DeFi Notebook:

DAR Assignment 6

Kaiwen Min

09/01/2021

Introductory Decentralized Finance (DeFi) Research Notebook

This notebook is broken into two main parts:

```
* Part 1 is a basic introduction to github and RStudio Server
* Part 2 is an introduction to the DeFi transaction dataset
```

This R Notebook and its related R scripts provide a very basic introduction to an interesting **Decentralized Finance (DeFi)** dataset. All data was obtained by querying an API on The Graph, an indexing protocol for querying networks like Ethereum, for transaction data based on the AAVE protocol. For more information on AAVE see the AAVE developer notes. The AAVE protocol is based on Ethereum, an important cryptocurrency platform.

The RPI github repository for all the code required for this notebook, including a snapshots of AAVE transaction and user data, may be found at:

- https://github.rpi.edu/DataINCITE/IDEA-Blockchain

The IDEA-Blockchain github also contains notebooks used to harvest the AAVE dataset, which you are welcome to examine.

BEFORE YOU BEGIN

To contribute or submit to any RPI github repository you must validate your RPI github.com ID and send a confirmation email to John Erickson at `erickj4@rpi.edu`. Please do the following **now**:

- Browse to http://github.rpi.edu
- Log in using your RPI credentials
- **PLEASE DO THIS IMMEDIATELY BEFORE READING ANY FURTHER!!**

DAR ASSIGNMENT 1: CLONING A NOTEBOOK AND UPDATING THE REPOSITORY

In this assignment we're asking you to...

- clone the `IDEA-Blockchain` github repository...
- create a personal branch using git...
- copy and change an R notebook...
- generate ("knit") a PDF based on that notebook, and...
- add these new files by "committing" and "pushing" to the github repository

The instructions which follow explain how to accomplish this.

NOTE: For DAR Fall 2021 you *must* be using RStudio Server on the IDEA Cluster. Instructions for accessing "The Cluster" appear at the end of this notebook. Also, don't forget to validate your RPI github ID as above and email `erickj4@rpi.edu`

Cloning an RPI github repository

The recommended procedure for cloning and using this repository is as follows:

- Access the RPI network via VPN
  - See https://itssc.rpi.edu/hc/en-us/articles/360008783172-VPN-Connection-and-Installation for information
- Access RStudio Server on the IDEA Cluster at http://lp01.idea.rpi.edu/rstudio-ose/
  - You must be on the RPI VPN!!
- Access the Linux shell on the IDEA Cluster by clicking the **Terminal** tab of RStudio Server (lower left panel).
  - You now see the Linux shell on the IDEA Cluster
  - `cd` (change directory) to enter your home directory using: `cd ~`
  - Type `pwd` to confirm
  - NOTE: Advanced users may use `ssh` to directly access the Linux shell from a macOS or Linux command line
- Type `git clone https://github.rpi.edu/DataINCITE/IDEA-Blockchain.git` from within your `home` directory
  - This will create a new directory `IDEA-Blockchain`
- In the Linux shell, `cd` to `IDEA-Blockchain/DefiResearch/StudentNotebooks`
  - Type `ls -al` to list the current contents
  - Don't be surprised if you see many files!
- In the Linux shell, type `git checkout -b dar-yourrcs` where `yourrcs` is your RCS id
  - For example, if your RCS is `erickj4`, your new branch should be `dar-erickj4`
  - It is *critical* that you include your RCS id in your branch id
- Now in the RStudio Server UI, navigate to the `IDEA-Blockchain/DefiResearch/StudentNotebooks` directory via the **Files** panel (lower right panel)
  - Under the **More** menu, set this to be your R **working directory**
  - *Setting the correct working directory is essential for interactive R use!*

REQUIRED FOR ASSIGMENT 1

1. In RStudio...
   - Open `blockchain-notebook-f21.Rmd`
     - **NOTE:** When opening this `.Rmd` RStudio may warning you that some required packages are not installed, and ask you if you wish to install them.
     - Go ahead and say yes. **DO NOT INTERUPT PACKAGE INSTALLATION!**
   - **Save As...** using a *new, original, descriptive* filename that **includes your RCS ID!**
   - Example filename for user `erickj4`: `erickj4-assignment1-f21.Rmd`
   - You should see the file appear in the listing in your **Files** panel.
2. Edit your new notebook using RStudio and save your results...
   - Change the `title:` and `subtitle:` headers (at the top of the file)
   - Change the `author:`
   - Change the `date:`
   - **Save** your changes (`<cntl-s>` works!)
3. Use the RStudio `Knit` command (top of the editing window) to create HTML and/or PDF files
   - Use the down arrow next to the word `Knit` and select **Knit to HTML** (Optional but handy for previewing)
   - Use the down arrow next to the word `Knit` and select **Knit to PDF** (Required for assignment)
   - Repeat as necessary
4. In the Linux terminal, use `git add` to add each new file you want to add to the repository

- Type: `git add yourfilename.Rmd`
- Type: `git add yourfilename.html` (created when you knit to HTML)
- Type: `git add yourfilename.pdf` (created when you knit to PDF)

5. When you're ready, in Linux commit your changes:
   - Type: `git commit -m "some comment"` where "some comment" is a useful comment describing your changes
   - This commits your changes to your local repo, and sets the stage for your next operation.
6. Finally, "push" the commits in your working branch to the RPI github repo
   - Type: `git push origin dar-yourrcs` (where `dar-yourrcs` is the branch you've been working in)
   - Your changes are now safely on the RPI github, under your branch name.
7. **REQUIRED:** On the RPI github, submit a pull request for your branch:
   - In a web browser, navigate to https://github.rpi.edu/DataINCITE/IDEA-Blockchain
   - In the drop-down that says **Master** (top left above the list of files) click and select *your* branch
   - It should give you the option to create a pull request
   - **Submit a pull request for your branch**
   - Eventually one of the DAR instructors will merge your branch into the master branch of the repo.
8. Confirm what you just did; make the following additional edits:
   - What is the location of the github: _https://github.rpi.edu/DataINCITE/IDEA-Blockchain/tree/dar-mink3_
   - What is your github ID: *mink3*
   - What is the name of your new branch: *dar-mink3*
   - What is the name of your new (copied) notebook: *mink3-assignment1-f21.Rmd*
   - Save your changes and `knit` an updated PDF.
9. Re-commit these fresh changes to the github
   - Confirm that you are still in your branch; type: `git branch`
   - `git add` your `Rmd` and `PDF`
   - `git commit -m` with a fresh message
   - `git push origin` your branch.
   - Go to github and select your branch again; if your previous push has already been merged, submit another pull request.
   - More than likely your previous pull request hasn't been merged and you newest commit was automatically added to your existing request.
10. Download your PDF and upload to LMS:

- In the RStudio **Files** panel, select your newly created PDF file (check the checkbox to its left) and select **Export** under the **More** menu.
- This downloads your PDF file to your personal machine.
- Now proceed to LMS and upload the PDF you created!

Please also see this handy github "cheatsheet": https://education.github.com/git-cheat-sheet-education.pdf

Exploring a DeFi Transaction Dataset using AAVE

This section is provided as background and is not required for Assignment 1.

What is AAVE?

From the developer site: *Aave is a decentralised non-custodial liquidity protocol where users can participate as depositors or borrowers. Depositors provide liquidity to the market to earn a passive income, while borrowers are able to borrow in an over-collateralised (perpetually) or under-collateralised (one-block liquidity) fashion... The (Aave) protocol is implemented as a set of smart contracts on top of the Ethereum blockchain. Smart contracts guarantee safety and do not require a middleman.*

For (much) more detail refer to the AAVE Protocol V2.0 Whitepaper

Prepare Transaction Data

We begin by loading our prepared AAVE transaction data into a dataframe. The dataset has over 400,000 rows, and 27 columns.

We are directly loading the dataframe from an Rds archive instead of a CSV file to conserve space.

```
#load Rds (binary version of csv file) into dataframe
df<-read_rds('../../Data/transactionsv2.rds')

# Let's take a quick look
head(df)
```

```
##       amount borrowRate borrowRateMode   onBehalfOf         pool reserve
## 1      15.00  0.2590658       Variable 1.117217e+48 1.034668e+48    WETH
## 2   41501.63  6.2749368       Variable 8.502518e+47 1.034668e+48     DAI
## 3 7000000.00  2.5896280       Variable 4.635974e+47 1.034668e+48    USDT
## 4   15000.00  8.8025409       Variable 3.735263e+47 1.034668e+48    USDC
## 5    8193.19 48.7470516         Stable 6.896232e+47 1.034668e+48    USDC
## 6   11000.00  3.2250550       Variable 1.089455e+48 1.034668e+48    USDT
##    timestamp         user   type reservePriceETH reservePriceUSD  amountUSD
## 1 1633275840 1.168069e+48 borrow    1.0000000000    3421.8708189   51328.06
## 2 1621340435 8.502518e+47 borrow    0.0002852900       0.9948044   41286.00
## 3 1622477822 4.635974e+47 borrow    0.0003812835       1.0000000 7000000.00
## 4 1619775984 3.735263e+47 borrow    0.0003611000       1.0043389   15065.08
## 5 1615481632 6.896232e+47 borrow    0.0005562201       0.9993909    8188.20
## 6 1626914745 1.089455e+48 borrow    0.0004971100       1.0000000   11000.00
##   collateralAmount collateralReserve liquidator principalAmount
## 1               NA                           NA              NA
## 2               NA                           NA              NA
## 3               NA                           NA              NA
## 4               NA                           NA              NA
## 5               NA                           NA              NA
## 6               NA                           NA              NA
##   principalReserve reservePriceETHPrincipal reservePriceUSDPrincipal
## 1                                        NA                       NA
## 2                                        NA                       NA
## 3                                        NA                       NA
## 4                                        NA                       NA
## 5                                        NA                       NA
## 6                                        NA                       NA
##   reservePriceETHCollateral reservePriceUSDCollateral amountUSDPincipal
## 1                        NA                        NA               NA
## 2                        NA                        NA               NA
## 3                        NA                        NA               NA
## 4                        NA                        NA               NA
## 5                        NA                        NA               NA
## 6                        NA                        NA               NA
##   amountUSDCollateral borrowRateModeFrom borrowRateModeTo stableBorrowRate
## 1                  NA                                                  NA
## 2                  NA                                                  NA
## 3                  NA                                                  NA
## 4                  NA                                                  NA
## 5                  NA                                                  NA
## 6                  NA                                                  NA
##   variableBorrowRate fromState toState protocolContract     user_alias
```

```
## 1                 NA                                True Gladys Marquez
## 2                 NA                               False  Angel Prather
## 3                 NA                               False   Jack Crowley
## 4                 NA                               False    Jim Dickens
## 5                 NA                               False  Leonard Reyes
## 6                 NA                               False      Jill Carn
##   onBehalfOf_alias            datetime
## 1  Evelyn Terrazas 2021-10-03 15:44:00
## 2    Angel Prather 2021-05-18 12:20:35
## 3     Jack Crowley 2021-05-31 16:17:02
## 4      Jim Dickens 2021-04-30 09:46:24
## 5    Leonard Reyes 2021-03-11 16:53:52
## 6        Jill Carn 2021-07-22 00:45:45
```

```
str(df)
```

```
## 'data.frame':    745612 obs. of  33 variables:
##  $ amount                   : num  15 41502 7000000 15000 8193 ...
##  $ borrowRate               : num  0.259 6.275 2.59 8.803 48.747 ...
##  $ borrowRateMode           : Factor w/ 3 levels "","Stable","Variable": 3 3 3 3 2 3 3 3 3 3 ...
##  $ onBehalfOf               : num  1.12e+48 8.50e+47 4.64e+47 3.74e+47 6.90e+47 ...
##  $ pool                     : num  1.03e+48 1.03e+48 1.03e+48 1.03e+48 1.03e+48 ...
##  $ reserve                  : Factor w/ 53 levels "","AAVE","AmmBptBALWETH",..: 50 29 48 47 47 48 48
##  $ timestamp                : int  1633275840 1621340435 1622477822 1619775984 1615481632 1626914745
##  $ user                     : num  1.17e+48 8.50e+47 4.64e+47 3.74e+47 6.90e+47 ...
##  $ type                     : Factor w/ 7 levels "borrow","collateral",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ reservePriceETH          : num  1 0.000285 0.000381 0.000361 0.000556 ...
##  $ reservePriceUSD          : num  3421.871 0.995 1 1.004 0.999 ...
##  $ amountUSD                : num  51328 41286 7000000 15065 8188 ...
##  $ collateralAmount         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ collateralReserve        : Factor w/ 27 levels "","AAVE","AmmBptBALWETH",..: 1 1 1 1 1 1 1 1 1 1
##  $ liquidator               : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ principalAmount          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ principalReserve         : Factor w/ 29 levels "","AmmDAI","AmmUSDC",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ reservePriceETHPrincipal : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ reservePriceUSDPrincipal : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ reservePriceETHCollateral: num  NA NA NA NA NA NA NA NA NA NA ...
##  $ reservePriceUSDCollateral: num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amountUSDPincipal        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amountUSDCollateral      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ borrowRateModeFrom       : Factor w/ 3 levels "","Stable","Variable": 1 1 1 1 1 1 1 1 1 1 ...
##  $ borrowRateModeTo         : Factor w/ 3 levels "","Stable","Variable": 1 1 1 1 1 1 1 1 1 1 ...
##  $ stableBorrowRate         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ variableBorrowRate       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ fromState                : Factor w/ 3 levels "","False","True": 1 1 1 1 1 1 1 1 1 1 ...
##  $ toState                  : Factor w/ 3 levels "","False","True": 1 1 1 1 1 1 1 1 1 1 ...
##  $ protocolContract         : Factor w/ 2 levels "False","True": 2 1 1 1 1 1 1 1 1 1 ...
##  $ user_alias               : Factor w/ 51421 levels "Aaron Adams",..: 17915 1861 20155 23570 29991 1
##  $ onBehalfOf_alias         : Factor w/ 50705 levels "Aaron Adams",..: 15651 1831 19875 23236 29581 1
##  $ datetime                 : Factor w/ 406174 levels "2020-11-30 23:11:40",..: 395517 192373 222043
```

```
summary(df)
```

```
##      amount             borrowRate       borrowRateMode     onBehalfOf
##  Min.   :        0   Min.   :    0.0           :635842   Min.   :2.578e+33
```

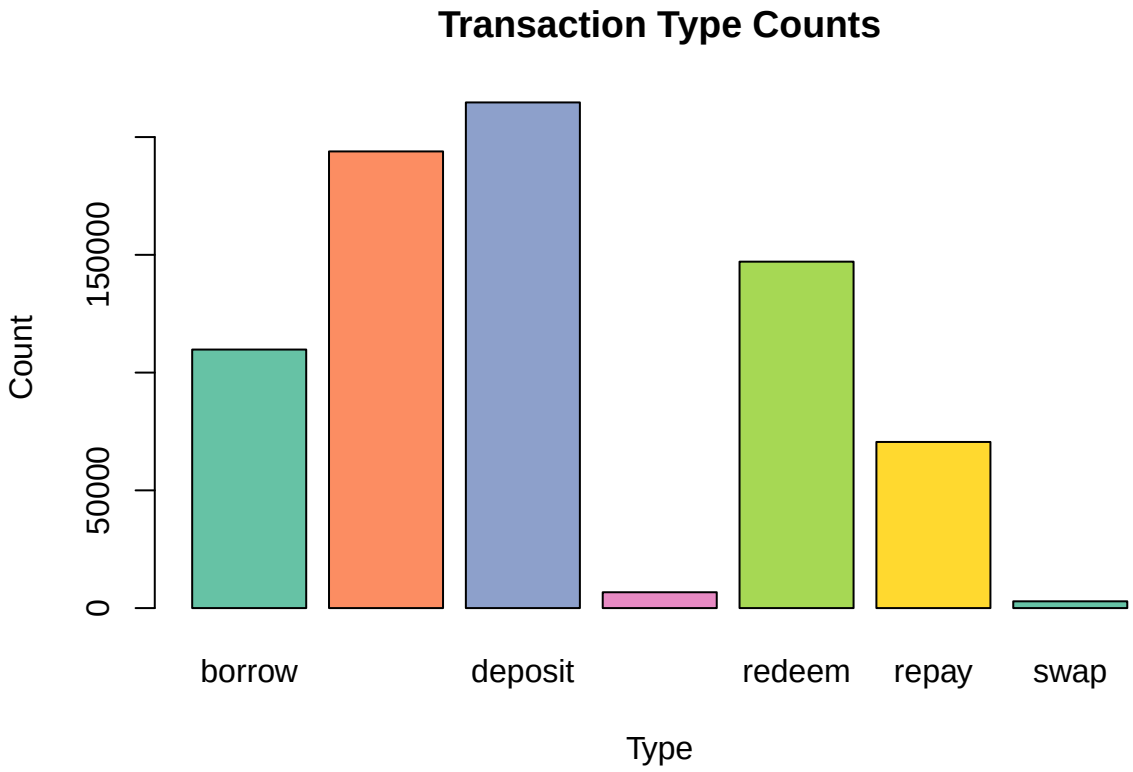
```
##  1st Qu.:       24   1st Qu.:    3.3   Stable  : 19665   1st Qu.:4.065e+47
##  Median :     1500   Median :    4.0   Variable: 90105   Median :7.436e+47
##  Mean   :   223490   Mean   :    9.2                     Mean   :7.533e+47
##  3rd Qu.:    25435   3rd Qu.:   10.6                     3rd Qu.:1.168e+48
##  Max.   :600000000   Max.   :10002.0                     Max.   :1.461e+48
##  NA's   :203509      NA's   :635842                      NA's   :203509
##       pool              reserve         timestamp             user
##  Min.   :9.862e+47   WETH   :171895   Min.   :1.607e+09   Min.   :1.000e+00
##  1st Qu.:1.035e+48   USDC   :152630   1st Qu.:1.615e+09   1st Qu.:4.199e+47
##  Median :1.035e+48   USDT   : 81628   Median :1.622e+09   Median :7.962e+47
##  Mean   :1.034e+48   DAI    : 78570   Mean   :1.621e+09   Mean   :7.783e+47
##  3rd Qu.:1.035e+48   WBTC   : 42990   3rd Qu.:1.626e+09   3rd Qu.:1.168e+48
##  Max.   :1.035e+48   LINK   : 42001   Max.   :1.634e+09   Max.   :1.461e+48
##                      (Other):175898
##           type         reservePriceETH     reservePriceUSD
##  borrow     :109770   Min.   :0.000e+00   Min.   :0.000e+00
##  collateral :193904   1st Qu.:0.000e+00   1st Qu.:1.000e+00
##  deposit    :214707   Median :0.000e+00   Median :1.000e+00
##  liquidation:  6731   Mean   :3.093e+05   Mean   :6.620e+08
##  redeem     :147090   3rd Qu.:1.000e+00   3rd Qu.:1.808e+03
##  repay      : 70536   Max.   :1.557e+10   Max.   :6.119e+13
##  swap       :  2874   NA's   :203509      NA's   :203509
##    amountUSD         collateralAmount  collateralReserve   liquidator
##  Min.   :        0   Min.   :      0          :738881    Min.   :2.379e+38
##  1st Qu.:     2986   1st Qu.:      1   WETH   :  2910    1st Qu.:2.755e+47
##  Median :    17560   Median :     13   LINK   :  1391    Median :6.125e+47
##  Mean   :   485120   Mean   :   5443   WBTC   :   719    Mean   :6.629e+47
##  3rd Qu.:    97638   3rd Qu.:    250   AAVE   :   342    3rd Qu.:1.048e+48
##  Max.   :768950954   Max.   :4638724   UNI    :   244    Max.   :1.457e+48
##  NA's   :203509      NA's   :738881    (Other):  1125    NA's   :738881
##  principalAmount   principalReserve reservePriceETHPrincipal
##  Min.   :      0          :738881   Min.   : 0.0
##  1st Qu.:   1026   USDC   :  2322   1st Qu.: 0.0
##  Median :   4651   USDT   :  1663   Median : 0.0
##  Mean   :  69034   DAI    :  1540   Mean   : 0.2
##  3rd Qu.:  22599   GUSD   :   249   3rd Qu.: 0.0
##  Max.   :4475668   TUSD   :   186   Max.   :42.0
##  NA's   :738881    (Other):   771   NA's   :738881
##  reservePriceUSDPrincipal reservePriceETHCollateral reservePriceUSDCollateral
##  Min.   :    0.1          Min.   :1.502e+14         Min.   :0.000e+00
##  1st Qu.:    1.0          1st Qu.:9.482e+15         1st Qu.:2.100e+01
##  Median :    1.0          Median :1.000e+18         Median :1.809e+03
##  Mean   :  350.3          Mean   :2.113e+21         Mean   :4.411e+06
##  3rd Qu.:    1.0          3rd Qu.:1.000e+18         3rd Qu.:2.550e+03
##  Max.   :83819.1          Max.   :9.471e+23         Max.   :2.607e+09
##  NA's   :738881           NA's   :738881            NA's   :738881
##  amountUSDPincipal amountUSDCollateral borrowRateModeFrom borrowRateModeTo
##  Min.   :      0   Min.   :      0            :742738           :742738
##  1st Qu.:   1263   1st Qu.:   1365     Stable  :  1349    Stable  :  1525
##  Median :   5022   Median :   5401     Variable:  1525    Variable:  1349
##  Mean   :  71425   Mean   :  76506
##  3rd Qu.:  23803   3rd Qu.:  25882
##  Max.   :4571839   Max.   :5029023
##  NA's   :738881    NA's   :738881
```

```
##  stableBorrowRate variableBorrowRate fromState       toState
##  Min.   :  0.0    Min.   :  0.0           :551708        :551708
##  1st Qu.:  8.9    1st Qu.:  3.8      False: 98690   False: 95938
##  Median : 10.8    Median :  3.9      True : 95214   True : 97966
##  Mean   : 11.8    Mean   :  5.8
##  3rd Qu.: 12.0    3rd Qu.:  5.2
##  Max.   :154.7    Max.   :148.7
##  NA's   :742738   NA's   :742738
##  protocolContract           user_alias            onBehalfOf_alias
##  False:685824     Gladys Marquez : 59787   John Dunn      :203509
##  True : 59788     Shela Hazzard  : 32168   Shela Hazzard  : 32173
##                   Darla Haas     : 25899   Gladys Marquez : 20746
##                   Janelle Lazarus: 25468   Peggy Strawser : 11298
##                   Peggy Strawser : 21503   Janelle Lazarus:  9205
##                   Patrick Johnson: 13700   Darla Haas     :  6778
##                   (Other)        :567087   (Other)        :461903
##                 datetime
##  2021-01-16 18:28:22:    40
##  2021-06-18 06:49:50:    38
##  2021-06-18 06:49:44:    34
##  2021-01-17 08:22:32:    32
##  2021-01-16 18:29:12:    24
##  2021-03-06 05:35:28:    20
##  (Other)            :745424
```

Analyze Transaction Types

Next, we will examine the different types of transactions present in the data. We make a bar plot to visualize the number of each transaction types. Deposit is the most common type of transaction, whereas swaps are the most rare.

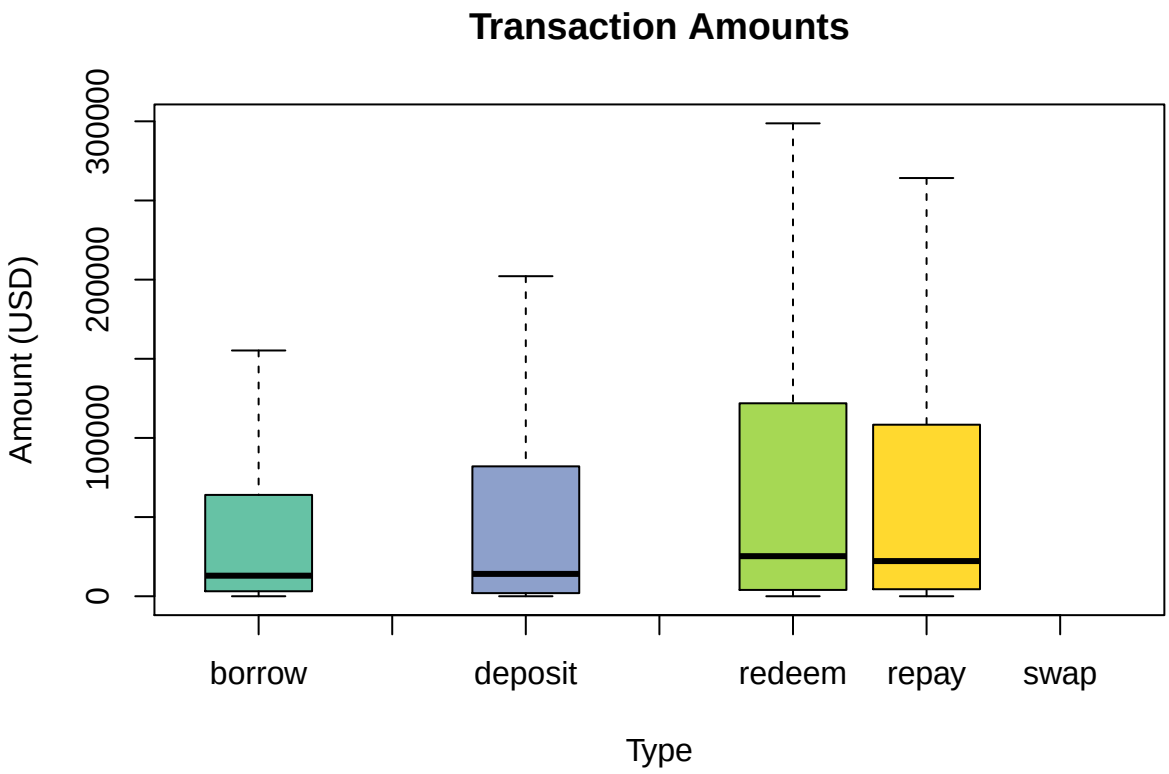
```
#set color palette
colors = brewer.pal(6,"Set2")

#create barplot
barplot(table(df$type), main='Transaction Type Counts', xlab='Type',ylab='Count',col=colors)
```



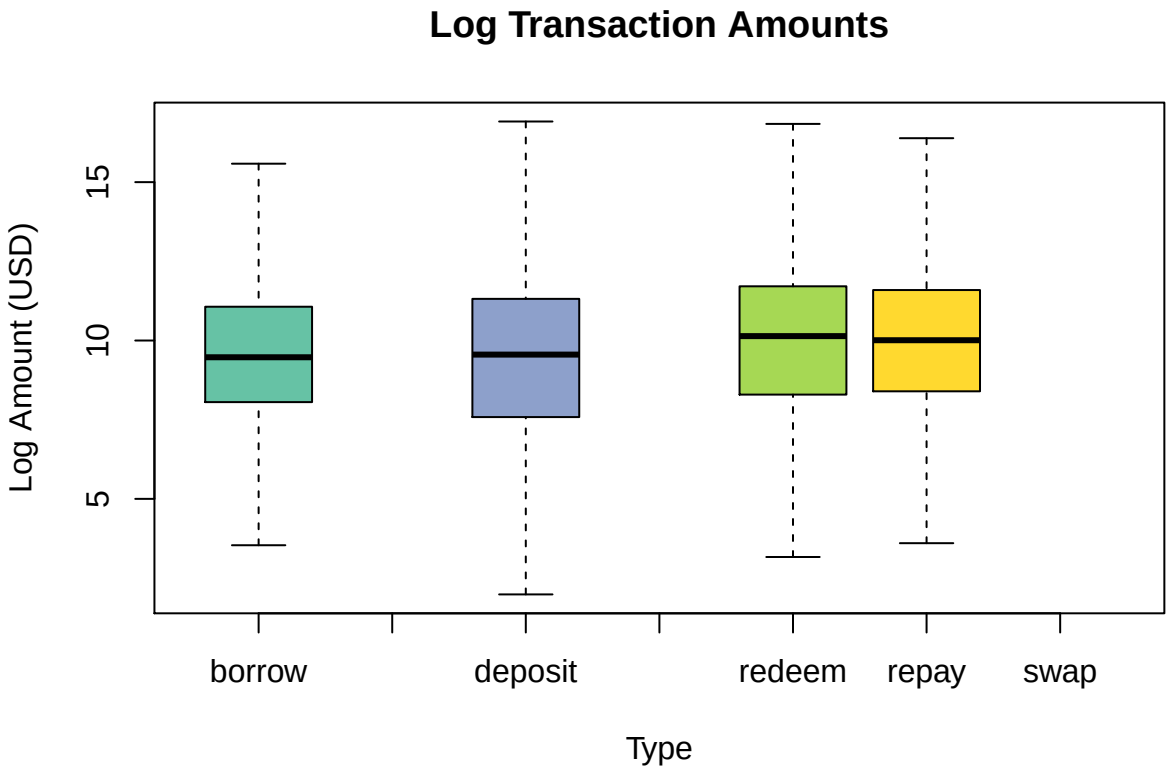
There are more deposits than borrows, because users often need to overcollateralize for loans.

Now, we will examine the amount of US dollars being used in the different types of transactions. We create box plots for the 4 types of transactions that have the amount feature associated with them, and visualize the distribution of that column for the different transactions. We can see that most transactions are completed with very little money.

```
#create boxplot
boxplot(amountUSD~type,data=df,outline=FALSE,col=colors,main="Transaction Amounts",xlab="Type",ylab="Am
```




```
boxplot(log(amountUSD)~type,data=df,outline=FALSE,col=colors,main="Log Transaction Amounts",xlab="Type"
```

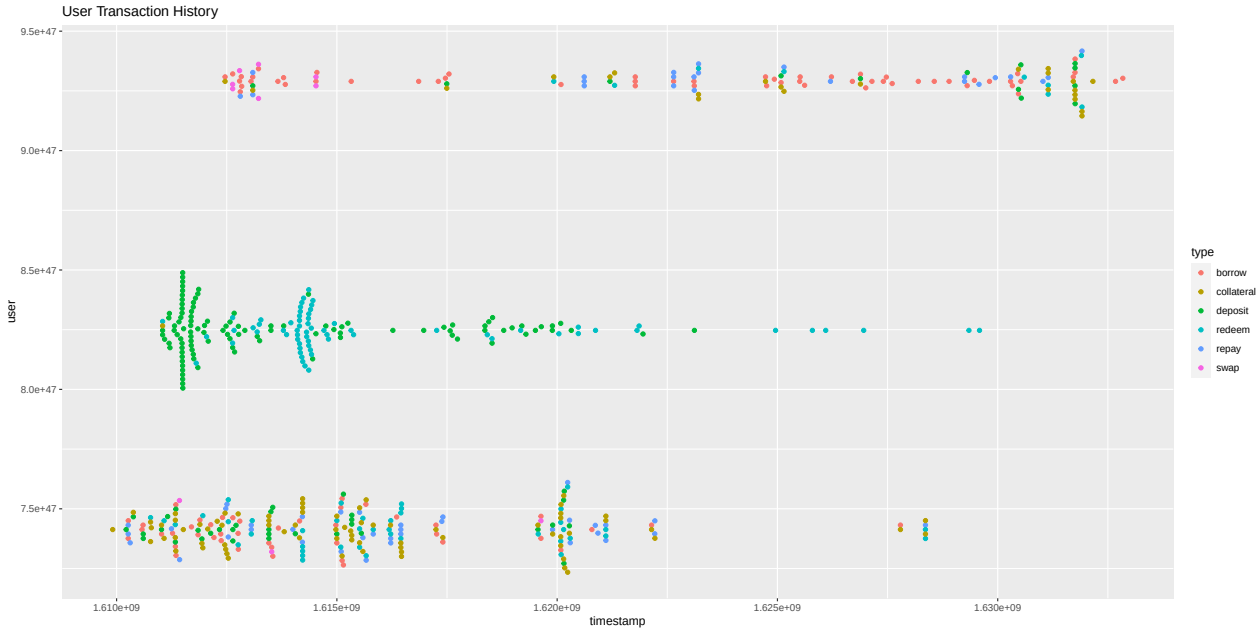


There are many borrows and repays with high transactions amounts, but deposits and redeems have much lower transactions amounts.

Look at Sample User Transaction Histories

Finally, we will examine the transaction history of different users. To do this, we will select 3 random users from the data who have completed between 100 and 300 transactions. Then, we create swarmplots displaying the different types of transactions those users made over time.

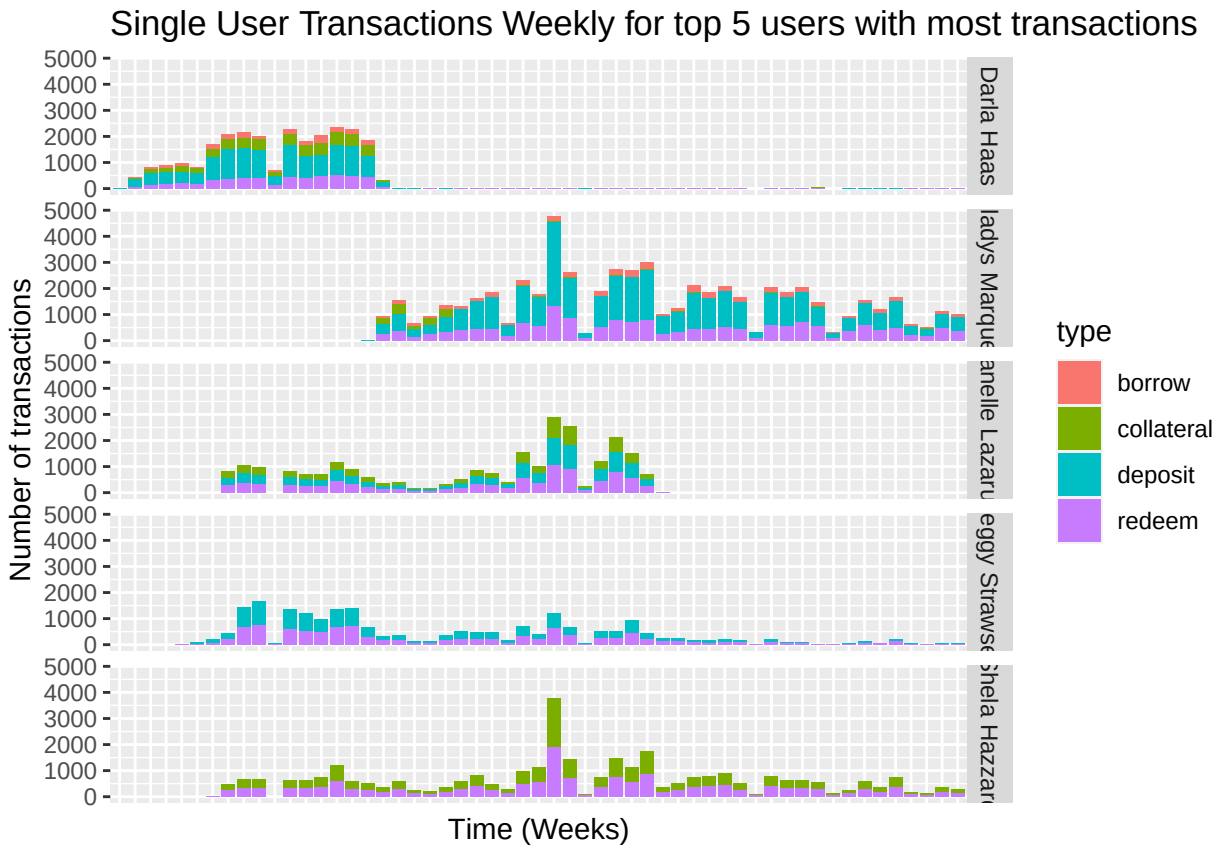
```
#set seed
set.seed(1)

#get 3 random users that have between 100 and 300 transactions
users<-vector(length=3)
count<-0
while(count<=3){
  success<-FALSE
  while(!success){
    #get random user
    ruser<-sample(df$user,1)

    #check for valid number of transactions
    length<-nrow(filter(df,user==ruser))
    if (length>100 && length<300){
      users[count]=ruser
      success<-TRUE
      count<-count+1
    }
  }
}
df.rusers<-filter(df, user %in%users)

#create swarmplot

#liquidations
#borrow
#deposit
#redeem
#repay
#swap

ggplot(df.rusers,aes(user, timestamp,color=type)) +
        geom_beeswarm(cex=1)+
        coord_flip()+
        ggtitle("User Transaction History")
```



Users have very different transactions patterns, which we will try to better understand.

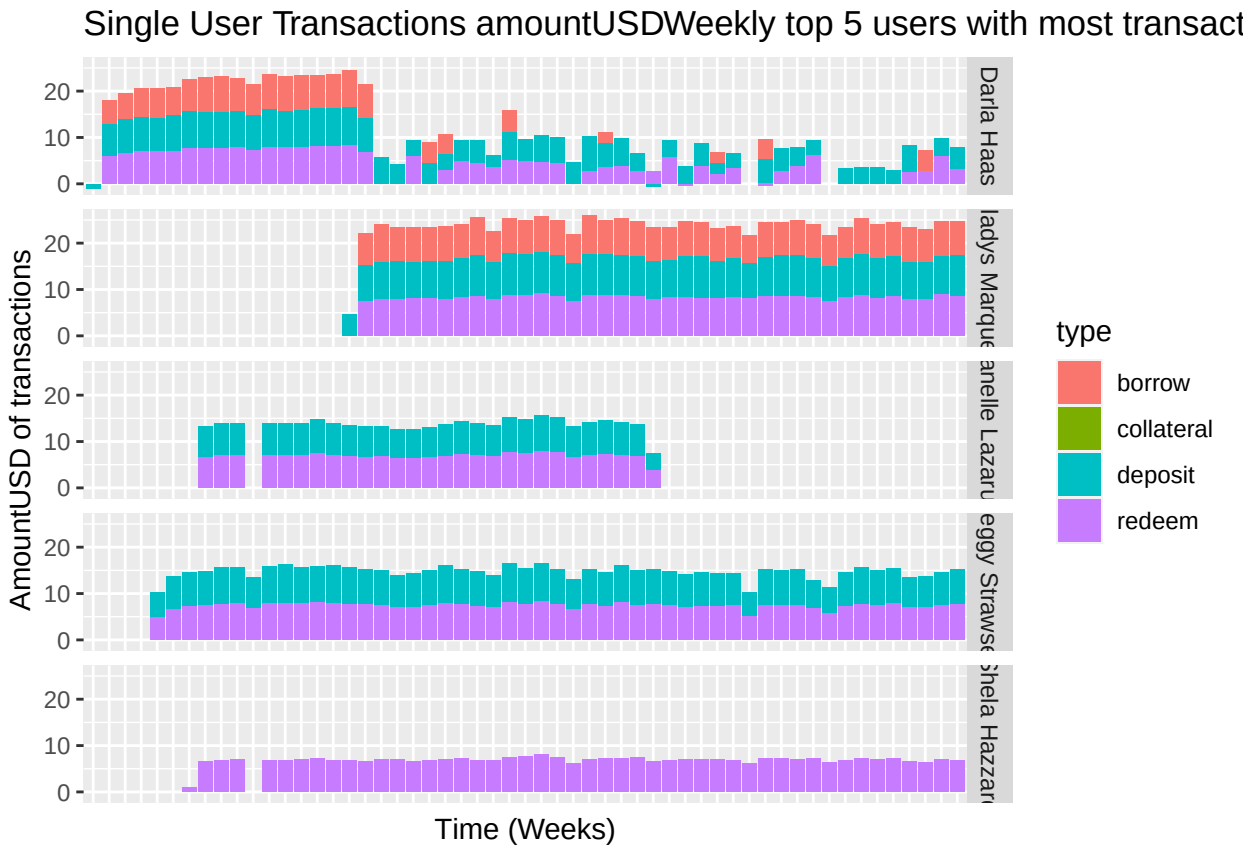
```
users <- plyr::count(as.factor(df$user_alias))
top20transactionuser <- users[order(users$freq, decreasing = TRUE),]$x[1:20]

df.1users<-filter(df, user_alias %in% top20transactionuser[1:5])

#ggplot(df.1users,aes(log10(amountUSD), timestamp,color=type)) +
        #geom_beeswarm(cex=1)+
        #coord_flip()+
        #ggtitle("User Transaction History")
```

```
df.1users$ymd <- as_datetime(df.1users$timestamp) # fix times for the transactions

setDT(df.1users)[, ymd_new := format(as.Date(ymd), '%Y-%m-%V') ] ## '%Y-%m' for just month-year

df.1user <- df.1users %>% dplyr::count(ymd_new, type, user_alias, protocolContract, name = "amount")

ggplot(df.1user) +
  geom_bar(aes(x=ymd_new,y= amount, fill = type),
           stat='identity') + ggtitle("Single User Transactions Weekly for top 5 users with most transa
  theme(
           axis.text.x=element_blank(),
           axis.ticks.x=element_blank())+facet_grid(rows = df.1user$user_alias)
```




```
df.1user <- df.1users %>%
  dplyr::count(ymd_new, type, user_alias, wt=amountUSD, name = "amountUSD")

ggplot(df.1user) +
  geom_bar(aes(x=ymd_new,y= log10(amountUSD), fill = type),
           stat='identity') + ggtitle("Single User Transactions amountUSDWeekly top 5 users with most t
  theme(
           axis.text.x=element_blank(),
           axis.ticks.x=element_blank())+facet_grid(rows = df.1user$user_alias)
```

```
## Warning: Removed 158 rows containing missing values (geom_bar).
```




```
df.1user <- df.1users %>% dplyr::count(user_alias, protocolContract, name = "amount")
df.1user
```

```
##         user_alias protocolContract amount
## 1:      Darla Haas            False  25899
## 2:  Gladys Marquez             True  59787
## 3: Janelle Lazarus            False  25468
## 4:  Peggy Strawser            False  21503
## 5:   Shela Hazzard            False  32168
```

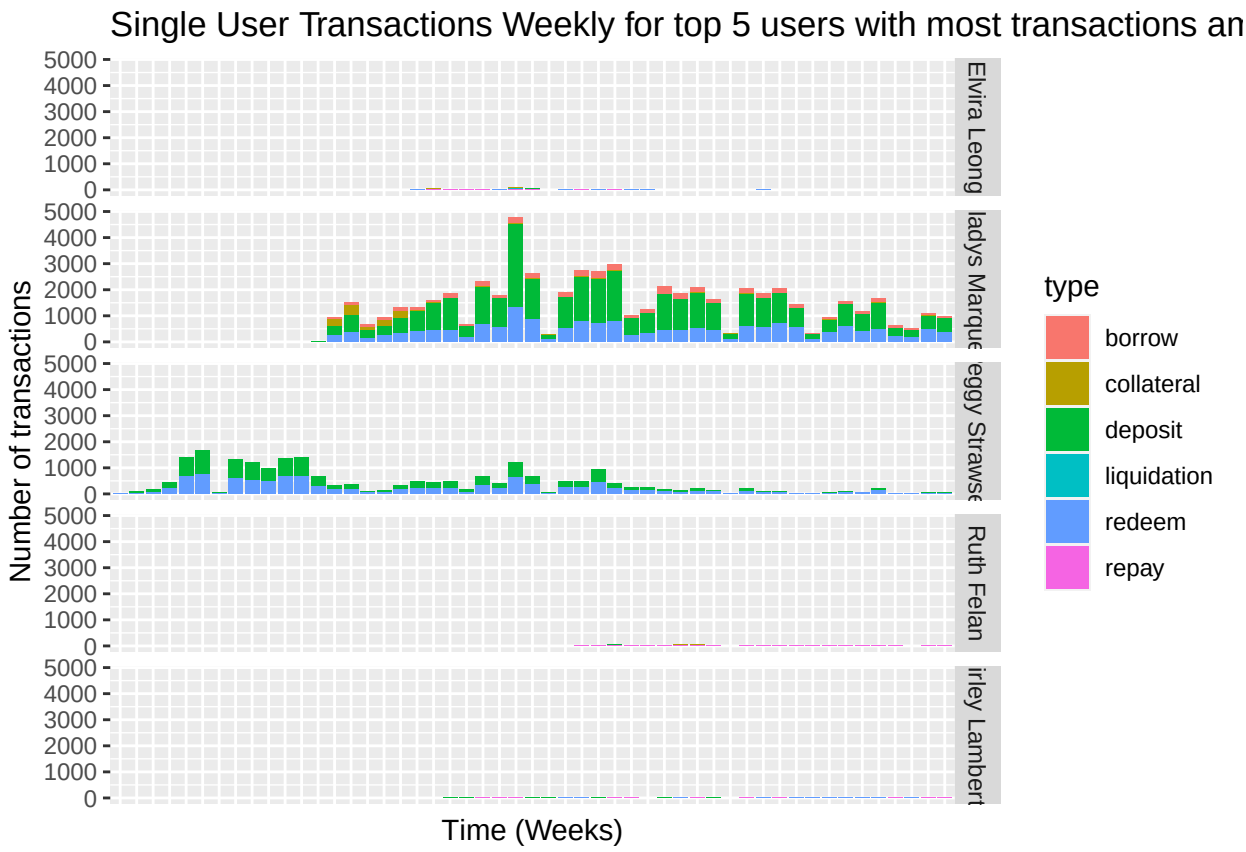
```
df.1user <- df.1users %>%
  dplyr::count(user_alias, wt=amountUSD, protocolContract, name = "amountUSD")
df.1user
```

```
##         user_alias protocolContract   amountUSD
## 1:      Darla Haas            False  2615264911
## 2:  Gladys Marquez             True 27033268579
## 3: Janelle Lazarus            False   736965030
## 4:  Peggy Strawser            False  4581506747
## 5:   Shela Hazzard            False   601088865
```

```
users <- df %>% dplyr::count(user_alias, wt=amountUSD, protocolContract, name = "amountUSD")
top20amountuser <- users[order(users$amountUSD, decreasing = TRUE),]$user_alias[1:20]

df.1users<-filter(df, user_alias %in% top20amountuser[1:5])

#ggplot(df.1users,aes(log10(amountUSD), timestamp,color=type)) +
        #geom_beeswarm(cex=1)+
        #coord_flip()+
```

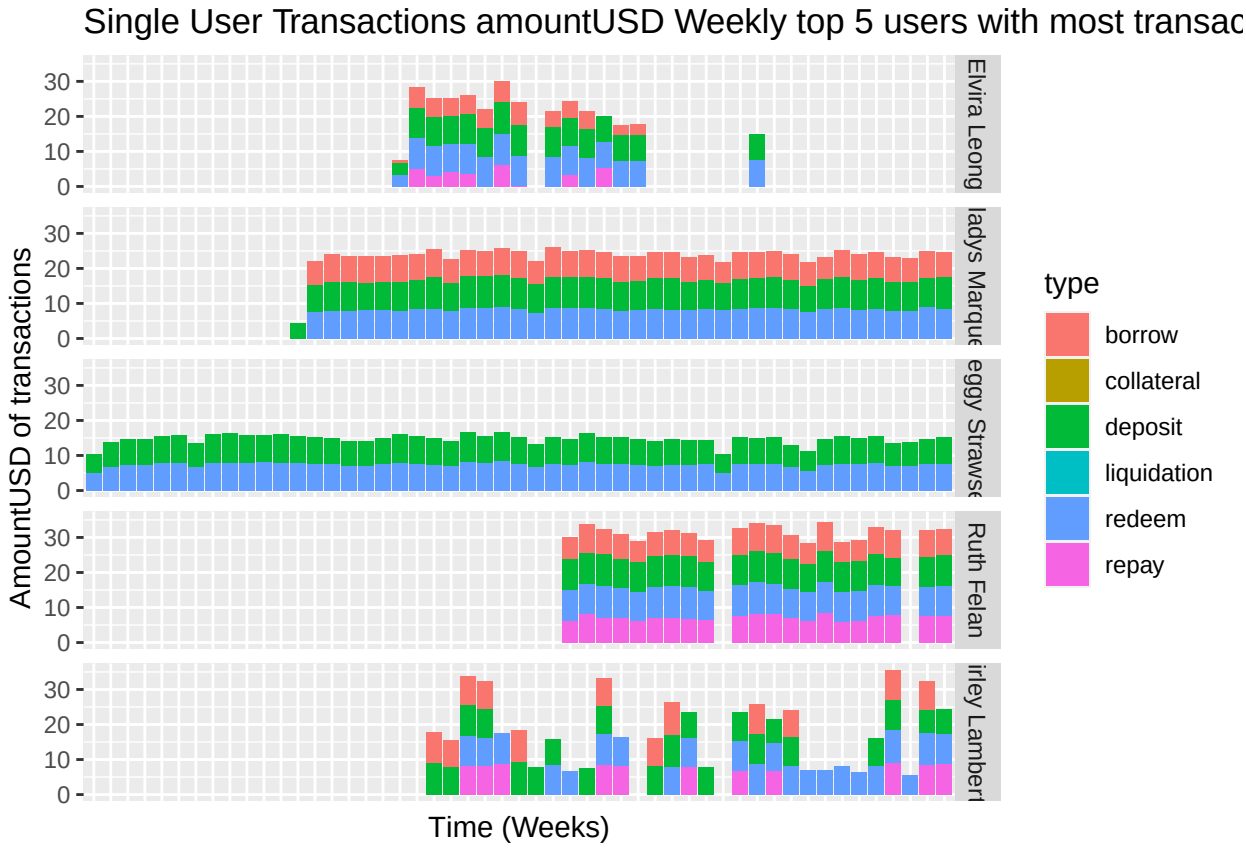
```
        #ggtitle("User Transaction History")
```

```
df.1users$ymd <- as_datetime(df.1users$timestamp) # fix times for the transactions

setDT(df.1users)[, ymd_new := format(as.Date(ymd), '%Y-%m-%V') ] ## '%Y-%m' for just month-year

df.1user <- df.1users %>% dplyr::count(ymd_new, type, user_alias, protocolContract, name = "amount")

ggplot(df.1user) +
  geom_bar(aes(x=ymd_new,y= amount, fill = type),
           stat='identity') + ggtitle("Single User Transactions Weekly for top 5 users with most transac
  theme(
           axis.text.x=element_blank(),
           axis.ticks.x=element_blank())+facet_grid(rows = df.1user$user_alias)
```




```
df.1user <- df.1users %>%
  dplyr::count(ymd_new, type, user_alias, wt=amountUSD, name = "amountUSD")

ggplot(df.1user) +
  geom_bar(aes(x=ymd_new,y= log10(amountUSD), fill = type),
           stat='identity') + ggtitle("Single User Transactions amountUSD Weekly top 5 users with most
  theme(
           axis.text.x=element_blank(),
           axis.ticks.x=element_blank())+facet_grid(rows = df.1user$user_alias)
```

```
## Warning: Removed 76 rows containing missing values (geom_bar).
```




```
df.1user <- df.1users %>% dplyr::count(user_alias, protocolContract, name = "amount")
df.1user
```

```
##           user_alias protocolContract amount
## 1:      Elvira Leong            False    424
## 2:    Gladys Marquez             True  59787
## 3:    Peggy Strawser            False  21503
## 4:        Ruth Felan            False    600
## 5: Shirley Lamberton            False    294
```

```
df.1user <- df.1users %>%
  dplyr::count(user_alias, wt=amountUSD, protocolContract, name = "amountUSD")
df.1user
```

```
##           user_alias protocolContract   amountUSD
## 1:      Elvira Leong            False  6420573071
## 2:    Gladys Marquez             True 27033268579
## 3:    Peggy Strawser            False  4581506747
## 4:        Ruth Felan            False 22223351085
## 5: Shirley Lamberton            False 22146712515
```

APPENDIX: Accessing RStudio Server on the IDEA Cluster

The IDEA Cluster provides five compute nodes (4x 48 cores, 1x 80 cores, 1x storage server)

- The Cluster requires RCS credentials, enabled via registration in class
  - email John Erickson for problems `erickj4@rpi.edu`
- RStudio, Jupyter, MATLAB, GPUs (on two nodes); lots of storage and computes

- Access via RPI physical network or VPN only

RStudio GUI Access for DAR:

- Access the RPI VPN
- Browse to: http://lp01.idea.rpi.edu/rstudio-ose/ (RStudio Server)
- Log in using your RCS username and password
  - If you cannot log in, contact John Erickson at erickj4@rpi.edu

Shared Data on Cluster:

- Users enrolled in DAR have access to `/academics/MATP-4910-F21`
  - Usually DAR users will see a symbolic ("soft") link in their home directories
  - If you do not, type the following in the **Terminal** via RStudio: `ln -s /academics/MATP-4910-F21/ MATP-4910-F21`

For advanced users:

- All idea_users have access to shared storage via `/data` ("data" in your home directories)
  - You might wish to use this for data sharing in team projects...
  - ...but we recommend using github for shared code development
- Shell access to nodes: You must access "landing pad" first, then compute node:
- `ssh your_rcs@lp01.idea.rpi.edu` For example: `ssh erickj4@lp01.idea.rpi.edu`
- Then, `ssh` to the desired compute node, e.g.: `ssh idea-node-02`