Searching for a Portuguese mortgage origination dataset using ED database

Ana Isabel Sá [*]
Faculdade de Economia Universidade do Porto

July 27, 2020

This document is an online appendix of the working paper "*Do low interest rates spur risk-taking in mortgage underwriting? An empirical study*". It explains how I have created a mortgage origination dataset of Portuguese loans using the information collected by the European Datawarehouse (ED).

[*]E-mail address: anaisabelsa@fep.up.pt (Ana Isabel Sá). Address: R Dr Roberto Frias, 4200–464 Porto, Portugal.

Contents

1 Compile European Datawarehouse files

European DataWarehouse (ED) is a centralized securitization repository for asset-backed securities (ABS) that stores loan-level data. Among the different portfolios, it includes residential mortgage-backed securities (RMBS).

For each RMBS, ED provides frequent updates on the loan performance. With the objective of creating a comprehensive origination dataset, I start by downloading all the reports on Portuguese RMBS and compile it into a single file. This new file has repeated information on each loan and and needs to be worked.

1.1 Keep only the relevant variables for the study

To reduce the size of the file, I have renamed and kept the relevant variables.

Stata Code:

```
//Rename Variables
rename poolcutoffdate CutOffDate
rename assetcountry Country
rename dataowner DataOwner
rename active Active
rename ar3 LoanId
rename ar5 OriginatorName
rename ar7 BorrowerId
rename ar8 PropertyId
rename ar21 BorrowerEmpStatus
rename ar26 PrimaryIncome
rename ar27 PrimaryIncomeVerification
rename ar28 SecondaryIncome
rename ar55 OriginationDate
rename ar56 MaturityDate
rename ar59 Purpose
rename ar61 OriginalTerm
rename ar65 Currency
rename ar66 OriginalAmount
rename ar67 CurrentAmount
rename ar69 RepaymentMethod
rename ar70 PaymentFrequency
rename ar71 PaymentDue
rename ar72 PaymentType
rename ar107 InterestRateType
rename ar108 CurrentIndex
rename ar109 CurrentInterestRate
rename ar110 CurrentSpread
rename ar113 SpreadFirstRevision
rename ar114 FirstRevisionDate
rename ar128 Region
rename ar129 PostalCode
rename ar131 PropertyType
rename ar135 OriginalLTV
rename ar136 OriginalHouseValue
rename ar138 OriginalValuationDate
rename ar143 CurrentHouseValue
rename ar166 LoanStatus

//Keep only the relevant variables
keep Active BorrowerEmpStatus BorrowerId Country Currency CurrentAmount ///
CurrentHouseValue CurrentIndex CurrentInterestRate CurrentSpread CutOffDate ///
DataOwner FirstRevisionDate InterestRateType LoanId LoanStatus MaturityDate ///
OriginalAmount OriginalHouseValue OriginalLTV OriginalTerm OriginationDate ///
OriginalValuationDate OriginatorName PaymentDue PaymentFrequency PaymentType ///
PostalCode PrimaryIncome PrimaryIncomeVerification PropertyId PropertyType ///
Purpose Region RepaymentMethod SecondaryIncome SpreadFirstRevision
```

Observations: 14,327,464

Deliverable: Part1_short.dta

2 Data correction - Main variables

I have inspected a set of critical points on main variables. I have found some inconsistencies that challenge the data integrity. The following is a summary of the findings for the variables *Originator Name*, *LoanId*, *Borrowerid*, and *Origination Date*.

2.1 Originator Name

Critical points:

- There are 31,887 observations with missing data;

- The name for each originator is not unique. For example, "Banco Santander" has 9 different entries.

Corrective actions:

- Drop observations where OriginatorName equals "ND,5";

- Create unique Originator identifier as IdBank.

Stata Code:

```
drop if OriginatorName=="ND,5"
gen temp1=substr(OriginatorName,1,10)
gen IdBank="BANCO SANTANDER" if temp1=="BANCO SANT"
replace IdBank="BANCO SANTANDER" if temp1=="Banco - Sa"
replace IdBank="BANCO SANTANDER" if temp1=="Banco Sant"
replace IdBank="BES" if temp1=="Banco Espi"
replace IdBank="BCP" if temp1=="Banco Come"
replace IdBank="BII" if temp1=="Banco de I"
replace IdBank="BARCLAYS" if temp1=="Barclays B"
replace IdBank="MONTEPIO GERAL" if temp1=="Caixa Econ"
replace IdBank="DEUTSCHE BANK" if temp1=="Deutsche B"
replace IdBank="NOVO BANCO" if temp1=="Novo Banco"
replace IdBank="UCI" if temp1=="UCI SA EFC"
replace IdBank="BANIF" if temp1=="Banif"
replace IdBank=temp1 if missing(IdBank)
```

```
drop temp1
```

2.2 LoanId

Critical points:

- The same loan has different LoanIds due to bank acquisitions. For example, LoanId equals “122474” as reported in February, 2013 by Banif and LoanId equals “122474,000800832000000” as reported in February, 2017 by Banco Santander.

Corrective actions:

- Create new Loan identifier as IdLoan that considers the following rule: If LoanId as a *comma*, then consider only the sequence up to the *comma*. If LoanId as no *comma*, then keep LoanId.

Stata Code:

```
//Correct LoanId by creating IdLoan
gen splitat=strpos(LoanId," ,")
replace splitat=strpos(LoanId,",") if splitat==0
replace splitat=strlen(LoanId)+1 if splitat==0
list LoanId if splitat==0
gen str1 IdLoan=""
replace IdLoan=substr(LoanId,1,splitat-1)
```

2.3 BorrowerId

Critical points:

- There are 571,299 observations with missing data;

- The same loan has different BorrowerIds due to errors in the 2012 Barclays report. For example, IdLoan “90002018510” has a BorrowerId “495666737,700185” as reported in December, 2012 and “495666737,700185668” as reported in all other dates. There are

16,215 observations with this type of error.

- The same loan has different BorrowerIds due to errors in the 2012 BES report. For example, IdLoan "AAAMOBZLUB" has a BorrowerId "HOMOOQOO" as reported in December, 2012 and "OOQOOMOH" as reported in all other dates. There are 57,223 observations with this type of error.

Corrective actions:

- Drop observations where BorrowerId equals "ND,5", "0", "#N/A", or "#REF!";

- In an origination dataset, there is only one record for each LoanId. Therefore, I do not need to follow Borrowers and will not implement any corrective actions on BorrowerId. In future steps, I will keep the first reported BorrowerId as the correct one;

- To uniform the code, I create a new variable IdBorrower.

Stata Code:

```
//Correct BorrowerId by creating IdBorrower
drop if BorrowerId=="ND,5"
drop if BorrowerId=="0"
drop if BorrowerId=="#N/A"
drop if BorrowerId=="#REF!"
sort BorrowerId
egen IdBorrower=group(BorrowerId)
```

2.4 OriginationDate

Critical points:

- There are 2,041,384 observations with missing data.

Corrective actions:

- Drop observations where OriginationDate equals ".".

Stata Code:

```
        drop if OriginationDate==.
```

Observations: 11,714,781

Deliverable: Part2.dta

3 Reduce to origination data

To create an origination dataset, I will keep only the first record for each IdLoan and inspect the data.

3.1 CutOffDate

First actions:

- For each IdLoan determine the first record as the minimum CutOffDate;

- Keep the observations where the first record is the same as the CutOffDate.

Stata Code:

```
//Keep only the first register for each IdLoan
bysort IdLoan: egen FirstCutOffDate=min(CutOffDate)
format FirstCutOffDate %td
keep if FirstCutOffDate==CutOffDate
drop FirstCutOffDate
```

Observations: 682,337

3.2 Data correction - Origination

Critical points:

- Repeated records for the same IdLoan on the first record date. There are 20,695 observations in this situation with three types of errors:

- Error Type I - 78 records on 2012 and 2013 BPI's report, for which the same loan has more than one record with the same characteristics, and with negative OriginalAmount. For example, IdLoan "001847943165004" has two records with OriginalAmount of "-50,000".

- Error Type II - 10 records on August, 2013 Montepio Geral's report, for which the same loan has different characteristics at the same CutOffDate. For example, IdLoan "0159466" has two records with OriginalAmount of "144,500" and "25,000", OriginalHouseValue of "174,000" and "140,000", among other differences.

- Error Type III - 20,607 records on July, 2018 UCI's report, for which the same loan has more than one record with the same characteristics. For example, IdLoan "1003716" has three records.

- The same loan with different LoanId on the first record date. All loan characteristics are the same, except for the IdLoan and (occasionally) the Purpose. There are 1,238 observations in this situation. For example, IdLoan "036500960016175" and "036500960016183" as reported in December, 2012 by Banco Santander.

Corrective actions:

- For Errors Type I and Type II, I cannot rely on the information, thus drop observations. For Type III, keep only the first observation;

- For same loan with different IdLoan, keep only the first observation.

Stata Code:

```
//Repeated IdLoan in the same CutOffDate
gen One=1
sort IdLoan Purpose
bysort IdLoan: egen RepsLoanId_before=sum(One)
gen ErrorTypeI=1 if(RepsLoanId_before>1 & OriginalAmount<0)
replace ErrorTypeI=0 if missing(ErrorTypeI)
gen ErrorTypeII=1 if(RepsLoanId_before>1 & RepsLoanId_before<3 & OriginalAmount>0)
replace ErrorTypeII=0 if missing(ErrorTypeII)
drop if ErrorTypeI==1
drop if ErrorTypeII==1
drop ErrorTypeI ErrorTypeII
bysort IdLoan: keep if _n==1
bysort IdLoan: egen Reps_after=sum(One)
tab RepsLoanId_before Reps_after
drop RepsLoanId_before Reps_after

//Different IdLoan but the same loan
egen SimulateIdLoan=group(IdBorrower OriginationDate OriginalAmount IdBank PropertyId)
sort SimulateIdLoan CutOffDate Purpose
bysort SimulateIdLoan: egen Reps_before=sum(One)
bysort SimulateIdLoan: keep if _n==1
bysort SimulateIdLoan: egen Reps_after=sum(One)
tab Reps_before Reps_after
drop Reps_before Reps_after SimulateIdLoan
```

Observations: 667,891

Deliverable: Part3.dta

4 Identify exposures

Some mortgages are divided in more than one loan. However, due to different criteria followed by different banks and due to different criteria followed by the same bank in different years, it is not possible to identify exposures using a unique rule. As a result, I start by defining five types of exposures.

Critical points:

- Exposures with unique record:

- Type I - Combination IdBorrower and OriginationDate is unique. There are 317,426 observations;

- Exposures with more than one record:

- Type II - Same IdBorrower, OriginationDate, PropertyId, OriginalHouseValue, and CutOffDate. There are 219,930 observations;
- Type III - Same IdBorrower, OriginationDate, PropertyId, and CutOffDate but different OriginalHouseValue. There are 79,737 observations;
- Type IV - Same IdBorrower, OriginationDate, OriginalHouseValue, and CutOffDate but different PropertyId. There are 12,396 observations;

- Potentially repeated exposures:

- Type V - Same IdBorrower and OriginationDate in different CutOffDate. There are 38,402 observations;

Stata Code:

```
//Type I - Exposures with only one record
egen Exp1=group(IdBorrower OriginationDate)
sort Exp1
bysort Exp1: egen RepsExp1=sum(One)
gen TypeI=1 if RepsExp1==1
replace TypeI=0 if missing(TypeI)

//Type II - Exposures with more than one record and same IdBorrower, OriginationDate,
    PropertyId, OriginalHouseValue, and CutOffDate
replace OriginalHouseValue=1 if missing(OriginalHouseValue)
egen Exp2=group(IdBorrower OriginationDate PropertyId OriginalHouseValue CutOffDate)
sort Exp2 CutOffDate Purpose
bysort Exp2: egen RepsExp2=sum(One)
gen TypeII=1 if RepsExp2>1
replace TypeII=0 if missing(TypeII)

//Type III - Exposures with more than one record and same IdBorrower, OriginationDate,
    PropertyId, and CutOffDate but different OriginalHouseValue
egen Exp3=group(IdBorrower OriginationDate PropertyId CutOffDate)
sort Exp3 CutOffDate Purpose
bysort Exp3: egen RepsExp3=sum(One)
gen TypeIII=1 if (RepsExp3>1&TypeII==0)
replace TypeIII=0 if missing(TypeIII)

//Type IV - Exposures with more than one record and same IdBorrower, OriginationDate,
    OriginalHouseValue, and CutOffDate but different PropertyId
egen Exp4=group(IdBorrower OriginationDate OriginalHouseValue CutOffDate)
sort Exp4 CutOffDate Purpose
bysort Exp4: egen RepsExp4=sum(One)
gen TypeIV=1 if (RepsExp4>1&TypeII==0&TypeIII==0)
replace TypeIV=0 if missing(TypeIV)

//Type V - Exposures potentially repeated
gen TypeV=1 if (RepsExp1>1&TypeII==0&TypeIII==0&TypeIV==0)
replace TypeV=0 if missing(TypeV)
```

Corrective actions:

- Create a exposure identifier according to the types described. All further corrective actions shall consider the exposure as a ”single record”;

- Keep only the first CutOffDate for the Type V exposures to eliminate exposures repeated duo to changes in the LoanId or renegotiation;

- Reclassify exposures and create a new Type VI for different exposures granted on the same day to the same Borrower.

Stata Code:

```
//Create IdExposure - Part I
gen Exp1_unic = "A" + string(Exp1, "%06.0f")
gen temp=Exp1_unic if TypeI==1
gen Exp2_unic = "B" + string(Exp2, "%06.0f")
replace temp=Exp2_unic if TypeII==1
gen Exp3_unic = "C" + string(Exp3, "%06.0f")
replace temp=Exp3_unic if TypeIII==1
replace temp=Exp1_unic if TypeIV==1
replace temp=Exp1_unic if TypeV==1
rename temp IdExposure

//Keep only the first CutOffDate for each Type V exposure
gen IdExposureTV=IdExposure*TypeV
sort IdExposureTV CutOffDate
by IdExposureTV: egen FirstCutOffDate=min(CutOffDate)
format FirstCutOffDate %td
keep if (missing(IdExposureTV)|FirstCutOffDate==CutOffDate)
drop FirstCutOffDate

//Reclassify exposures
//Drop intermediate variables
drop Exp* RepsExp* IdExposureTV IdExposure Type*

//Type I - Exposures with only one record
egen Exp1=group(IdBorrower OriginationDate)
sort Exp1
bysort Exp1: egen RepsExp1=sum(One)
gen TypeI=1 if RepsExp1==1
```

```
replace TypeI=0 if missing(TypeI)

//Type II - Exposures with more than one record and same IdBorrower, OriginationDate,
    PropertyId, OriginalHouseValue, and CutOffDate
replace OriginalHouseValue=1 if missing(OriginalHouseValue)
egen Exp2=group(IdBorrower OriginationDate PropertyId OriginalHouseValue ///
CutOffDate)
sort Exp2 CutOffDate Purpose
bysort Exp2: egen RepsExp2=sum(One)
gen TypeII=1 if RepsExp2>1
replace TypeII=0 if missing(TypeII)

//Type III - Exposures with more than one record and same IdBorrower, OriginationDate,
    PropertyId, and CutOffDate but different OriginalHouseValue
egen Exp3=group(IdBorrower OriginationDate PropertyId CutOffDate)
sort Exp3 CutOffDate Purpose
bysort Exp3: egen RepsExp3=sum(One)
gen TypeIII=1 if (RepsExp3>1&TypeII==0)
replace TypeIII=0 if missing(TypeIII)

//Type IV - Exposures with more than one record and same IdBorrower, OriginationDate,
    OriginalHouseValue, and CutOffDate but different PropertyId
egen Exp4=group(IdBorrower OriginationDate OriginalHouseValue CutOffDate)
sort Exp4 CutOffDate Purpose
bysort Exp4: egen RepsExp4=sum(One)
gen TypeIV=1 if (RepsExp4>1&TypeII==0&TypeIII==0)
replace TypeIV=0 if missing(TypeIV)

//Type VI - Borrowers with more than one exposure at the same date
gen TypeVI=1 if (RepsExp1>1&TypeII==0&TypeIII==0&TypeIV==0)
replace TypeVI=0 if missing(TypeVI)
egen Exp6=group(IdBorrower OriginationDate CutOffDate) if (TypeVI==1)
gen Exp7=Exp6+700000
drop Exp6
rename Exp7 Exp6

//Recode IdExposure
//Recode IdExposure - PartII
gen Exp1_unic = "A" + string(Exp1, "%06.0f")
gen temp=Exp1_unic if TypeI==1
gen Exp2_unic = "B" + string(Exp2, "%06.0f")
replace temp=Exp2_unic if TypeII==1
gen Exp3_unic = "C" + string(Exp3, "%06.0f")
replace temp=Exp3_unic if TypeIII==1
replace temp=Exp1_unic if TypeIV==1
replace temp=Exp1_unic if TypeV==1
rename temp IdExposure

//Recode single exposures as Type I
bysort IdExposure: egen Reps_single=sum(One)
replace TypeI=1 if(TypeII==1&Reps_single==1)
replace TypeI=1 if(TypeIII==1&Reps_single==1)
replace TypeI=1 if(TypeIV==1&Reps_single==1)
replace TypeII=0 if(TypeII==1&Reps_single==1)
replace TypeIII=0 if(TypeIII==1&Reps_single==1)
replace TypeIV=0 if(TypeIV==1&Reps_single==1)
drop IdExposure *_unic

//Recode IdExposure - PartIII
gen Exp1_unic = "A" + string(Exp1, "%06.0f")
gen temp=Exp1_unic if TypeI==1
gen Exp2_unic = "B" + string(Exp2, "%06.0f")
replace temp=Exp2_unic if TypeII==1
gen Exp3_unic = "C" + string(Exp3, "%06.0f")
replace temp=Exp3_unic if TypeIII==1
replace temp=Exp1_unic if TypeIV==1
replace temp=Exp1_unic if TypeV==1
egen IdExposure=group(temp)
sum temp IdExposure
drop temp
```

Observations: 649,432

Exposures: 491,740

Deliverable: Part4.dta

5 Data correction - Other variables

With the exposures identified, I proceed by inspecting data and implementing corrective measures at the exposure level. The following is a summary of the findings for the variables *OriginalAmount*, *OriginalHouseValue*, *InterestRateType*, *CurrentIndex*, *CurrentSpread*, *OriginalValuationDate*, *Purpose*, and *PropertyType*.

5.1 OriginalAmount

Critical points:

- Some exposures have negative OriginalAmount. For example, IdLoan "009838961165003" has an OriginalAmount of "36,408.68".

Corrective actions:

- Delete exposures with negative or missing OriginalAmount.

Stata Code:

```
// Drop Exposures with, at least, one missing or negative OriginalAmount
gen temp=1 if OriginalAmount==.
replace temp=1 if OriginalAmount==0
replace temp=1 if OriginalAmount<0
replace temp=0 if OriginalAmount>0
bysort IdExposure: egen ExposureAmountDelete=sum(temp)
tab ExposureAmountDelete
drop if ExposureAmountDelete>0
drop temp ExposureAmountDelete
egen temp2=group(IdExposure)
sum temp2
drop temp2
```

Observations: 558,505

Exposures: 426,536

5.2 OriginalHouseValue

Critical points:

- Some exposures have missing OriginalHouseValue or values less than to 5,000 euros. For example, IdLoan "70567655653" has a missing OriginalHouseValue.

Corrective actions:

- Delete exposures with OriginalHouseValue missing or values inferior to 5,000 euros.

Stata Code:

```
//Drop Exposures with, at least, one missing OriginalHouseValue
gen temp=1 if OriginalHouseValue==.
replace temp=1 if OriginalHouseValue<5000 //Missing values substituted by 1
replace temp=0 if OriginalHouseValue==5000
replace temp=0 if OriginalHouseValue>5000
bysort IdExposure: egen ExposureHouseDelete=sum(temp)
tab ExposureHouseDelete
drop if ExposureHouseDelete>0
drop temp ExposureHouseDelete
egen temp2=group(IdExposure)
sum temp2
drop temp2
```

Observations: 546,431

Exposures: 417,159

5.3 InterestRateType

Critical points:

- Some exposures have InterestRateType equal to “others”. For example, IdLoan “6220844165001”.

- Some exposures are not adjustable-rate mortgages (ARM).

Corrective actions:

- Delete exposures with InterestRateType different than ARM.

Stata Code:

```
//Drop Exposures with, at least, one missing InterestRateType or not ARM
gen temp=0 if InterestRateType==2
replace temp=0 if InterestRateType==1
replace temp=1 if missing(temp)
bysort IdExposure: egen ExposureTypeDelete=sum(temp)
tab ExposureTypeDelete
drop if ExposureTypeDelete>0
drop temp ExposureTypeDelete
egen temp2=group(IdExposure)
sum temp2
drop temp2
```

Observations: 530,992

Exposures: 404,048

5.4 CurrentIndex

Critical points:

- Some exposures have missing CurrentIndex. For example, IdLoan “000300857010096”.

- Some exposures are not indexed to Euribor, and that is not the usual in Portugal.

For example, IdLoan “192809” has a CurrentIndex.

Corrective actions:

- Delete exposures with CurrentIndex different than Euribor.

Stata Code:

```
//Drop Exposures with CurrentIndex different from EURIBOR1M, EURIBOR3M, EURIBOR6M or
    EURIBOR12M
gen temp=0 if CurrentIndex==2
replace temp=0 if CurrentIndex==4
replace temp=0 if CurrentIndex==6
replace temp=0 if CurrentIndex==8
replace temp=1 if missing(temp)
bysort IdExposure: egen ExposureIndexDelete=sum(temp)
tab ExposureIndexDelete
drop if ExposureIndexDelete>0
drop temp ExposureIndexDelete
egen temp2=group(IdExposure)
sum temp2
drop temp2
```

Observations: 524,311

Exposures: 399,291

5.5 CurrentSpread

Critical points:

- Some exposures have CurrentSpread of zero with no indication of SpreadRevision. This would mean that mortgages had a zero spread since origination. For example, IdLoan ”000303398996096” has a CurrentSpread zero and no SpreadRevision.

- Some exposures have CurrentSpread missing or with unreasonable values. For example, IdLoan “347263” has a CurrentSpread of 11%.

Corrective actions:

- Delete exposures with CurrentSpread zero and no indication of SpreadRevision.

- Delete exposures with CurrentSpread missing or greater than 10%

Stata Code:

```
//Drop Exposures with CurrentSpread equal to zero and no indication of Spread Revision
gen temp=0 if missing(SpreadFirstRevision)
replace temp=0 if SpreadFirstRevision==CurrentSpread
replace temp=0 if SpreadFirstRevision==0
replace temp=1 if missing(temp)
tab temp
gen temp2=1 if(temp==0 & CurrentSpread==0)
bysort IdExposure: egen ExposureSpreadDelete=sum(temp2)
tab ExposureSpreadDelete
drop if ExposureSpreadDelete>0
drop temp temp2 ExposureSpreadDelete
egen temp3=group(IdExposure)
sum temp3
drop temp3

//Drop Exposures with CurrentSpread missing or above 10
gen temp=1 if missing(CurrentSpread)
replace temp=1 if CurrentSpread>10
replace temp=0 if missing(temp)
bysort IdExposure: egen ExposureSpreadMisDelete=sum(temp)
tab ExposureSpreadMisDelete
drop if ExposureSpreadMisDelete>0
drop temp ExposureSpreadMisDelete
egen temp2=group(IdExposure)
sum temp2
drop temp2
```

Observations: 523,790

Exposures: 398,912

5.6 OriginalValuationDate

Critical points:

- Some exposures have an OriginalValuationDate a longtime before the Origination-Date as it might sign an error. For example, IdLoan “285324” was originated on July, 2009 and the house was valuated on March, 1994.

Corrective actions:

- Drop exposures whose property’s original valuation date and loan origination date distance more than one year.

Stata Code:

```
//Drop Exposures with OriginalValuationDate distancing more than one year from the
    OriginationDate
gen temp=OriginationDate-OriginalValuationDate
gen temp2=1 if temp>365
replace temp2=0 if missing(temp2)
bysort IdExposure: egen ExposureOrigDelete=sum(temp2)
tab ExposureOrigDelete
drop if ExposureOrigDelete>0
drop temp temp2 ExposureOrigDelete
egen temp3=group(IdExposure)
sum temp3
drop temp3
```

Observations: 482,810

Exposures: 364,144

5.7 Purpose

Critical points:

- Some exposures have a Purpose different from 1-Purchase and 7-Others. For example, IdLoan “000307229023096” has purpose 3-Renovation.

Corrective actions:

- Many exposures have missing values. Keep only the exposures with purpose 1, 7 or missing.

Stata Code:

```
//Keep only the exposures with purpose 1 or 7
gen temp=1 if Purpose==1
replace temp=1 if Purpose==7
replace temp=1 if missing(Purpose)
replace temp=0 if missing(temp)
bysort IdExposure: egen ExposurePurposeDelete=sum(temp)
tab ExposurePurposeDelete
drop if ExposurePurposeDelete==0
drop temp ExposurePurposeDelete
egen temp2=group(IdExposure)
sum temp2
drop temp2
```

Observations: 447,422

Exposures: 329,014

5.8 PropertyType

Critical points:

- Some exposures have a Property Type that is not residential. For example, IdLoan “PT 0035034400002609085” has a ”Partially commercial use” property.

Corrective actions:

- Keep only exposures with PropertyType residential.

Stata Code:

```
//Keep only residential properties
gen temp=1 if PropertyType==1
replace temp=1 if PropertyType==2
replace temp=1 if PropertyType==3
replace temp=1 if PropertyType==4
replace temp=0 if missing(temp)
bysort IdExposure: egen ExposurePropertyDelete=sum(temp)
tab ExposurePropertyDelete
drop if ExposurePropertyDelete==0
drop temp ExposurePropertyDelete
egen temp2=group(IdExposure)
sum temp2
drop temp2
```

Observations: 421,637

Exposures: 310,506

6 New variables

In addition to the corrections implemented in Section 5, I have created new variables that are relevant for an origination dataset.

6.1 OriginalSpread

Critical points:

- The ED database does not provide OriginalSpread. However, it is possible to compute it from CurrentSpread and SpreadRevision information.

Corrective actions:

- If there is no SpreadRevision, define OriginalSpread as the CurrentSpread. Otherwise, consider the FirstSpreadRevision.

Stata Code:

```
//Create OriginalSpread and drop nonsense values
gen dSpreadRevision=0 if missing(SpreadFirstRevision)
replace dSpreadRevision=0 if SpreadFirstRevision==CurrentSpread
replace dSpreadRevision=0 if SpreadFirstRevision==0
replace dSpreadRevision=1 if missing(dSpreadRevision)
gen OriginalSpread=SpreadFirstRevision if(dSpreadRevision==1)
replace OriginalSpread=CurrentSpread if missing(OriginalSpread)
gen temp=1 if missing(OriginalSpread)
replace temp=1 if OriginalSpread>10
replace temp=0 if missing(temp)
bysort IdExposure: egen Exposure0SpreadDelete=sum(temp)
tab Exposure0SpreadDelete
drop if Exposure0SpreadDelete>0
drop temp Exposure0SpreadDelete
```

6.2 Term

Critical points:

- The OriginalTerm included in ED database is the difference between the MaturityDate and the CutOffDate.

Corrective actions:

- Define Term as the difference between the MaturityDate and the OriginationDate and drop nonsense values.

Stata Code:

```
//Create Term variable (the one in the dataset is the difference between the MaturityDate
     and the CutOffDate) and drop nonsense values
gen OriginationMonth=mofd(OriginationDate)
format OriginationMonth %tm
```

```
gen MaturityMonth=mofd(MaturityDate)
format MaturityMonth %tm
gen Term=MaturityMonth-OriginationMonth
gen temp=1 if Term<0
replace temp=0 if missing(temp)
bysort IdExposure: egen ExposureTermDelete=sum(temp)
tab ExposureTermDelete
drop if ExposureTermDelete>0
drop temp ExposureTermDelete
```

6.3 Income

Critical points:

- The ED database provides information on the PrimaryIncome and SecondaryIncome. However, there is no information on TotalIncome, and there is no guarantee that all banks report SecondaryIncome.

Corrective actions:

- Correct missing values on PIncome and SIncome. Create TIncome as the sum of PIncome and SIncome.

Stata Code:

```
//Create Income variables and correct
gen PIncome=PrimaryIncome
replace PIncome=0 if missing(PIncome)
gen SIncome=SecondaryIncome
replace SIncome=0 if missing(SIncome)
gen TIncome=PIncome+SIncome
```

6.4 PropertyId

Critical points:

- There are different references for missing PropertyId.

Corrective actions:

- Correct missing values on PropertyId and create new variable as IdProperty.

Stata Code:

```
//Create new variable for IdProperty
replace PropertyId="" if PropertyId=="ND,5"
replace PropertyId="" if PropertyId=="ND,2"
replace PropertyId="" if PropertyId=="999999999"
sort PropertyId
egen IdProperty=group(PropertyId IdExposure)
```

6.5 PostalCode

Critical points:

- Portuguese PostalCode has four digits plus three, but some banks only report the first four digits.

Corrective actions:

- Readjust PostalCode to include only the first four digits and uniform missing values.

Stata Code:

```
//Create new PostalCodeNum
gen temp=substr(PostalCode,1,4)
destring temp, generate(PostalCodeNum) force
replace PostalCodeNum=. if PostalCodeNum==0
replace PostalCodeNum=. if PostalCodeNum==9999
drop PostalCode
rename PostalCodeNum PostalCode
drop temp
```

Observations: 421,528

Exposures: 310,405

Deliverable: Part6.dta

7 Merge exposures into a single record

After identifying exposures, it is necessary to merge the information into a single record, as if it is a single loan.

7.1 Keep only the relevant variables for the study

Before starting the merging, I will rename, reduce and order the dataset.

Stata Code:

```
//Keep variables relevant for Origination dataset
rename OriginalAmount Amount
rename OriginalHouseValue HouseValue
rename CurrentIndex Index
rename OriginalSpread Spread

keep IdBank IdBorrower IdExposure IdLoan OriginationDate MaturityDate Term ///
Amount HouseValue IdProperty PostalCode Index Spread PIncome SIncome TIncome ///
CutOffDate One TypeI TypeII TypeIII TypeIV TypeV TypeVI

order IdBank IdBorrower IdExposure IdLoan OriginationDate MaturityDate Term ///
Amount HouseValue IdProperty PostalCode Index Spread PIncome SIncome TIncome ///
CutOffDate One TypeI TypeII TypeIII TypeIV TypeV TypeVI

gsort +IdExposure -Amount
save "‘path’\Part7All.dta"
```

7.2 Split database according to the type of exposure

As set in Section 4, different banks have different criterias for reporting the exposures, and even within the same bank there are differences in the reporting according to the reporting year. The criteria to merge exposures depends on the type defined in Section 4. For each exposure type, I will create a subset.

Stata Code:

```
//Separate IdExposures according to Type
//Type I and VI -No need to merge
keep if (TypeI==1|TypeVI==1)
save "`path'\Part7TypeI_VI.dta"

//Type II - Same IdBorrower, OriginationDate, PropertyId, OriginalHouseValue, and
    CutOffDate
clear all
use "`path'\Part7All.dta"
keep if TypeII==1
save "`path'\Part7TypeII.dta"

//Type III - Same IdBorrower, OriginationDate, PropertyId, and CutOffDate
clear all
use "`path'\Part7All.dta"
keep if TypeIII==1
save "`path'\Part7TypeIII.dta"

//Type IV - Same IdBorrower, OriginationDate, OriginalHouseValue, and CutOffDate
clear all
use "`path'\Part7All.dta"
keep if TypeIV==1
save "`path'\Part7TypeIV.dta"
```

7.3 Merge Type I and Type VI exposures

Type I and Type VI exposures are single record exposures, thus the only treatment that they need is to uniform the dataset to the final format.

Stata Code:

```
//Format Type I and VI Exposures
clear all
use "`path'\Part7TypeI_VI.dta"
gen IdBank_m=IdBank
gen IdBorrower_m=IdBorrower
gen IdExposure_m=IdExposure
gen OriginationDate_m=OriginationDate
format OriginationDate_m %td
gen MaturityDate_m=MaturityDate
format MaturityDate_m %td
gen Term_m=Term
gen Amount_m=Amount
gen HouseValue_m=HouseValue
gen IdProperty_m=IdProperty
gen PostalCode_m=PostalCode
gen Index_m=Index
gen Spread_m=Spread
gen PIncome_m=PIncome
gen SIncome_m=SIncome
gen TIncome_m=TIncome
gen CutOffDate_m=CutOffDate
format CutOffDate_m %td
keep *_m Type*
save "`path'\Part7TypeIandVI_merged.dta"
```

7.4 Merge Type II exposures

Type II exposures have the same IdBorrower, OriginationDate, PropertyId, Original-HouseValue, and CutOffDate. To merge information I (1) check for possible repetitions, (2) merge variables that are equal within the exposure, (3) merge *MaturityDate*, *Term*, *Spread* using weighted average, (4) merge *Amount* using sum, and (5) merge *PostalCode*, *Index*, *PIncome*, *SIncome*, and *TIncome* by keeping the first record.

Stata Code:

```
//Merge Type II Exposures
clear all
use "`path'\Part7TypeII.dta"

//Number of lines by exposure and weights
bysort IdExposure: egen Reps_Exp=sum(One)
bysort IdExposure: egen temp=sum(Amount)
gen Weight=Amount/temp
drop temp

//Check for possible repetitions. If more than one loan has the same amount, then it is
    possibly repeated
bysort IdExposure: egen Amount_Exp=sum(Amount)
gen Amount_ExpAvg=Amount_Exp/Reps_Exp
gen Amount_ExpReps=Amount_ExpAvg-Amount_Exp
gen DRepeated_Amount=1 if Amount_ExpReps==0
replace DRepeated_Amount=0 if missing(DRepeated_Amount)
sum DRepeated_Amount // if any DRepeated_Amount=0, then check for repetitions. Otherwise
    continue

//Merge variables that are equal within the Exposure
gen IdBank_m=IdBank
gen IdBorrower_m=IdBorrower
gen IdExposure_m=IdExposure
gen OriginationDate_m=OriginationDate
format OriginationDate_m %td
gen HouseValue_m=HouseValue
gen IdProperty_m=IdProperty
gen CutOffDate_m=CutOffDate
format CutOffDate_m %td

//Merge MaturityDate - method: weighted average
// - Identify pattern
bysort IdExposure: egen temp=sum(MaturityDate)
gen MaturityDate_ExpAvg=temp/Reps_Exp
```

```
drop temp
gen MaturityDate_ExpReps=MaturityDate_ExpAvg-MaturityDate
gen DRepeated_MaturityDate=1 if MaturityDate_ExpReps==0
replace DRepeated_MaturityDate=0 if missing(DRepeated_MaturityDate)
sum DRepeated_MaturityDate

// - Create merge variable
gen MaturityDate_m=MaturityDate if DRepeated_MaturityDate==1
gen temp=Weight*MaturityDate
bysort IdExposure: egen temp2=sum(temp)
replace MaturityDate_m=temp2 if DRepeated_MaturityDate==0
format MaturityDate_m %td
drop temp temp2

//Merge Term - method: weighted average
// - Identify pattern
bysort IdExposure: egen temp=sum(Term)
gen Term_ExpAvg=temp/Reps_Exp
drop temp
gen Term_ExpReps=Term_ExpAvg-Term
gen DRepeated_Term=1 if Term_ExpReps==0
replace DRepeated_Term=0 if missing(DRepeated_Term)
sum DRepeated_Term

// - Create merge variable
gen Term_m=Term if DRepeated_Term==1
gen temp=Weight*Term
bysort IdExposure: egen temp2=sum(temp)
replace Term_m=temp2 if DRepeated_Term==0
drop temp temp2

//Merge Amount - method: sum
// - Create merge variable
bysort IdExposure: egen temp=sum(Amount)
gen Amount_m=temp
drop temp

//Merge PostalCode - method: keep the first
bysort IdExposure: gen temp=_n
gen temp2=PostalCode if temp==1
```

```
replace temp2=0 if missing(temp2)
bysort IdExposure: egen PostalCode_m=max(temp2)
drop temp temp2

//Merge Index - method: keep the first
// - Identify pattern
bysort IdExposure: egen temp=sum(Index)
gen Index_ExpAvg=temp/Reps_Exp
drop temp
gen Index_ExpReps=Index_ExpAvg-Index
gen DRepeated_Index=1 if Index_ExpReps==0
replace DRepeated_Index=0 if missing(DRepeated_Index)
sum DRepeated_Index // if any DRepeated_Index=0, then evaluate

//*Some observations have index Eur6M and Eur3M. I decided to consider only the index of
    the most representative loan*
bysort IdExposure: gen temp=_n
gen temp2=Index if temp==1
replace temp2=0 if missing(temp2)
bysort IdExposure: egen Index_m=max(temp2)
drop temp temp2

//Merge Spread - method: weighted average
// - Identify pattern
bysort IdExposure: egen temp=sum(Spread)
gen Spread_ExpAvg=temp/Reps_Exp
drop temp
gen Spread_ExpReps=Spread_ExpAvg-Spread
gen DRepeated_Spread=1 if Spread_ExpReps==0
replace DRepeated_Spread=0 if missing(DRepeated_Spread)
sum DRepeated_Spread

// - Create merge variable
gen Spread_m=Spread if DRepeated_Spread==1
gen temp=Weight*Spread
bysort IdExposure: egen temp2=sum(temp)
replace Spread_m=temp2 if DRepeated_Spread==0
drop temp temp2

//Merge PIncome, SIncome, and TIncome - method: keep the first
```

```
bysort IdExposure: gen temp=_n
gen temp2=PIncome if temp==1
replace temp2=0 if missing(temp2)
bysort IdExposure: egen PIncome_m=max(temp2)
gen temp3=SIncome if temp==1
replace temp3=0 if missing(temp3)
bysort IdExposure: egen SIncome_m=max(temp3)
gen temp4=TIncome if temp==1
replace temp4=0 if missing(temp4)
bysort IdExposure: egen TIncome_m=max(temp4)
drop temp temp2 temp3 temp4

//Merge Exposures
keep *_m D* Type*
bysort IdExposure_m: keep if _n==1
save "`path'\Part7TypeII_merged.dta"
```

7.5 Merge Type III exposures

Type III exposures have the same IdBorrower, OriginationDate, PropertyId, and Cut-OffDate. To merge information I (1) check for possible repetitions, (2) merge variables that are equal within the exposure, (3) merge *MaturityDate*, *Term*, *Spread* using weighted average, (4) merge *Amount* using sum, (5) *HouseValue* by keeping the highest value and (6) merge *PostalCode*, *Index*, *PIncome*, *SIncome*, and *TIncome* by keeping the first record.

Stata Code:

```
//Merge Type III Exposures
clear all
use "`path'\Part7TypeIII.dta"

//Number of lines by exposure and weights
bysort IdExposure: egen Reps_Exp=sum(One)
bysort IdExposure: egen temp=sum(Amount)
gen Weight=Amount/temp
drop temp
```

```
//Check for possible repetions. If more than one loan has the same amount, then it is
    possibly repeated
bysort IdExposure: egen Amount_Exp=sum(Amount)
gen Amount_ExpAvg=Amount_Exp/Reps_Exp
gen Amount_ExpReps=Amount_ExpAvg-Amount
gen DRepeated_Amount=1 if Amount_ExpReps==0
replace DRepeated_Amount=0 if missing(DRepeated_Amount)
sum DRepeated_Amount // if any DRepeated_Amount=0, then check for repetitions. Otherwise
    continue

//Merge variables that are equal within the Exposure
gen IdBank_m=IdBank
gen IdBorrower_m=IdBorrower
gen IdExposure_m=IdExposure
gen OriginationDate_m=OriginationDate
format OriginationDate_m %td
gen IdProperty_m=IdProperty
gen CutOffDate_m=CutOffDate
format CutOffDate_m %td

//Merge MaturityDate - method: weighted average
// - Identify pattern
bysort IdExposure: egen temp=sum(MaturityDate)
gen MaturityDate_ExpAvg=temp/Reps_Exp
drop temp
gen MaturityDate_ExpReps=MaturityDate_ExpAvg-MaturityDate
gen DRepeated_MaturityDate=1 if MaturityDate_ExpReps==0
replace DRepeated_MaturityDate=0 if missing(DRepeated_MaturityDate)
sum DRepeated_MaturityDate

// - Create merge variable
gen MaturityDate_m=MaturityDate if DRepeated_MaturityDate==1
gen temp=Weight*MaturityDate
bysort IdExposure: egen temp2=sum(temp)
replace MaturityDate_m=temp2 if DRepeated_MaturityDate==0
format MaturityDate_m %td
drop temp temp2

//Merge Term - method: weighted average
// - Identify pattern
```

```
bysort IdExposure: egen temp=sum(Term)
gen Term_ExpAvg=temp/Reps_Exp
drop temp
gen Term_ExpReps=Term_ExpAvg-Term
gen DRepeated_Term=1 if Term_ExpReps==0
replace DRepeated_Term=0 if missing(DRepeated_Term)
sum DRepeated_Term

// - Create merge variable
gen Term_m=Term if DRepeated_Term==1
gen temp=Weight*Term
bysort IdExposure: egen temp2=sum(temp)
replace Term_m=temp2 if DRepeated_Term==0
drop temp temp2

//Merge Amount - method: sum
// - Create merge variable
bysort IdExposure: egen temp=sum(Amount)
gen Amount_m=temp
drop temp

//Merge HouseValue - method: keep the highest
gsort +IdExposure -HouseValue
bysort IdExposure: gen temp=_n
gen temp2=HouseValue if temp==1
replace temp2=0 if missing(temp2)
bysort IdExposure: egen HouseValue_m=max(temp2)
drop temp temp2
gsort +IdExposure -Amount

//Merge PostalCode - method: keep the first
bysort IdExposure: gen temp=_n
gen temp2=PostalCode if temp==1
replace temp2=0 if missing(temp2)
bysort IdExposure: egen PostalCode_m=max(temp2)
drop temp temp2

//Merge Index - method: keep the first
// - Identify pattern
bysort IdExposure: egen temp=sum(Index)
```

```
gen Index_ExpAvg=temp/Reps_Exp
drop temp
gen Index_ExpReps=Index_ExpAvg-Index
gen DRepeated_Index=1 if Index_ExpReps==0
replace DRepeated_Index=0 if missing(DRepeated_Index)
sum DRepeated_Index // if any DRepeated_Index=0, then evaluate

//*Some observations have index Eur6M and Eur3M. I decided to consider only the index of
    the most representative loan*
bysort IdExposure: gen temp=_n
gen temp2=Index if temp==1
replace temp2=0 if missing(temp2)
bysort IdExposure: egen Index_m=max(temp2)
drop temp temp2

//Merge Spread - method: weighted average
// - Identify pattern
bysort IdExposure: egen temp=sum(Spread)
gen Spread_ExpAvg=temp/Reps_Exp
drop temp
gen Spread_ExpReps=Spread_ExpAvg-Spread
gen DRepeated_Spread=1 if Spread_ExpReps==0
replace DRepeated_Spread=0 if missing(DRepeated_Spread)
sum DRepeated_Spread

// - Create merge variable
gen Spread_m=Spread if DRepeated_Spread==1
gen temp=Weight*Spread
bysort IdExposure: egen temp2=sum(temp)
replace Spread_m=temp2 if DRepeated_Spread==0
drop temp temp2

//Merge PIncome, SIncome, and TIncome - method: keep the first
bysort IdExposure: gen temp=_n
gen temp2=PIncome if temp==1
replace temp2=0 if missing(temp2)
bysort IdExposure: egen PIncome_m=max(temp2)
gen temp3=SIncome if temp==1
replace temp3=0 if missing(temp3)
bysort IdExposure: egen SIncome_m=max(temp3)
```

```
gen temp4=TIncome if temp==1
replace temp4=0 if missing(temp4)
bysort IdExposure: egen TIncome_m=max(temp4)
drop temp temp2 temp3 temp4

//Merge Exposures
keep *_m D* Type*
bysort IdExposure_m: keep if _n==1
save "`path'\Part7TypeIII_merged.dta"
```

7.6 Merge Type IV exposures

Type IV exposures have the same IdBorrower, OriginationDate, OriginalHouseValue, and CutOffDate. To merge information I (1) check for possible repetitions, (2) merge variables that are equal within the exposure, (3) merge *MaturityDate*, *Term*, *Spread* using weighted average, (4) merge *Amount* using sum, and (5) merge *IdProperty*, *PostalCode*, *Index*, *PIncome*, *SIncome*, and *TIncome* by keeping the first record.

Stata Code:

```
//Merge Type IV Exposures
clear all
use "`path'\Part7TypeIV.dta"

//Number of lines by exposure and weights
bysort IdExposure: egen Reps_Exp=sum(One)
bysort IdExposure: egen temp=sum(Amount)
gen Weight=Amount/temp
drop temp

//Check for possible repetitions. If more than one loan has the same amount, then it is
    possibly repeated
bysort IdExposure: egen Amount_Exp=sum(Amount)
gen Amount_ExpAvg=Amount_Exp/Reps_Exp
gen Amount_ExpReps=Amount_ExpAvg-Amount
gen DRepeated_Amount=1 if Amount_ExpReps==0
replace DRepeated_Amount=0 if missing(DRepeated_Amount)
```

```
sum DRepeated_Amount // if any DRepeated_Amount=0, then check for repetitions. Otherwise
    continue

//Merge variables that are equal within the Exposure
gen IdBank_m=IdBank
gen IdBorrower_m=IdBorrower
gen IdExposure_m=IdExposure
gen OriginationDate_m=OriginationDate
format OriginationDate_m %td
gen HouseValue_m=HouseValue
gen CutOffDate_m=CutOffDate
format CutOffDate_m %td

//Merge MaturityDate - method: weighted average
// - Identify pattern
bysort IdExposure: egen temp=sum(MaturityDate)
gen MaturityDate_ExpAvg=temp/Reps_Exp
drop temp
gen MaturityDate_ExpReps=MaturityDate_ExpAvg-MaturityDate
gen DRepeated_MaturityDate=1 if MaturityDate_ExpReps==0
replace DRepeated_MaturityDate=0 if missing(DRepeated_MaturityDate)
sum DRepeated_MaturityDate

// - Create merge variable
gen MaturityDate_m=MaturityDate if DRepeated_MaturityDate==1
gen temp=Weight*MaturityDate
bysort IdExposure: egen temp2=sum(temp)
replace MaturityDate_m=temp2 if DRepeated_MaturityDate==0
format MaturityDate_m %td
drop temp temp2

//Merge Term - method: weighted average
// - Identify pattern
bysort IdExposure: egen temp=sum(Term)
gen Term_ExpAvg=temp/Reps_Exp
drop temp
gen Term_ExpReps=Term_ExpAvg-Term
gen DRepeated_Term=1 if Term_ExpReps==0
replace DRepeated_Term=0 if missing(DRepeated_Term)
sum DRepeated_Term
```

```
// - Create merge variable
gen Term_m=Term if DRepeated_Term==1
gen temp=Weight*Term
bysort IdExposure: egen temp2=sum(temp)
replace Term_m=temp2 if DRepeated_Term==0
drop temp temp2

//Merge Amount - method: sum
// - Create merge variable
bysort IdExposure: egen temp=sum(Amount)
gen Amount_m=temp
drop temp

//Merge IdProperty - method: keep the first
bysort IdExposure: gen temp=_n
gen temp2=IdProperty if temp==1
replace temp2=0 if missing(temp2)
bysort IdExposure: egen IdProperty_m=max(temp2)
drop temp temp2

//Merge PostalCode - method: keep the first
bysort IdExposure: gen temp=_n
gen temp2=PostalCode if temp==1
replace temp2=0 if missing(temp2)
bysort IdExposure: egen PostalCode_m=max(temp2)
drop temp temp2

//Merge Index - method: keep the first
// - Identify pattern
bysort IdExposure: egen temp=sum(Index)
gen Index_ExpAvg=temp/Reps_Exp
drop temp
gen Index_ExpReps=Index_ExpAvg-Index
gen DRepeated_Index=1 if Index_ExpReps==0
replace DRepeated_Index=0 if missing(DRepeated_Index)
sum DRepeated_Index // if any DRepeated_Index=0, then evaluate

//*Some observations have index Eur6M and Eur3M. I decided to consider only the index of
    the most representative loan*
```

```
bysort IdExposure: gen temp=_n
gen temp2=Index if temp==1
replace temp2=0 if missing(temp2)
bysort IdExposure: egen Index_m=max(temp2)
drop temp temp2

//Merge Spread - method: weighted average
// - Identify pattern
bysort IdExposure: egen temp=sum(Spread)
gen Spread_ExpAvg=temp/Reps_Exp
drop temp
gen Spread_ExpReps=Spread_ExpAvg-Spread
gen DRepeated_Spread=1 if Spread_ExpReps==0
replace DRepeated_Spread=0 if missing(DRepeated_Spread)
sum DRepeated_Spread

// - Create merge variable
gen Spread_m=Spread if DRepeated_Spread==1
gen temp=Weight*Spread
bysort IdExposure: egen temp2=sum(temp)
replace Spread_m=temp2 if DRepeated_Spread==0
drop temp temp2

//Merge PIncome, SIncome, and TIncome - method: keep the first
bysort IdExposure: gen temp=_n
gen temp2=PIncome if temp==1
replace temp2=0 if missing(temp2)
bysort IdExposure: egen PIncome_m=max(temp2)
gen temp3=SIncome if temp==1
replace temp3=0 if missing(temp3)
bysort IdExposure: egen SIncome_m=max(temp3)
gen temp4=TIncome if temp==1
replace temp4=0 if missing(temp4)
bysort IdExposure: egen TIncome_m=max(temp4)
drop temp temp2 temp3 temp4



//Merge Exposures
keep *_m Type*
bysort IdExposure_m: keep if _n==1
```

```
save "`path'\Part7TypeIV_merged.dta"
```

7.7 Compile exposures into a single file

Stata Code:

```
//Merge files into single dataset
clear all
use "`path'\Part7TypeIandVI_merged.dta"
append using "`path'\Part7TypeII_merged.dta"
append using "`path'\Part7TypeIII_merged.dta"
append using "`path'\Part7TypeIV_merged.dta"
save "`path'\Part7All_merged.dta"
```

Exposures: 310,405

Deliverable: Part7All_merged.dta

8 Data correction - After merging

After merging, I have reduced the timespan and reinspected a set of critical points. I have found some inconsistencies in *OriginalHouseValue*, *Amount* and Income variables.

8.1 Reduce the timespan to 1999-2016

Stata Code:

```
//Reduce the timespan to 1999-2016
drop if OriginationDate_m < date("19990101","YMD")
drop if OriginationDate_m > date("20161231","YMD")
```

8.2 Drop unreasonable House Values and Amounts

Critical points:

- Given the Portuguese housing market reality, *OriginalHouseValue* below 50,000 euros and above 6,000,000 euros are not probable. The same for mortgage amounts below 25,000 euros.

Corrective actions:

- Drop observations with *OriginalHouseValue* below 50,000 and above 6,000,000 euros.

- Drop observations with *Amount* below 25,000 euros.

Stata Code:

```
//Drop unreasonable house values and amounts
drop if HouseValue_m<50000
drop if HouseValue_m>6000000
drop if Amount_m<25000
gen LTV_m=(Amount_m/HouseValue_m)*100
```

8.3 Correct Income variables and drop unreasonable values

Critical points:

- After inspecting income data, I conclude that some banks might be reportinh monthly income whether others might be reporting annual income. There many observations with income below the minimum wage, which is not reasonable.

Corrective actions:

- Drop observations with *PIncome* below 500 euros.

- If income variables are below the equivalent to the annual minimum wage, then they are for sure reporting monthly income. In those cases, multiply income by 14 months.

Stata Code:

```
//Drop unreasonable Income and correction
drop if PIncome_m<500

//Some banks reported PIncome_m monthly, instead of annually. To inspect if this is the
    case, I compare the PIncome_m with annual minimum wage at the time of the origination
gen MinWage_m=4280.7 if Year_m==1999
replace MinWage_m=4455.3 if Year_m==2000
replace MinWage_m=4678.7 if Year_m==2001
replace MinWage_m=4872.2 if Year_m==2002
replace MinWage_m=4992.4 if Year_m==2003
replace MinWage_m=5118.4 if Year_m==2004
replace MinWage_m=5245.8 if Year_m==2005
replace MinWage_m=5402.6 if Year_m==2006
replace MinWage_m=5642 if Year_m==2007
replace MinWage_m=5964 if Year_m==2008
replace MinWage_m=6300 if Year_m==2009
replace MinWage_m=6650 if Year_m==2010
replace MinWage_m=6790 if Year_m==2011
replace MinWage_m=6790 if Year_m==2012
replace MinWage_m=6790 if Year_m==2013
replace MinWage_m=6790 if Year_m==2014
replace MinWage_m=7070 if Year_m==2015
replace MinWage_m=7420 if Year_m==2016

gen PBelowMinWage_m=1 if (PIncome_m<MinWage_m)
replace PBelowMinWage_m=0 if missing(PBelowMinWage_m)
gen SBelowMinWage_m=1 if (SIncome_m<MinWage_m)
replace SBelowMinWage_m=0 if missing(SBelowMinWage_m)
gen PIncomeC_m=PIncome_m*14 if (PBelowMinWage_m==1)
replace PIncomeC_m=PIncome_m if missing(PIncomeC_m)
gen SIncomeC_m=SIncome_m*14 if (SBelowMinWage_m==1)
replace SIncomeC_m=SIncome_m if missing(SIncomeC_m)
gen TIncomeC_m=PIncomeC_m+SIncomeC_m
```

Exposures: 199,211

Deliverable: Part8.dta