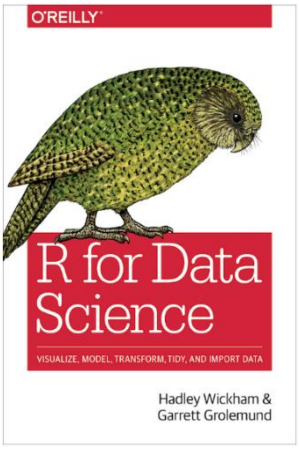
11. Týždeň

Data science a získavanie znalostí pomocou R



1) Jemný úvod do R

- **a) Čo je R?**
- **b) Vektory a data frame-y**
- **c) IO operácie**
- **d) Balíky a knižnice**
- **e) Grafika**

2) Databázové operácie a *tidyverse*

3) Pivot tabuľka polikliniky – 2

4) Balík rpivotTable

5) NASA - exoplanety

1) Jemný úvod do R

a) Čo je R?

```
* R: A Computer Language for Statistical Data Analysis
* Copyright (C) 1995, 1996 Robert Gentleman and Ross Ihaka
* Copyright (C) 1997--2019 The R Core Team
  https://svn.r-project.org/R/trunk/src/main/util.c
```

Populárnosť programovacích jazykov

R je interaktívny, vektorový jazyk s podporou funkcionálneho programovania. Jeho predchodcom bol jazyk S. Najprv sa inštaluje R a potom RStudio.

b) Vektory a data frame-y

Hodnoty do vektora k(c)ombinujeme pomocou funkcie c(...) [či lgl(...)]

```
z <- c(1, 2, NA, 4.1, 1/3); z;
## [1] 1.0000000 2.0000000       NA 4.1000000 0.3333333

length(z)  # 5
sort(z, decreasing = FALSE)
## [1] 0.3333333 1.0000000 2.0000000 4.1000000
```

Vektory môžeme vytvárať pomocou metódy sekvencie *sec*, *rep* alebo : operátora.

```
# from, to, by
seq(-2, 2, 0.4)    # <=> by=0.4
## [1] -2.0 -1.6 -1.2 -0.8 -0.4  0.0  0.4  0.8  1.2  1.6  2.0
# <=>
seq(-2, 2, len = 11)
- 2: 2/ 5            # -0.4 -0.2  0.0  0.2  0.4
-20:20/50           # -0.40 -0.38 -0.36 ... 0.38  0.40
1.1:5.6/5           # 0.22 0.42 0.62 0.82 1.02
```

```
rep(1:2, each=5)
## [1] 1 1 1 1 1 2 2 2 2 2
```

```
rep(1:5, each=2)
## [1] 1 1 2 2 3 3 4 4 5 5
```

```
rep(1:5, times=2)
## [1] 1 2 3 4 5 1 2 3 4 5
rep(1:5,each=2, len=5)
## [1] 1 1 2 2 3
```

Vektory z písmen dostaneme pomocou

```
letters
## [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p" "q"
## [18] "r" "s" "t" "u" "v" "w" "x" "y" "z"
```

```
LETTERS[1:3]
## [1] "A" "B" "C"
```

Vektory s pseudonáhodnými veličinami obdržíme vďaka *runif* či *rnorm*

```
# 2000 hodnot medzi 2 a 10.5:
set.seed(345)
x = runif(2000, 2, 10.5); x
##   [1]  3.838157  4.335494  5.314363  7.573788  5.704864
##        ...
## [996]  8.606280  7.030828  7.435523  3.318158  4.356125
```

```
# 2000 hodnot s m=2 a s=10:
set.seed(347); x = rnorm(2000, 2, 10);
mean(x); var(x)
## [1] 1.966481
## [1] 99.97764
```

Z vektorov ľahko vytvárame matice pomocou *dim* alebo *cbind*, *rbind*.

```
A = 1:12
dim(A) = c(3,4); A
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   10
## [2,]    2    5    8   11
## [3,]    3    6    9   12
```

```
rbind(A[1:3], 4*A[1:3])
```
```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    8   12
```

```
is.matrix(A)
```
```
## [1] TRUE
```

```
is.vector(A)
```
```
## [1] FALSE
```

Data frame – DB tabuľka

```
df1 <- data.frame(x = c(1, 2, 3), y = 7:5, z = letters[3:5]);  df1
```
```
##   x y z
## 1 1 7 c
## 2 2 6 d
## 3 3 5 e
```
```
class(df1)  # data.frame
```

c) IO operácie

```
write.csv(A, file = "maz.csv", row.names = FALSE)
AA = read.csv("maz.csv"); AA
```
```
##   V1 V2 V3 V4
## 1  1  4  7 10
## 2  2  5  8 11
## 3  3  6  9 12
```

Kde sa nachádza súbor *maz.csv* zistíme pomocou

```
getwd()
```
```
## [1] "C:/Users/Csaba/…/DBS1/___Cvic_R"
```

Pre data môžeme vytvoriť konkrétny adresár príkazom *setwd*("C:/Users/Data"). Ďalšie príkazy

```
dir(getwd())
R.home()                # domovsky adresar R
list.files(R.home())
```

Funkcia *read.table* slúži nielen na načítanie štrukturovaných údajov zo súboru, podobne ako aj *read.csv*, ale aj na vytvorenie *data.frame* z reťazca so štrukturou:

```
df1 <- read.table(text = "
```

```
a b x y
0 0 0 7
0 0 1 1
0 1 0 3
", header = TRUE);
df1
```

```
##   a b x y
## 1 0 0 0 7
## 2 0 0 1 1
## 3 0 1 0 3
```

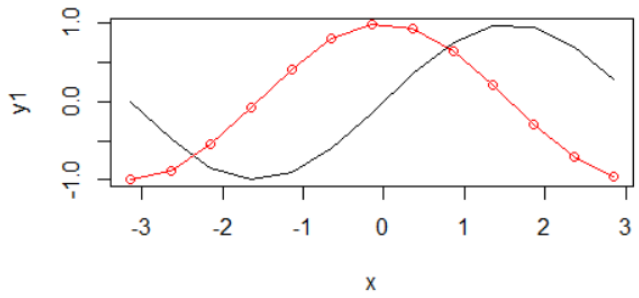
d) Balíky a knižnice

Systém R má viactisíc balíkov, knižníc (package, library). Každý balík treba raz nainštalovať a potom hocikedy načítať do pamäte. Napr.

```
if (!require(ggplot2, quietly = TRUE)) install.packages("ggplot2")
library(ggplot2)
```

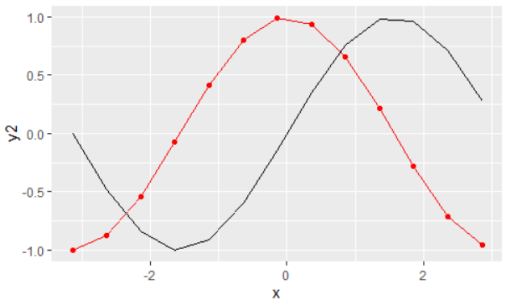
e) Grafika

```
x = seq(-pi,pi,.5)
y1=sin(x)
y2=cos(x)
plot(x,y1, type='l')
lines(x,y2, col='red')
points(x,y2, col='red')
```




```
# <=> Pomocou matplot s pouzitim add=TRUE
matplot(x, y1, type='l')
matplot(x, y2, type='l', col='red', add=T)
points (x, y2, col='blue')
matplot(x, y2, type='p', pch = 1, col='red', add=T)
```

```
# <=> az na nadpis y-ej osi
A = cbind(y1, y2, y2)
matplot(x, A, type = c('l', 'l', 'p'), lty=1, pch = 1, col = c('black', 'red', 'red'))
```



Pomocou balíka *ggplot2* môžeme vytvárať krajšie grafy.

```
xy12 = as.data.frame(cbind(x, y1,y2))
g = ggplot(data = xy12, aes(x = x, y = y2)); g
g = g + geom_line(col = 'red'); g
g = g + geom_point(col = 'red'); g     # scatterplot
g = g + geom_line(aes(x = x, y = y1)); g
```

Daný kód nie je efektívny, veď sa trikrát prekresluje obrázok. Jeho jedinou výhodou je, že môžeme postupne sledovať, čo sa urobí jednotlivými krokmi. Správny postup kódovania je zreťazenie/piping:

```
# <=> lepsie piping pomocou +
ggplot(xy12, aes(x = x, y = y2)) +
 geom_line(col = 'red') +
  geom_point(col = 'red') +
   geom_line(aes(x = x, y = y1))
```

2) Databázové operácie a *tidyverse*

Teraz ukážeme ako sa robí data science v R. Už vieme ako dostať dáta do R, resp. ako ich vytvárať a oboznámili sme sa aj s najužitočnejšími dátovými štruktúrami R. Tu uvidíme ako sa transformujú dáta v podobe data frame resp. tibble pomocou DB-ých operácií knižnice dplyr https://r4ds.had.co.nz/transform.html

SQL	-	**R, dplyr**
SELECT	-	*select()*
Vytvorenie nových premenných	-	*mutate()*
ORDER BY	-	*arrange()*
WHERE	-	*filter()*
GROUP BY a a*gregácia*	-	*group_by a summarise()*
Pivot tabuľky	-	*pivot_wider a pivot_longe*

V súčastnosti sa vykonaním kódových riadkov pre *tidyverse* https://www.tidyverse.org/blog/2019/09/tidyr-1-0-0/

```
if (!require(tidyverse, quietly = TRUE)) install.packages("tidyverse");
library(tidyverse)
```

nainštalujú sa a pripoja nasledujúce balíky

tibble, purrr, tidyr, dplyr, readr, stringr, forcats.

Medzi data frame a tibble sú tri rozdiely, pozri https://cran.r-project.org/web/packages/tibble/vignettes/tibble.html

Atomické typy sú

integer, double, character, logical, complex, raw.

Ak máme veľa stĺpcov s podobným názvom, pre vertikálnu filtráciu *select* sú užitočné nasl. funkcie

- *starts_with*(), *ends_with*(), *contains*()
- *matches*()
- *num_range*()
- *one_of*()
- *everything*()
- *group_cols*()

```
Editovať: kvôli obrázku napravo nesmie byť prázdny znak vnútri " " na konci riadku!
dfW_ <- read.table(text ="
aa bb age vv
Pepsi likes 10 6
```

```
Pepsi likes 15 8
Pepsi gets 5 2
Pepsi non 10 4
Coca likes 5 10
Coca non 10 8
Coca likes 15 6
Coca gets 5 8
Coca gets 10 6
", header = TRUE);
dfW <- as_tibble(dfW_)

m1 = mutate     ( dfW,  ha   = 2*vv,  haha = 4*vv );  m1
m2 = transmute( dfW,  ha   = 2*vv,  haha = 4*vv );  m2
# <=>
select(m1, contains("ha"))
# <=>
select(m1, starts_with("ha"))
```

	aa	bb	age	vv
1	Pepsi	likes	10	6
2	Pepsi	likes	15	8
3	Pepsi	gets	5	2
4	Pepsi	non	10	4
5	Coca	likes	5	10
6	Coca	non	10	8
7	Coca	likes	15	6
8	Coca	gets	5	8
9	Coca	gets	10	6

dfW

	aa	bb	age	vv	ha	haha
1	Pepsi	likes	10	6	12	24
2	Pepsi	likes	15	8	16	32
3	Pepsi	gets	5	2	4	8
4	Pepsi	non	10	4	8	16
5	Coca	likes	5	10	20	40
6	Coca	non	10	8	16	32
7	Coca	likes	15	6	12	24
8	Coca	gets	5	8	16	32
9	Coca	gets	10	6	12	24

m1

	ha	haha
1	12	24
2	16	32
3	4	8
4	8	16
5	20	40
6	16	32
7	12	24
8	16	32
9	12	24

m2

```
s1 = select( arrange(dfW, age, bb, aa), age, bb, aa); s1
s2 = select(dfW, -c(age:vv));   s2
s3 = select(dfW, age, everything());   s3
```

	age	bb	aa
1	5	gets	Coca
2	5	gets	Pepsi
3	5	likes	Coca
4	10	gets	Coca
5	10	likes	Pepsi
6	10	non	Coca
7	10	non	Pepsi
8	15	likes	Coca
9	15	likes	Pepsi

s1

	aa	bb
1	Pepsi	likes
2	Pepsi	likes
3	Pepsi	gets
4	Pepsi	non
5	Coca	likes
6	Coca	non
7	Coca	likes
8	Coca	gets
9	Coca	gets

s2

	age	aa	bb	vv
1	10	Pepsi	likes	6
2	15	Pepsi	likes	8
3	5	Pepsi	gets	2
4	10	Pepsi	non	4
5	5	Coca	likes	10
6	10	Coca	non	8
7	15	Coca	likes	6
8	5	Coca	gets	8
9	10	Coca	gets	6

s3

```
if (!require(gridExtra, quietly = TRUE)) install.packages("gridExtra");
library(gridExtra);

grid.arrange(tableGrob(dfW), tableGrob(m1), tableGrob(m2), newpage = TRUE,
             tableGrob(s1), tableGrob(s2), tableGrob(s3), nrow = 2, ncol = 3);
```

Výpočtovo náročnú metódu *grid.arrange* z balíka *gridExtra* používame tu iba kvôli úspornému , prehľadnému zobrazeniu viac dát. štruktúr.

```
a1 = arrange(dfW, age, bb, aa);  a1
a2 = select( a1, age, bb, aa);  a2

f1 = filter(dfW, vv<7);  f1
f2 = filter(dfW, vv<7 & age>=10);  f2
```

	aa	bb	age	vv
1	Cocao	gets	5	8
2	Pepsi	gets	5	2
3	Cocao	likes	5	10
4	Cocao	gets	10	6
5	Pepsi	likes	10	6
6	Cocao	non	10	8
7	Pepsi	non	10	4
8	Cocao	likes	15	6
9	Pepsi	likes	15	8

	age	bb	aa
1	5	gets	Cocao
2	5	gets	Pepsi
3	5	likes	Cocao
4	10	gets	Cocao
5	10	likes	Pepsi
6	10	non	Cocao
7	10	non	Pepsi
8	15	likes	Cocao
9	15	likes	Pepsi

	aa	bb	age	vv
1	Pepsi	likes	10	6
2	Pepsi	gets	5	2
3	Pepsi	non	10	4
4	Cocao	likes	15	6
5	Cocao	gets	10	6

	aa	bb	age	vv
1	Pepsi	likes	10	6
2	Pepsi	gets	5	2
3	Pepsi	non	10	4
4	Cocao	likes	15	6
5	Cocao	gets	10	6

```
grid.arrange(tableGrob(a1), tableGrob(a2),newpage = TRUE,
             tableGrob(f1), tableGrob(f1), nrow = 2, ncol = 2)
```

group_by a summarise

Zoskupenie nemení vzhľad údajov!

```
# zoskupenie nemení vzhľad údajov
 gg1 <- group_by(dfW, age, aa);   gg1
 gg2 <- group_by(dfW, age, bb);  gg2
```

	aa	bb	age	vv
1	Pepsi	likes	10	6
2	Pepsi	likes	15	8
3	Pepsi	gets	5	2
4	Pepsi	non	10	4
5	Cocao	likes	5	10
6	Cocao	non	10	8
7	Cocao	likes	15	6
8	Cocao	gets	5	8
9	Cocao	gets	10	6

	aa	bb	age	vv
1	Pepsi	likes	10	6
2	Pepsi	likes	15	8
3	Pepsi	gets	5	2
4	Pepsi	non	10	4
5	Cocao	likes	5	10
6	Cocao	non	10	8
7	Cocao	likes	15	6
8	Cocao	gets	5	8
9	Cocao	gets	10	6

```
writeLines("Groups: age, bb, vv [6]\n !!! vs \nGroups: bb, vv [6]")
grid.arrange(tableGrob(gg1), tableGrob(gg2), ncol = 2)
```

```
 s1 = summarise(gg1); s1  # kombinacie
 s2 = summarise(gg2); s2
 ### !!!!! V summarise 1. arg. je group_by a dalsie su agr.funkcie!
 s3 = summarise(gg1, sum = sum(vv, na.rm = TRUE)); s3
 s4 = summarise(gg2, pct = sum(!is.na(vv) )); s4
 s5 = summarise(gg2, sum = sum(vv, na.rm = TRUE)); s5
```

```
grid.arrange(tableGrob(gg2), tableGrob(s1), tableGrob(s2), newpage = TRUE,
             tableGrob(s3), tableGrob(s4), tableGrob(s5), nrow = 2, ncol = 3)
```

	aa	bb	age	vv
1	Pepsi	likes	10	6
2	Pepsi	likes	15	8
3	Pepsi	gets	5	2
4	Pepsi	non	10	4
5	Cocao	likes	5	10
6	Cocao	non	10	8
7	Cocao	likes	15	6
8	Cocao	gets	5	8
9	Cocao	gets	10	6

	age	aa
1	5	Cocao
2	5	Pepsi
3	10	Cocao
4	10	Pepsi
5	15	Cocao
6	15	Pepsi

	age	bb
1	5	gets
2	5	likes
3	10	gets
4	10	likes
5	10	non
6	15	likes

	age	aa	sum
1	5	Cocao	18
2	5	Pepsi	2
3	10	Cocao	14
4	10	Pepsi	10
5	15	Cocao	6
6	15	Pepsi	8

	age	bb	pct
1	5	gets	2
2	5	likes	1
3	10	gets	1
4	10	likes	1
5	10	non	2
6	15	likes	2

	age	bb	sum
1	5	gets	10
2	5	likes	10
3	10	gets	6
4	10	likes	6
5	10	non	12
6	15	likes	14

Inverzné operácie pivot_wider a pivot_longer

V pivot_wider chýbajúce stĺpce tvoria prvé stĺpce vo výsledku. Kým
- *names_from* určuje stĺpce pomocou c( ), z diskrétnych hodnôt ktorých sa vytvárajú výstupné stĺpce,
- *values_from* určuje stĺpce, z ktorých sa výpočítavajú (agregačné) hodnoty pre bunky,
- *values_fn* () – agregačná funkcia:

```
* Use `values_fn = list(Obrat = list)`         to suppress this warning.
* Use `values_fn = list(Obrat = length)`       to identify where the duplicates arise
* Use `values_fn = list(Obrat = summary_fun)` to summarise duplicates
```

V pivot_longer
- cols – columns to pivot into longer format.
- names_to – the column to create from the data stored in the columns.
- values_to – the column to create from the data stored in cell values.

```
names(dfW)
## "aa"  "bb"  "age" "vv"
```

```
pt1 = dfW %>%
          tidyr::pivot_wider(names_from = bb,          # aa je 1. stlpec
                             values_from = vv)
pt2 = dfW %>% pivot_wider(names_from = aa,           # bb je 1. stlpec
                             values_from = vv)
pt3 = dfW %>% pivot_wider(names_from = c(bb,aa),     # age je 1. stlpec
                             values_from = vv)
### <=>
pt4 = dfW %>% pivot_wider(names_from = c(aa,bb),
                             values_from = vv)
pt5 = dfW %>% pivot_wider(names_from = c(bb,age),    # aa je 1. stlpec
                             values_from = vv)
pt6 = dfW %>% pivot_wider(names_from = c(aa,age),    # bb je 1. stlpec
                             values_from = vv)
grid.arrange(arrangeGrob(tableGrob(pt1),
                         tableGrob(pt2),ncol = 2),
             tableGrob(pt3), tableGrob(pt5), tableGrob(pt6), nrow = 4)
```

	aa	age	likes	gets	non
1	Pepsi	10	6	NA	4
2	Pepsi	15	8	NA	NA
3	Pepsi	5	NA	2	NA
4	Cocao	5	10	8	NA
5	Cocao	10	NA	6	8
6	Cocao	15	6	NA	NA

	bb	age	Pepsi	Cocao
1	likes	10	6	NA
2	likes	15	8	6
3	gets	5	2	8
4	non	10	4	8
5	likes	5	NA	10
6	gets	10	NA	6

	age	likes_Pepsi	gets_Pepsi	non_Pepsi	likes_Cocao	non_Cocao	gets_Cocao
1	10	6	NA	4	NA	8	6
2	15	8	NA	NA	6	NA	NA
3	5	NA	2	NA	10	NA	8

	aa	likes_10	likes_15	gets_5	non_10	likes_5	gets_10
1	Pepsi	6	8	2	4	NA	NA
2	Cocao	NA	6	8	8	10	6

	bb	Pepsi_10	Pepsi_15	Pepsi_5	Cocao_5	Cocao_10	Cocao_15
1	likes	6	8	NA	10	NA	6
2	gets	NA	NA	2	8	6	NA
3	non	4	NA	NA	NA	8	NA

```
L0 <- read.table(text = "
mesto kvart xx
A    Q2    4
C    Q1    3
D    Q3    2
A    Q1    5
B    Q2    7
", header = TRUE); print(L0)
L0 = as_tibble(L0); print(L0)
### Wider
LG = group_by(L0, mesto, kvart); #print(dLG)
LGW = pivot_wider(LG, names_from = kvart, values_from = xx); #print(dLGI)
### Longer
LGWI1 = pivot_longer(LGW, cols = starts_with("Q")); print(LGWI1)
LGWI2 = pivot_longer(LGW, cols = starts_with("Q"),values_drop_na = TRUE
                                , names_to = "KVART",values_to = "XX"  ); print(LGWI2)
###
### TEST setdiff
###
names(LGWI2) = names(L0);
jaj = setdiff(L0,LGWI2); print(jaj)

L0; LG; LGW; LGWI1; LGWI2
grid.arrange(tableGrob(arrange(L0, mesto)),
        tableGrob(arrange(LGWI1, mesto)),
        tableGrob(arrange(LGWI2, mesto)), ncol = 3);
```

	mesto	kvart	xx
1	A	Q2	4
2	A	Q1	5
3	B	Q2	7
4	C	Q1	3
5	D	Q3	2

L0

	mesto	name	value
1	A	Q2	4
2	A	Q1	5
3	A	Q3	NA
4	B	Q2	7
5	B	Q1	NA
6	B	Q3	NA
7	C	Q2	NA
8	C	Q1	3
9	C	Q3	NA
10	D	Q2	NA
11	D	Q1	NA
12	D	Q3	2

LGVI1

	mesto	kvart	xx
1	A	Q2	4
2	A	Q1	5
3	B	Q2	7
4	C	Q1	3
5	D	Q3	2

LGWI1 = L0

3) Pivot tabuľka polikliniky – 2

Zistite počet pacientov u jednotlivých lekárov v jednotlivých mesiacoch!
Po ozrejmení práce s metódami *pivot_wider* a *pivot_longer* vrátime sa k pivotovaniu DB polikliniky a uvedieme dve ekvivalentné riešenia DF2 a DF3 (ekv. riešenie DF1 sme vytvorili na predch. prednáške).

```
library(RODBC)
library(tidyverse)
# conn <- odbcDriverConnect(conSS);              # SQL Server  conSS - pozri predchád. prednášku
conn <- odbcConnect('mysql_poliklinika');        # MySQL
qq = "select krstne, spec, month(den) mes, poplatok
   from lekari L join navstevy N ON L.idL=N.idL;"
df2 <- sqlQuery(conn, qq); df2
close(conn)

if (!require(data.table, quietly = TRUE))install.packages("data.table")
library(data.table)

DF2 = reshape2::dcast(df2, mes ~ krstne,
         value.var = c("poplatok"), fun=length)
#         value.var = c("poplatok"), fun=sum)

df3 = df2 %>%
  #dplyr::filter(!is.na(result)) %>%
  dplyr::count(mes, krstne) %>%
    tidyr::pivot_wider(
      names_from = krstne
      ,values_from = n
      ,values_fill = list(n = 0) # un/comment
   # ,values_fn = list(y = sum)
    )
#df3[,order(colnames(df3),decreasing = FALSE)]
juj = c(1,1+order(colnames(df3)[2:length(colnames(df3))], decreasing = FALSE))
DF3 <- df3[,juj];
DF2; DF3; # DF1;
grid.arrange(arrangeGrob(tableGrob(DF1), tableGrob(DF2),
             tableGrob(DF3),ncol = 3), nrow = 1)
```

```
# Sumarny poplatok:
df2 %>% select(mes,krstne,poplatok) %>%
        tidyr::pivot_wider(
          names_from = krstne
          ,values_from = poplatok
          ,values_fn = sum   # length
        ) %>% arrange(.,mes)
```

	mes	Imro	Klara	Oto	Zoli	Zuzka
1	5	0	2	0	1	0
2	6	0	0	3	0	1
3	7	0	2	1	0	0
4	8	0	0	0	2	1
5	9	1	0	3	0	0
6	10	1	0	1	0	1
7	11	1	0	1	0	0

	mes	Imro	Klara	Oto	Zoli	Zuzka
1	5	0	2	0	1	0
2	6	0	0	3	0	1
3	7	0	2	1	0	0
4	8	0	0	0	2	1
5	9	1	0	3	0	0
6	10	1	0	1	0	1
7	11	1	0	1	0	0

	mes	Imro	Klara	Oto	Zoli	Zuzka
1	5	0	2	0	1	0
2	6	0	0	3	0	1
3	7	0	2	1	0	0
4	8	0	0	0	2	1
5	9	1	0	3	0	0
6	10	1	0	1	0	1
7	11	1	0	1	0	0

Pred záverom o možnostiach R v oblasti pivotovania ilustrujeme ešte **množinové** a **join** operácie v jazyku R.

Množinové operácie

union, intersect, setdiff

```
tb3 <- tibble(x = 1:2, y = c(1L, 1L)); tb3
tb4 <- tibble(x = 1:2, y = 1:2);       tb4
se2 = union(tb3, tb4)
se1 = intersect(tb3, tb4)
se3 = setdiff(tb3, tb4)
grid.arrange(arrangeGrob(tableGrob(tb3),
                         tableGrob(tb4),ncol = 2),
             arrangeGrob(tableGrob(se1), tableGrob(se2),
                         tableGrob(se3),ncol=3), nrow = 2)
```

	x	y
1	1	1
2	2	1

	x	y
1	1	1
2	2	2

	x	y
1	1	1

	x	y
1	1	1
2	2	1
3	2	2

	x	y
1	2	1

JOIN

inner_join vs left_join [semi_join, nest_join, anti_join]

```
tb1 <- tibble(x = c(1, 2, 3), y = 2:0);          tb1
tb2 <- tibble(x = c(1, 3), a = 10, b = "a");  tb2

# Vsetko z tb1:
j1 = tb1 %>% left_join(tb2); j1

# Iba zhodne x, y:
j2 = tb1 %>% inner_join(tb2); j2
j2 %>% knitr::kable()

j3 = setdiff(j1, j2)
dim(j3)
grid.arrange(arrangeGrob(tableGrob(tb1),
                         tableGrob(tb2),ncol = 2),
             arrangeGrob(tableGrob(j1), tableGrob(j2),
                         tableGrob(j3),ncol=3), nrow = 2)
```

	x	y
1	1	2
2	2	1
3	3	0

	x	a	b
1	1	10	a
2	3	10	a

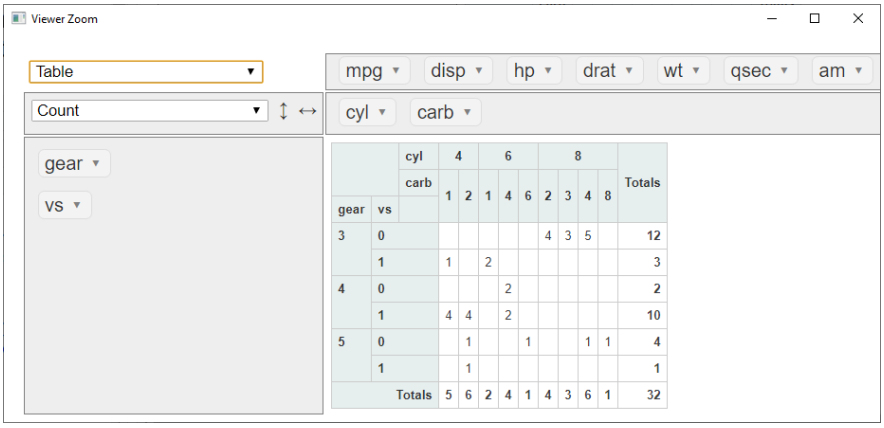
	x	y	a	b
1	1	2	10	a
2	2	1	NA	NA
3	3	0	10	a

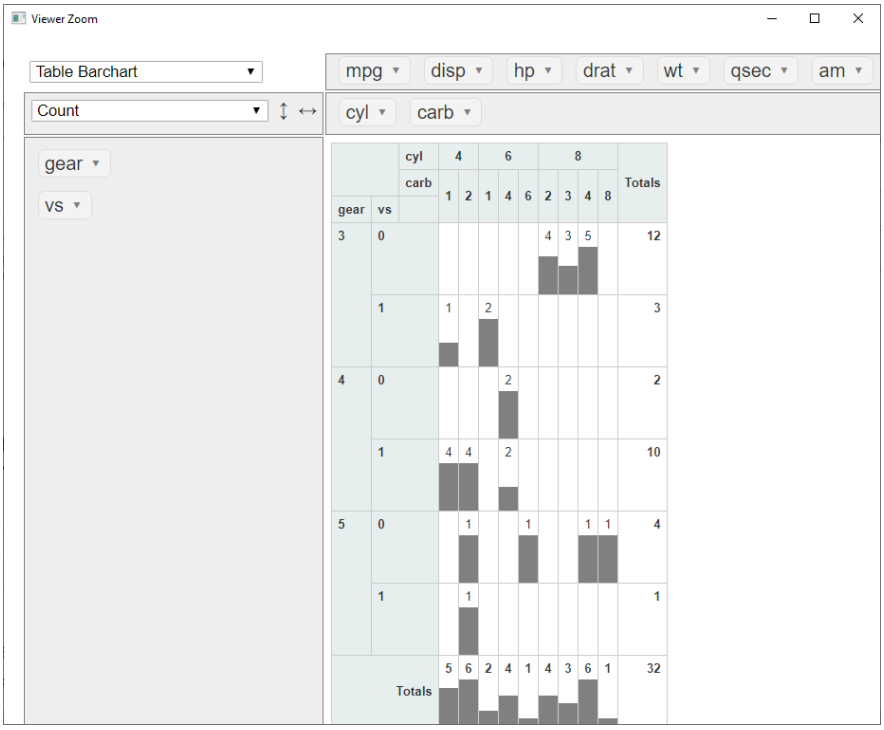
	x	y	a	b
1	1	2	10	a
2	3	0	10	a

	x	y	a	b
1	1	2	10	a
2	3	0	10	a

4) Balík rpivotTable https://cran.r-project.org/web/packages/rpivotTable/vignettes/rpivotTableIntroduction.html

```
if (!require(htmlwidgets, quietly = TRUE)) install.packages("htmlwidgets")
library(htmlwidgets)
if (!require(rpivotTable, quietly = TRUE)) install.packages("rpivotTable")
library(rpivotTable)

data(mtcars)
rpivotTable(mtcars, rows=c("gear","vs"),
                    cols=c("cyl","carb"),
                    width="100%", height="400px")
```





V záverečnej prednáške o PT ukážeme, že v MS Excel okrem rovnakého interaktívneho pivotovania môžeme dosiahnuť rôzne 3D zobrazenia.

5) Exoplanety – NASA

Dáta o exoplanetach, ktoré obiehajú nie okolo Slnka ale iných hviezd, načítame zo servera a vybrané pôvodné dlhé stĺpce premenujeme.

```
dri <- "SQL Server"
ser <- "DESKTOP-DT61NCP"
DB <- "NASA"
conSS <- paste0( "Driver=", dri,
                 "; Server=", ser,
                 "; Database=", DB)
```

Pozri bod 2) časti II) Pomocou balíka RODBC poslednej prednášky.
2) ... spustit program (Ctrl + Esc) ODBC Data Sources 64 bit
Kliknúť na Add
Zvoliť MySQL ODBC 8.0 Unicode Driver
Vyplniť
Data Source Name: mysql_NASA
TCP/IP Server: localhost Port: 3306
User: root
Password: Pozor HESLO MySQL !
Database: nasa_exoplanets
Test => OK

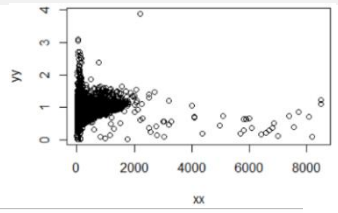
```
library(tidyverse)
#conn <- odbcConnect('mysql_NASA');  # MySQL
#qq = "select * from Exoplanets;"
#df2_ <- sqlQuery(conn, qq); df2_;      #close(conn)
folder="C://Csaba//__________Vyuka//_________Vyuka_2019//DBS1_ZS//___PDF//_NASA_Exoplanets//"
meno="NASA_Exoplanets_OK.csv"
df2_= read.table(paste(folder, meno, sep=""), header = TRUE, sep = ","); df2_

is.data.frame(df2_)
names(df2_);
head(df2_[1,13:14])
```

Výber šesť stĺpcov:

```
library(tidyverse)
df2 = transmute(df2_, dist = st_distance, mass = st_mass,
                      type = st_spectral_type, temp = st_effective_temp,
                      radi = st_solar_radii, disc = pl_discovery_method );
names(df2)
```

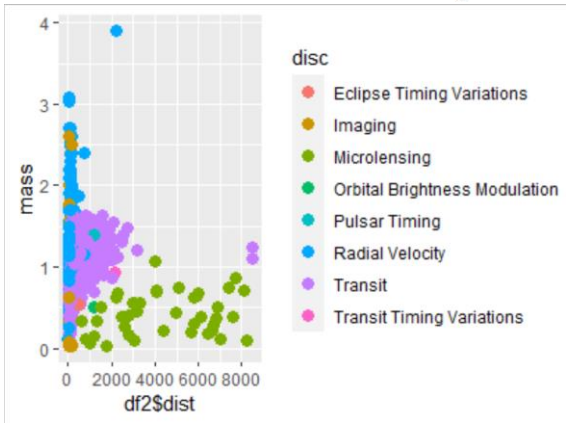
Nakreslíme bodový graf dvojice hodnôt zo stĺpcov *dist, mass*.

```
plot(df2$dist, df2$mass)
#plot(df2$dist, df2$mass, col=df2$dist)
```



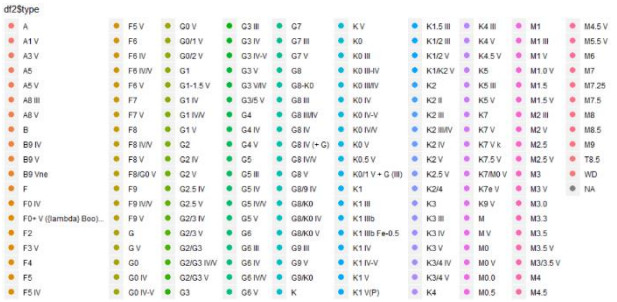
Nakreslíme bodový graf pre *dist, mass* farebne v závislosti od spôsobu objavenia *disc*.

```
library(ggplot2)
# resp. dist namiesto df2$dist:
ggplot(df2, aes(x=df2$dist, y=mass, color=disc)) +
 geom_point(size=3)
```



Nakreslíme bodový graf pre *dist, mass* farebne v závislosti od deviatich najpočetnejších *type* hodnôt.

```
### Naslepo - prilis vela typov
ggplot(df2, aes(x=df2$dist, y=df2$mass, color=df2$type)) +
  geom_point(size=3)
```

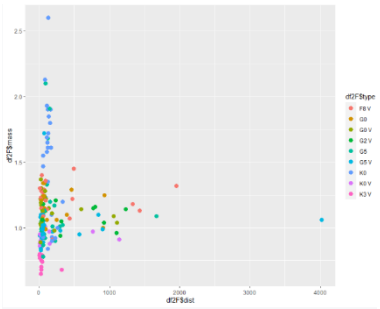


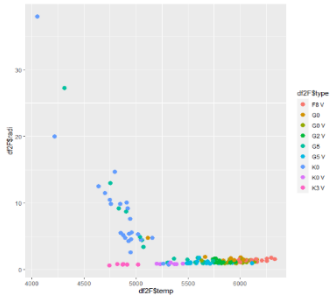
```
gg <- group_by(df2, type);   #gg
s3 = summarise( gg ); s3
s3p= summarise( gg, pct = sum(!is.na(type)) );
s3p[1:15,]
### typ10  = c("A", "B", "F", "G", "K", "M");  # - cvic.
#typ10 = c("G5 V","K0","G0 V","G5","F8 V","K3 V","G2 V","G0","K0 V");
### - takto automatizovat: !!!!!
typ10  = as.vector( t(arrange(s3p,desc(pct))[2:10,1]) );
typ10

df2F = filter(df2, type %in% typ10)
ggF <- group_by(df2F, type);   #ggF
s3pF= summarise(ggF, pct = sum(!is.na(type) ));
s3pF
ggplot(df2F, aes(x=df2F$dist, y=mass, color=type))  +  geom_point(size=3)
```

Nižšie zistené dva typy závislostí, pozri druhý obrázok (modré, zelené a ostatné), je možné ďalej skúmať:

```
ggplot(df2F, aes(x=df2F$temp, y=radi, color=type))  +   geom_point(size=3)
```

```
> s3pF
# A tibble: 9 x 2
  type    pct
  <chr> <int>
1 F8 V     29
2 G0       23
3 G0 V     34
4 G2 V     26
5 G5       32
6 G5 V     52
7 K0       40
8 K0 V     23
9 K3 V     28
```





Cvičenie. Zistite v R pomocou *pivot_wider* sumárne poplatky pre každý poplatok z daných troch poplatkov {200, 500, 800} a pre každého špecialistu na základe importovaných tabuliek Lekari a Navstevy (pozri pr. 5b v T_08b_VnoreneDopyty_Rollup.pdf).

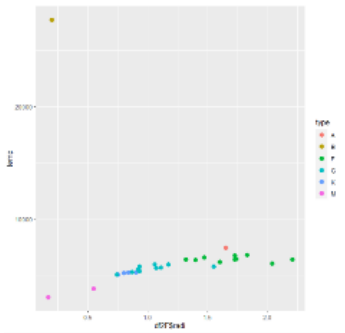
	Spec	poplatok	suma
1	Ocny	200	1000
2	Ocny	500	500
3	Zubny	500	1000
4	Zubny	800	1600

Výsledok

```
# A tibble: 2 x 4
  spec   `200` `500` `800`
  <fct>  <int> <int> <int>
1 Ocny    1000   500    NA
2 Zubny     NA  1000  1600
```

Cvičenie:

Nakreslite bodový graf pre dvojicu stĺpcov *radi, temp* farebne v závislosti od šiestich hodnôt A,B,F,G,K,M stĺpca *type*.



```
### Pivot tabulka DF3
library(RODBC)
library(tidyverse)
conn <- odbcConnect('mysql_poliklinika');    # MySQL
qq = "select krstne, spec, month(den) mes, poplatok
   from lekari L join navstevy N ON L.idL=N.idL;"
df2 <- sqlQuery(conn, qq); df2
close(conn)

if (!require(data.table, quietly = TRUE))install.packages("data.table")
library(data.table)
df3 = df2 %>%
  #dplyr::filter(!is.na(result)) %>%
  dplyr::count(mes, krstne) %>%
  tidyr::pivot_wider(
    names_from = krstne
    ,values_from = n
    ,values_fill = list(n = 0) # un/comment
    # ,values_fn = list(y = sum)
  )
#df3[,order(colnames(df3),decreasing = FALSE)]
juj = c(1,1+order(colnames(df3)[2:length(colnames(df3))], decreasing = FALSE))
DF3 <- df3[,juj];
DF3; # DF1;

### Kodove riadky pre Exoplanety – NASA
library(tidyverse)
#conn <- odbcConnect('mysql_NASA');         # MySQL
#qq = "select * from Exoplanets;"
#df2_ <- sqlQuery(conn, qq); df2_;       #close(conn)
folder="C://Csaba//___Vyuka//___Vyuka_2021-
2022//_____%__Vyuka_2021_ZS//DBS1//_____DWH_DataWarehouse_OLAP_Pivot//NASA_Exoplanets//"
meno="NASA_Exoplanets_OK.csv"
df2_= read.table(paste(folder, meno, sep=""), header = TRUE, sep = ","); df2_

is.data.frame(df2_)
names(df2_);
head(df2_[1,13:14])

library(tidyverse)
df2 = transmute(df2_, dist = st_distance, mass = st_mass,
            type = st_spectral_type, temp = st_effective_temp,
            radi = st_solar_radii, disc = pl_discovery_method );
names(df2)

plot(df2$dist, df2$mass)
#plot(df2$dist, df2$mass, col=df2$dist)

library(ggplot2)
```

```
# resp. dist namiesto df2$dist
ggplot(df2, aes(x=df2$dist, y=mass, color=disc)) +
  geom_point(size=3)

### Naslepo - prilis vela typov
#ggplot(df2, aes(x=df2$dist, y=df2$mass, color=df2$type)) + geom_point(size=3)

gg <- group_by(df2, type);   #gg
s3 = summarise( gg ); s3
s3p= summarise( gg, pct = sum(!is.na(type)) );
s3p[1:15,]
#typ10  = c("A", "B", "F", "G", "K", "M");
#typ10 = c("G5 V","K0","G0 V","G5","F8 V","K3 V","G2 V","G0","K0 V");
### - takto automatizovat: !!!!!
typ10  = as.vector( t(arrange(s3p,desc(pct))[2:10,1]) );
typ10

df2F = filter(df2, type %in% typ10)
ggF <- group_by(df2F, type);   #ggF
s3pF= summarise(ggF, pct = sum(!is.na(type) ));
s3pF

ggplot(df2F, aes(x=df2F$dist, y=mass, color=type)) +
  geom_point(size=3)

ggplot(df2F, aes(x=df2F$temp, y=radi, color=type)) +
  geom_point(size=3)
```