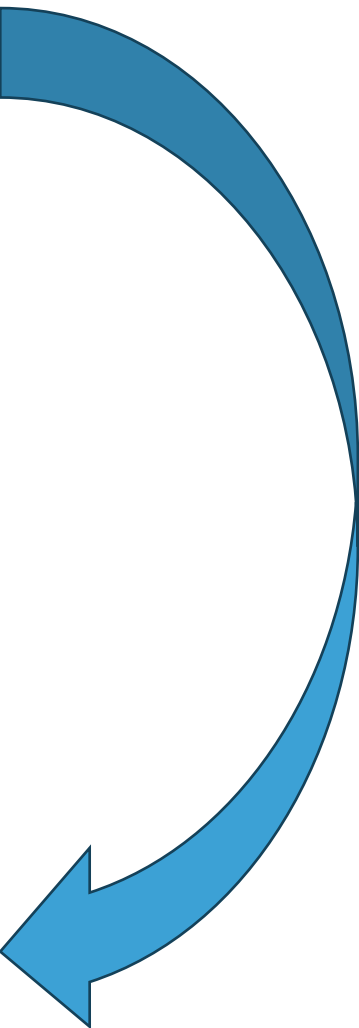
a) Import this data to STATA after placing it in your Excel folder.

```
188  
189 import delimited "$xlsout\date_problem.csv", clear varnames(1)  
190
```

b) List the variable observations to have an initial look.

```
192  
193 list assessment_date  
194
```

	assessment~e
1.	1 - Nov
2.	05/012/2018
3.	16/21207
4.	3.1.12018
5.	Feb-21
6.	01/05/2019
7.	01/06/2018
8.	01/06/2017

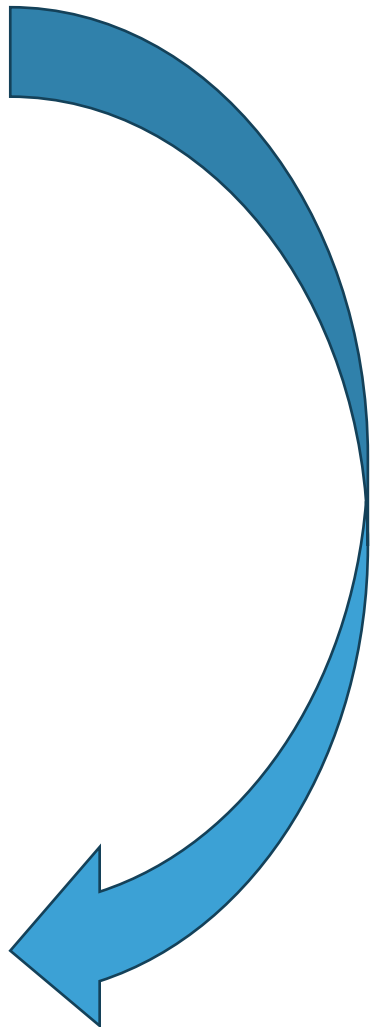


Exercise 4 – Solutions continued

c) Browse records where the date variable cannot be converted into a valid date format in any of the specified formats: "DMY", "MDY", "YMD", or "MYD".

```
197 browse if missing(date(assessment_date, "DMY")) & missing(date(
198 assessment_date, "MDY")) & missing(date(assessment_date, "YMD")) & missing(
199 date(assessment_date, "MYD"))
```

	assessment_date
1	1-Nov
2	05/012/2018
3	16/21207
4	3.1.12018
5	Feb-21



Exercise 4 – Solutions continued

d) Correct the records that could not be converted following the example of code line 202.

```
201  
202  replace assessment_date = "01/11/2019" if assessment_date == "1-Nov"  
203  replace assessment_date = "05/12/2018" if assessment_date == "05/012/2018"  
204  replace assessment_date = "16/12/2007" if assessment_date == "16/21207"  
205  replace assessment_date = "01/02/2021" if assessment_date == "Feb-21"  
206  replace assessment_date = "03/01/2018" if assessment_date == "3.1.12018"
```

Exercise 4 – Solutions continued

- e) Run the code from lines 211 to 234 to convert all the date variable records to STATA format after the correction.

```
210
211 local date_vars assessment_date
212
213 foreach var of local date_vars {
214     * Generate the corrected date variable
215     generate `var'_new = date(`var', "DMY")
216     format `var'_new %td
217     replace `var'_new = date(`var', "DM20Y") if missing(`var'_new)
218     replace `var'_new = date(`var', "MDY") if missing(`var'_new)
219     replace `var'_new = date(`var', "Y") if missing(`var'_new)
220
221     * Handle remaining cases where date conversion failed
222     generate `var'_numeric = `var' if missing(`var'_new)
223     destring `var'_numeric, replace force
224     replace `var'_new = `var'_numeric if missing(`var'_new)
225
226     * Clean up temporary variables
227     drop `var'_numeric
228 }
229
230 * Drop old date variables
231 drop assessment_date
232
233 * Rename new date variables as old ones
234 rename assessment_date_new assessment_date
235
```

More information available online

wider.unu.edu

Follow us on LinkedIn, Facebook, and Twitter **@UNUWIDER**