Relational Data

Uma vez feito todas as transformações e manipulações nos dados, é o momento de unir todas as fontes de dados que estão relacionados.

```
base_1 <- read_delim("~/Documentos/base_1.csv", delim=",")
```

```
## Rows: 15 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): cpf_completo, logradouro, numero
## dbl (2): lat, long
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
glimpse(base_1)
```

```
## Rows: 15
## Columns: 5
## $ cpf_completo <chr> "00000036832", "00000675156", "01238495725", "00074960362~
## $ logradouro   <chr> "Rua Joaquim Nabuco", "Rua Visconde de Inhauma", "Rua San~
## $ numero       <chr> " 123", " 456", " 101", " 202", " 303", " 404", " 505", "~
## $ lat          <dbl> -22.906, -22.986, -22.866, -22.916, -22.926, -22.936, -22~
## $ long         <dbl> -43.1729, -43.1249, -43.1359, -43.1569, -43.1679, -43.178~
```

```
base_2 <- read_delim("~/Documentos/base_2.csv", delim=",")
```

```
## Rows: 8 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (2): cpf_completo, sexo
## date (1): data_nascimento
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
glimpse(base_2)
```

```
## Rows: 8
## Columns: 3
## $ cpf_completo    <chr> "00000036832", "74628391548", "01238495725", "00074960~
## $ sexo            <chr> "F", "F", "F", "O", "M", "F", "M", "O"
## $ data_nascimento <date> 1982-01-19, 1993-08-31, 1979-02-28, 1999-10-01, 2009-0~
```

duplicated

Antes de unir bases, é importante verificar se existem duplicidades para evitar cartesiano.

```
qtde_dup1 <- sum(duplicated(base_1))
qtde_dup2 <- sum(duplicated(base_2))

glue("Qtde de duplicações base 1: {qtde_dup1}")
```

```
## Qtde de duplicações base 1: 6
```

```
glue("Qtde de duplicações base 2: {qtde_dup2}")
```

```
## Qtde de duplicações base 2: 0
```

distinct

Outra alternativa é olhar a quantidade de registros distintos.

```
qtde1      <- nrow(base_1)
qtde1_dist <- nrow(distinct(base_1))

qtde2      <- nrow(base_2)
qtde2_dist <- nrow(distinct(base_2))

glue("Qtde de registros base 1: {qtde1}")
```

```
## Qtde de registros base 1: 15
```

```
glue("Qtde de registros distintos base 1: {qtde1_dist}")
```

```
## Qtde de registros distintos base 1: 9
```

```
glue("Qtde de registros base 2: {qtde2}")
```

```
## Qtde de registros base 2: 8
```

```
glue("Qtde de registros distintos base 2: {qtde2_dist}")
```

```
## Qtde de registros distintos base 2: 8
```

Podemos verificar distintos na linha inteira (sem passar parâmetros na função), ou podemos nos basear em um campo.

```
distintos <- base_1 %>% distinct(cpf_completo)
distintos
```

```
## # A tibble: 9 x 1
##   cpf_completo
##   <chr>
## 1 00000036832
## 2 00000675156
## 3 01238495725
## 4 00074960362
## 5 00000648038
## 6 05276981448
## 7 00687925356
## 8 84796132025
## 9 00000946132
```

No entanto as demais colunas são eliminadas, é possível preserva-las com o parâmetro .keep_all

```
distintos <- base_1 %>% distinct(cpf_completo, .keep_all = T)
distintos
```

```
## # A tibble: 9 x 5
##   cpf_completo logradouro                    numero   lat  long
##   <chr>        <chr>                         <chr>  <dbl> <dbl>
## 1 00000036832  Rua Joaquim Nabuco            " 123" -22.9 -43.2
## 2 00000675156  Rua Visconde de Inhauma       " 456" -23.0 -43.1
## 3 01238495725  Rua Santa Luzia               " 101" -22.9 -43.1
## 4 00074960362  Rua Senador Dantas            " 202" -22.9 -43.2
## 5 00000648038  Rua Marechal Deodoro          " 303" -22.9 -43.2
## 6 05276981448  Rua Conde de Bonfim           " 404" -22.9 -43.2
## 7 00687925356  Rua Nossa Senhora de Copacabana " 505" -22.9 -43.2
## 8 84796132025  Rua Santa Clara               " 606" -23.0 -43.2
## 9 00000946132  Rua Dona Mariana              " 707" -23.0 -43.2
```

inner join

Feito a tratativa nas duplicações é hora de juntar os dados. O inner join junta apenas os dados que existirem em ambas bases.

```
join1 <- inner_join(base_2,
                    distintos,
                    by = "cpf_completo")

join1
```

```
## # A tibble: 7 x 7
##   cpf_completo sexo  data_nascimento logradouro                numero   lat  long
##   <chr>        <chr> <date>          <chr>                     <chr>  <dbl> <dbl>
## 1 00000036832  F     1982-01-19      Rua Joaquim Nabuco        " 123" -22.9 -43.2
## 2 01238495725  F     1979-02-28      Rua Santa Luzia           " 101" -22.9 -43.1
## 3 00074960362  O     1999-10-01      Rua Senador Dantas        " 202" -22.9 -43.2
## 4 00000648038  M     2009-06-14      Rua Marechal Deodoro      " 303" -22.9 -43.2
## 5 00687925356  F     1982-03-18      Rua Nossa Senhora de Co~ " 505" -22.9 -43.2
## 6 84796132025  M     1991-09-22      Rua Santa Clara           " 606" -23.0 -43.2
## 7 00000946132  O     1977-11-02      Rua Dona Mariana          " 707" -23.0 -43.2
```

right, left, full join

No entanto dependendo da necessidade queremos que todos dados de uma base seja preservado mesmo que não haja um registro correspondente na segunda base.

No right ou left estamos escolhendo qual tabela vamos preservar os dados, é mais comum utilizar o left.

No exemplo para ficar claro, vamos manter todos os dados da base distintos e entender onde a tabela fica em cada uma das funções.

```
qtde_linhas_dist  <- nrow(distintos)
qtde_linhas_base2 <- nrow(base_2)

glue("Distintos: {qtde_linhas_dist} linhas")
```

```
## Distintos: 9 linhas
```

```
glue("Base_2: {qtde_linhas_base2} linhas")
```

```
## Base_2: 8 linhas
```

```
join2 <- left_join(distintos,
                   base_2,
                   by= "cpf_completo")
nrow(join2)
```

```
## [1] 9
```

```
join3 <- right_join(base_2,
                    distintos,
                    by= "cpf_completo")
nrow(join3)
```

```
## [1] 9
```

A linha existente na base distintos, mas não existente na base 2 ficará com campos nulos nas colunas correspondentes a base 2.

```
join3
```

```
## # A tibble: 9 x 7
##   cpf_completo sexo  data_nascimento logradouro                numero   lat  long
##   <chr>        <chr> <date>          <chr>                     <chr>  <dbl> <dbl>
## 1 00000036832  F     1982-01-19      Rua Joaquim Nabuco        " 123" -22.9 -43.2
## 2 01238495725  F     1979-02-28      Rua Santa Luzia           " 101" -22.9 -43.1
## 3 00074960362  O     1999-10-01      Rua Senador Dantas        " 202" -22.9 -43.2
## 4 00000648038  M     2009-06-14      Rua Marechal Deodoro      " 303" -22.9 -43.2
## 5 00687925356  F     1982-03-18      Rua Nossa Senhora de Co~ " 505" -22.9 -43.2
## 6 84796132025  M     1991-09-22      Rua Santa Clara           " 606" -23.0 -43.2
## 7 00000946132  O     1977-11-02      Rua Dona Mariana          " 707" -23.0 -43.2
## 8 00000675156  <NA>  NA              Rua Visconde de Inhauma   " 456" -23.0 -43.1
## 9 05276981448  <NA>  NA              Rua Conde de Bonfim       " 404" -22.9 -43.2
```

Agora se queremos manter os dados de ambas tabelas, o full join é a solução.

```
join4 <- full_join(base_2,
                   distintos,
                   by= "cpf_completo")

nrow(join4)
```

```
## [1] 10
```

Haverá casos nulos nas colunas provenientes de ambas as bases, visto que há registros que existem em uma só das tabelas. A quantidade de linhas aqui é maior que em ambas bases.

```
join4
```

```
## # A tibble: 10 x 7
##    cpf_completo sexo  data_nascimento logradouro                numero   lat  long
##    <chr>        <chr> <date>          <chr>                     <chr>  <dbl> <dbl>
##  1 00000036832  F     1982-01-19      Rua Joaquim Nabuco        " 123" -22.9 -43.2
##  2 74628391548  F     1993-08-31      <NA>                       <NA>   NA    NA
##  3 01238495725  F     1979-02-28      Rua Santa Luzia           " 101" -22.9 -43.1
##  4 00074960362  O     1999-10-01      Rua Senador Dantas        " 202" -22.9 -43.2
##  5 00000648038  M     2009-06-14      Rua Marechal Deodoro      " 303" -22.9 -43.2
##  6 00687925356  F     1982-03-18      Rua Nossa Senhora de C~ " 505" -22.9 -43.2
##  7 84796132025  M     1991-09-22      Rua Santa Clara           " 606" -23.0 -43.2
##  8 00000946132  O     1977-11-02      Rua Dona Mariana          " 707" -23.0 -43.2
##  9 00000675156  <NA>  NA              Rua Visconde de Inhauma " 456" -23.0 -43.1
## 10 05276981448  <NA>  NA              Rua Conde de Bonfim       " 404" -22.9 -43.2
```

merge

A função merge é uma alternativa que pode se comportar com qualquer um dos joins. Se nada for especificado será um inner join.

```
join5 <- merge(base_2,
               distintos,
               by="cpf_completo")

nrow(join5)
```

```
## [1] 7
```

Para simular outros tipos de join basta usar o parâmetro all

```
join6 <- merge(base_2,
               distintos,
               by="cpf_completo",
               all.x = T,
               all.y = T)

nrow(join6)
```

```
## [1] 10
```