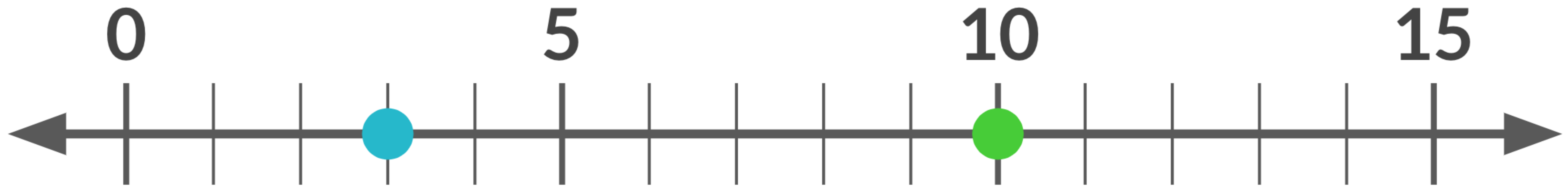
Comparing strings

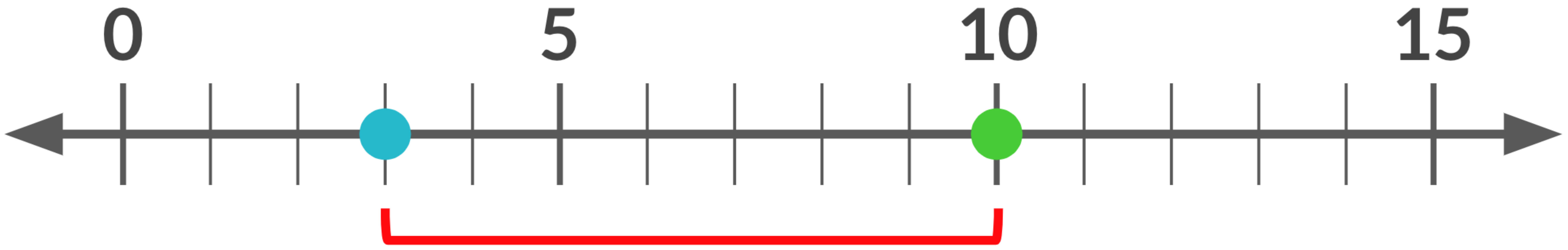
CLEANING DATA IN R

Maggie Matsui
Content Developer @ DataCamp

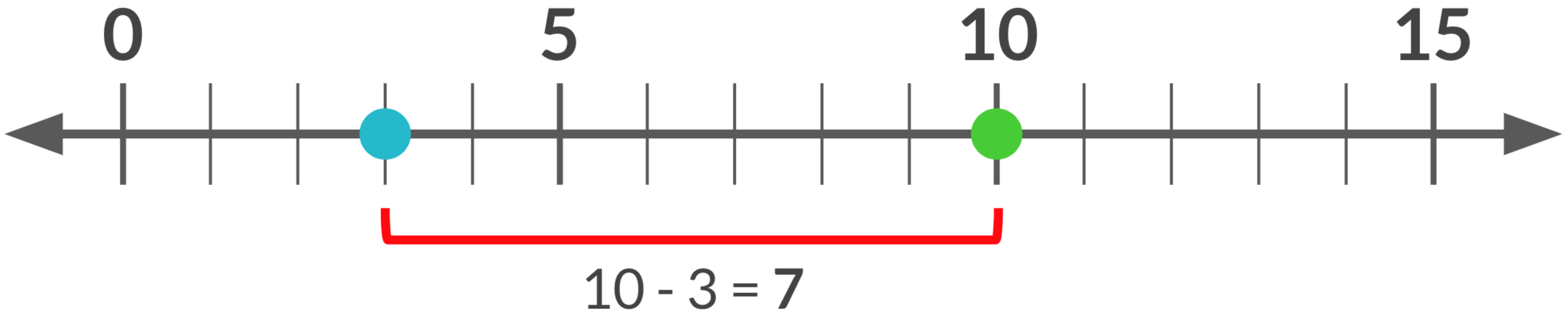
Measuring distance between values



Measuring distance between values



Measuring distance between values



What's the distance between *typhoon* and *baboon*?

Minimum edit distance

How many typos are needed to get from one string to another?

Insertion

Minimum edit distance

How many typos are needed to get from one string to another?

Insertion

Deletion

Minimum edit distance

How many typos are needed to get from one string to another?

Insertion

Deletion

Substitution

Minimum edit distance

How many typos are needed to get from one string to another?

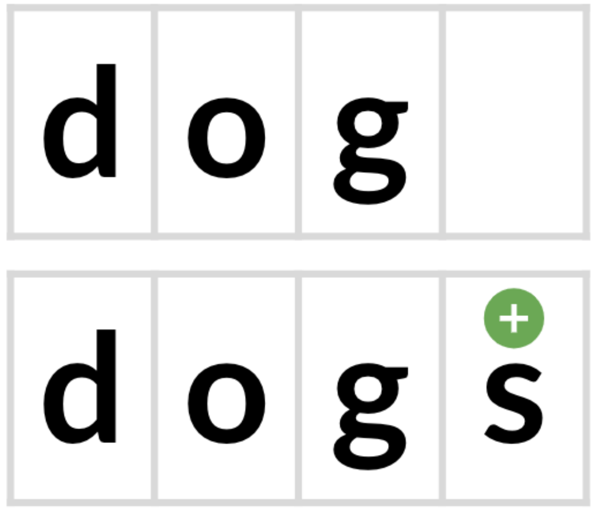
Insertion

Deletion

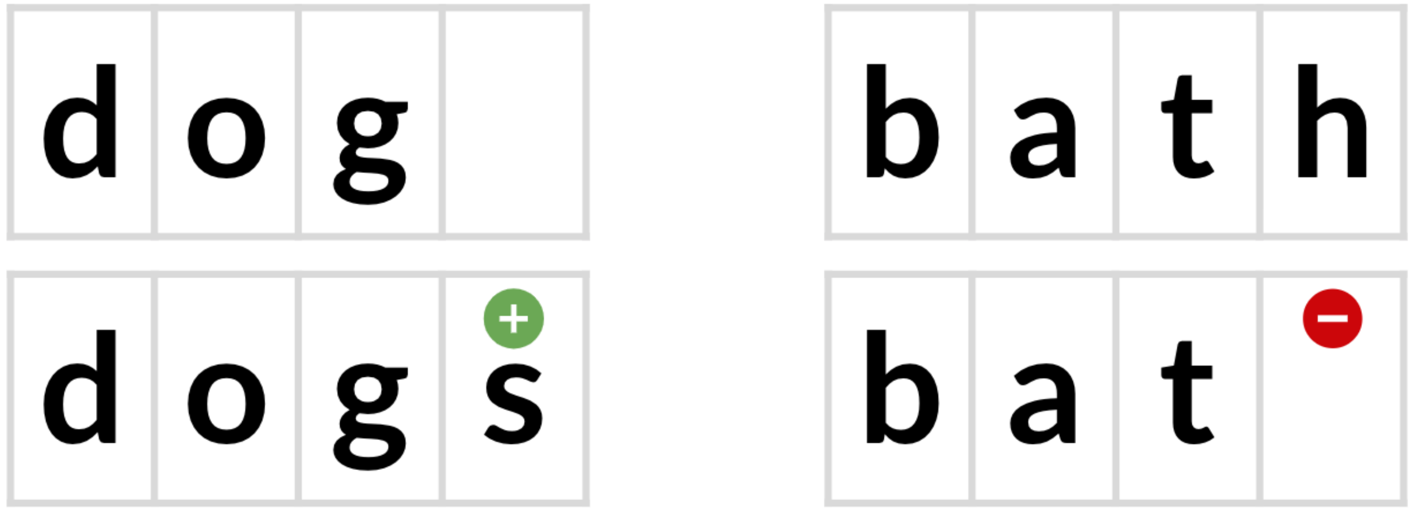
Substitution

Transposition

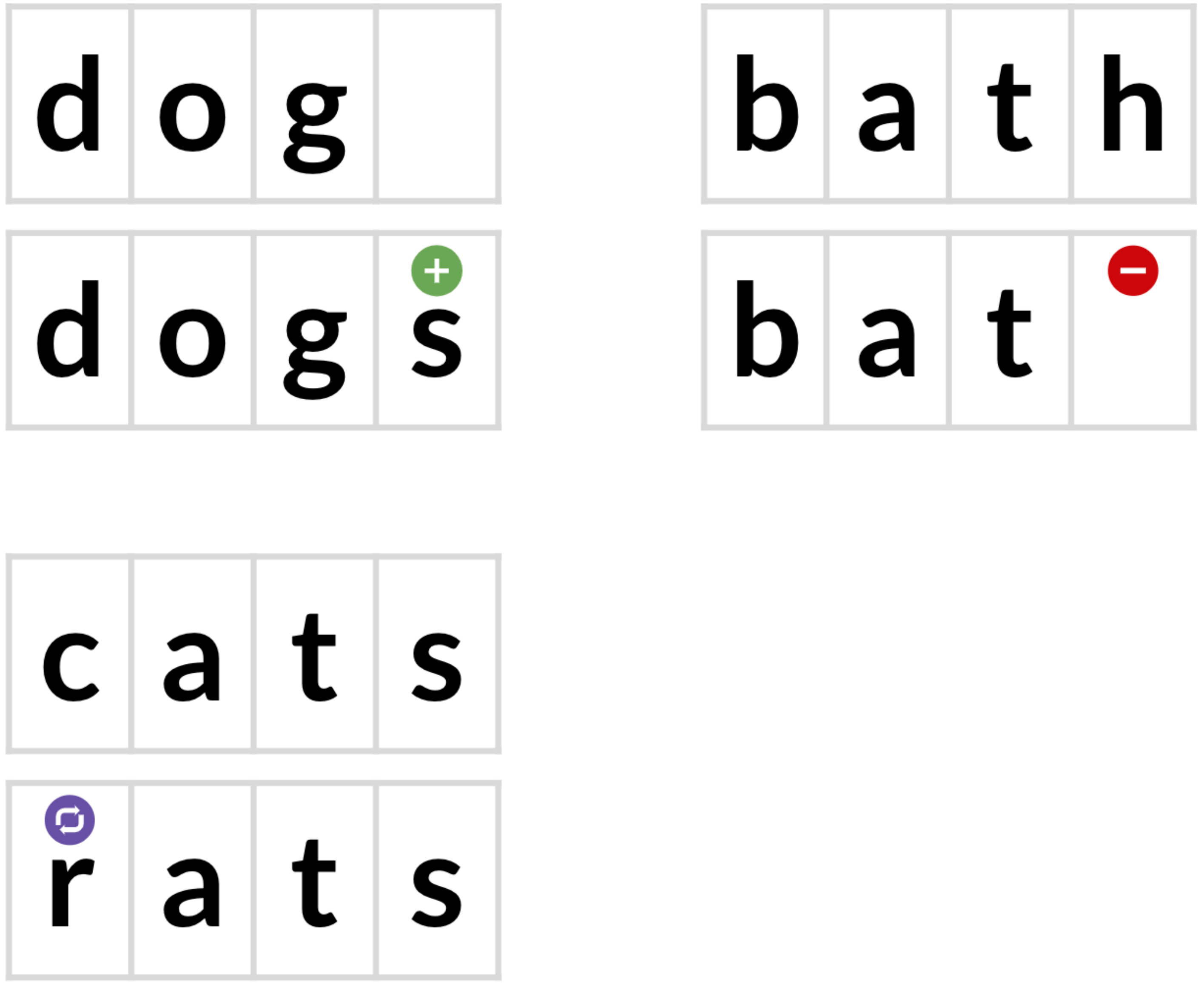
Edit distance = 1



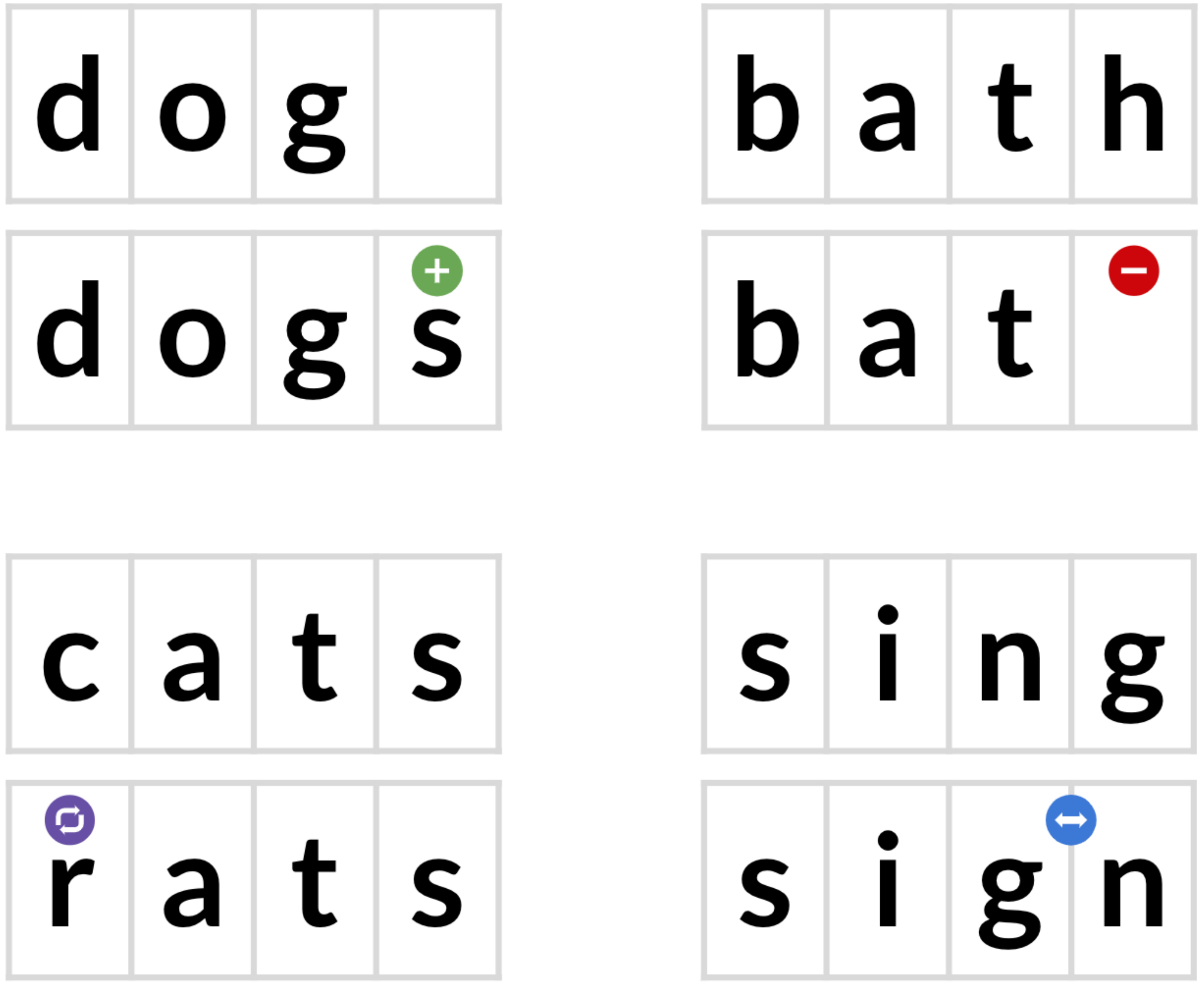
Edit distance = 1



Edit distance = 1

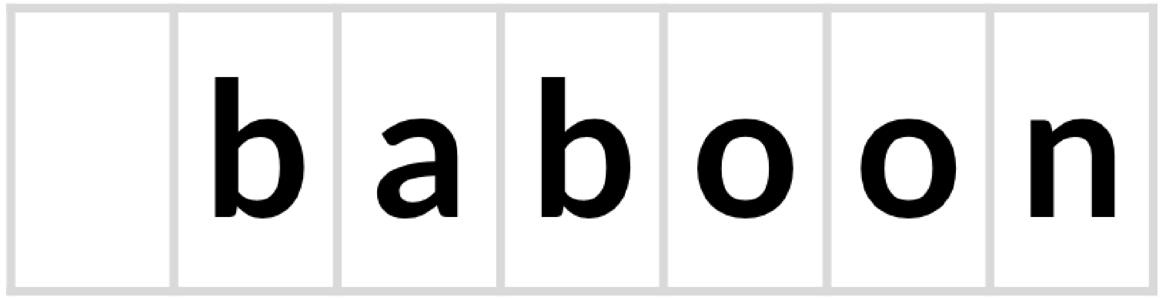


Edit distance = 1



A more complex example

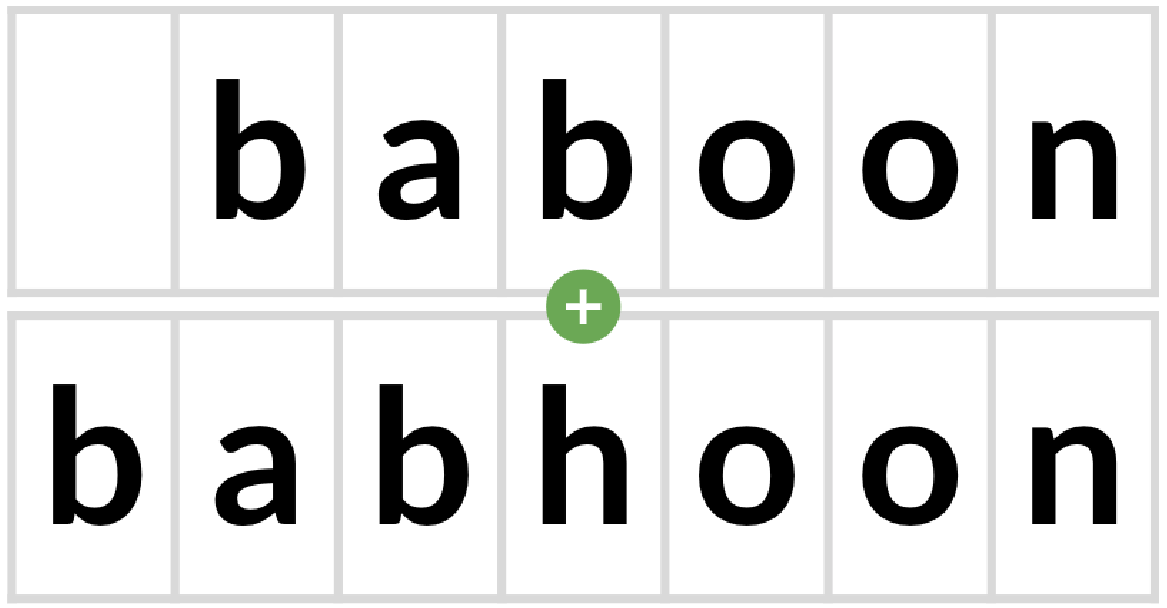
baboon $\rightarrow$ *typhoon*



A more complex example

baboon $\longrightarrow$ ***typhoon***

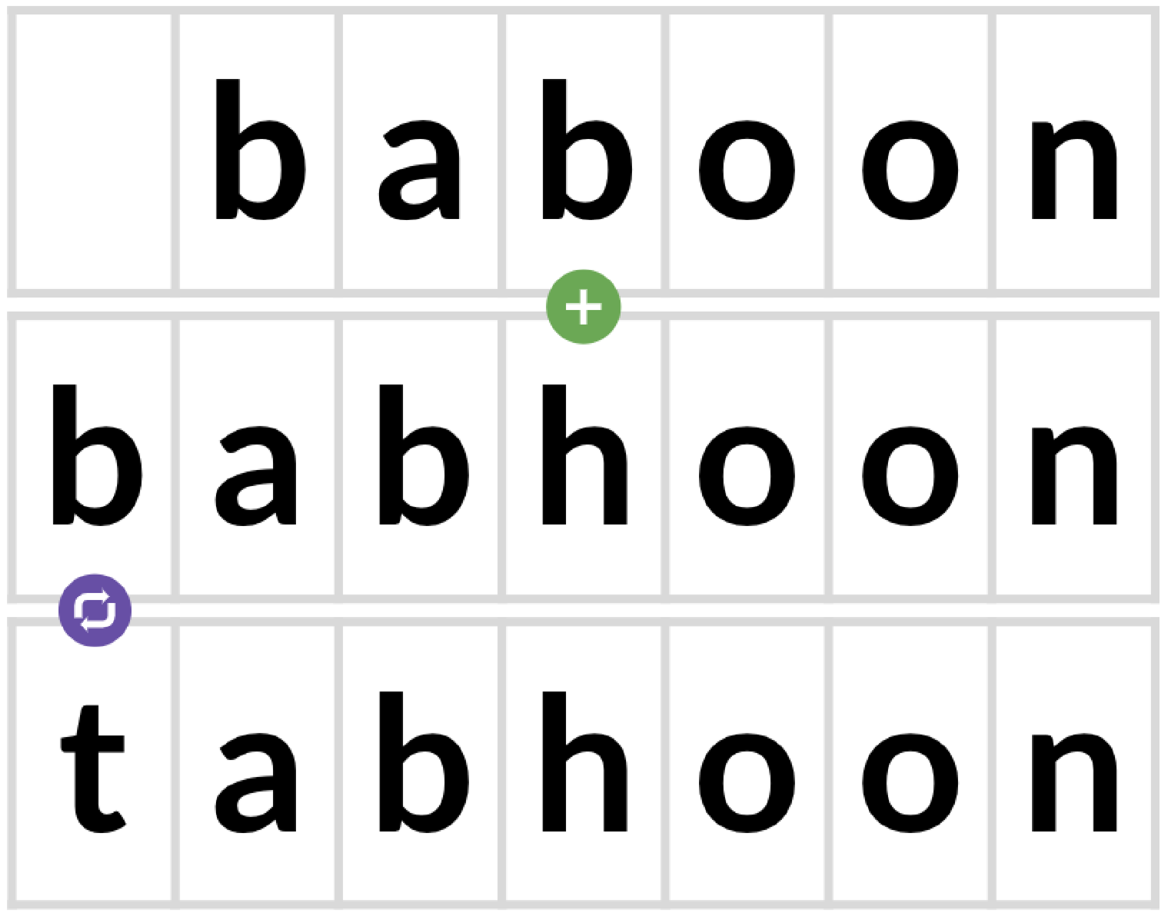
- Insert **h**



A more complex example

baboon $\rightarrow$ ***typhoon***

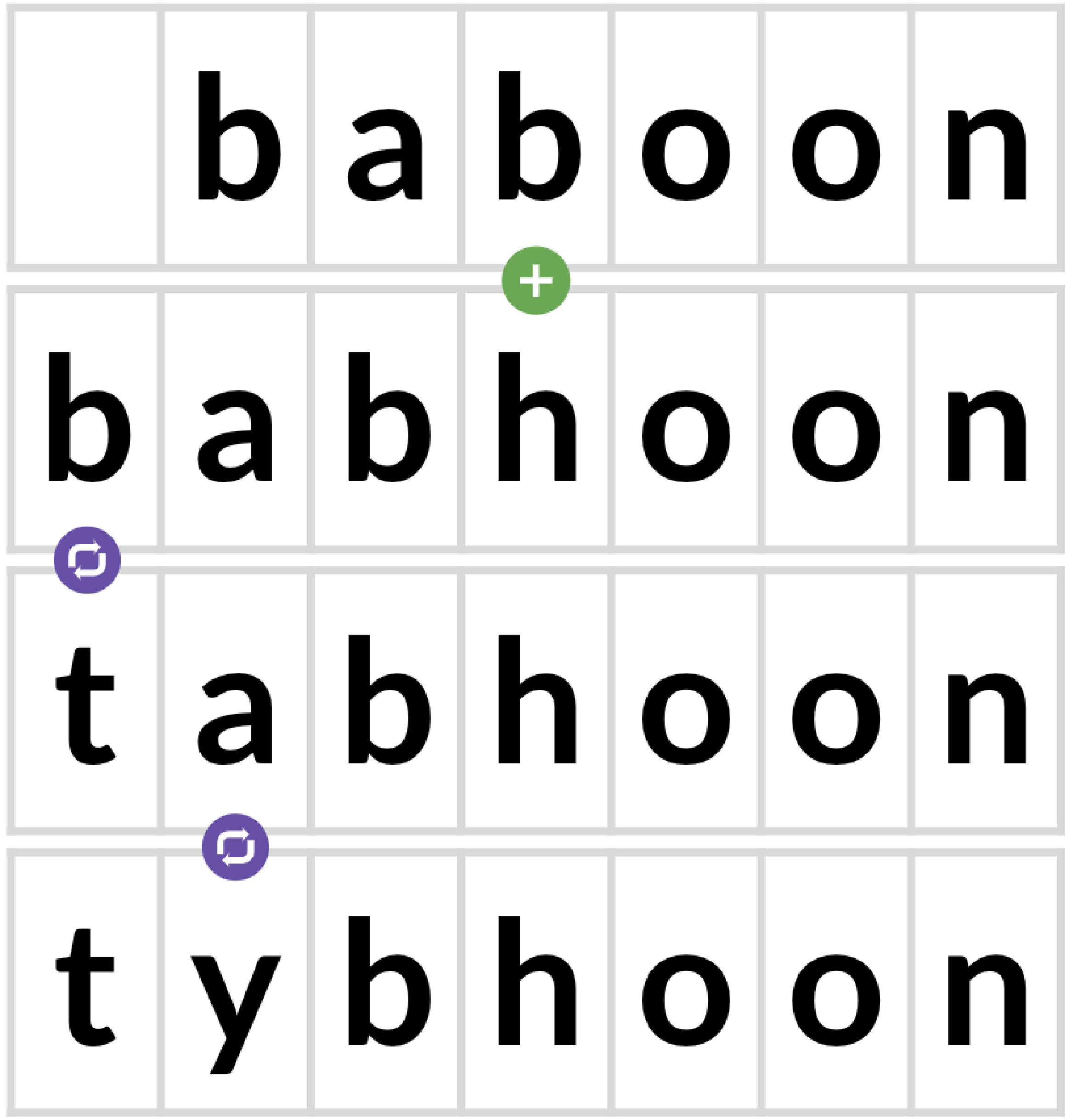
- Insert **h**
- Substitute **b** $\rightarrow$ **t**



A more complex example

baboon $\rightarrow$ ***typhoon***

- Insert **h**
- Substitute **b** $\rightarrow$ **t**
- Substitute **a** $\rightarrow$ **y**

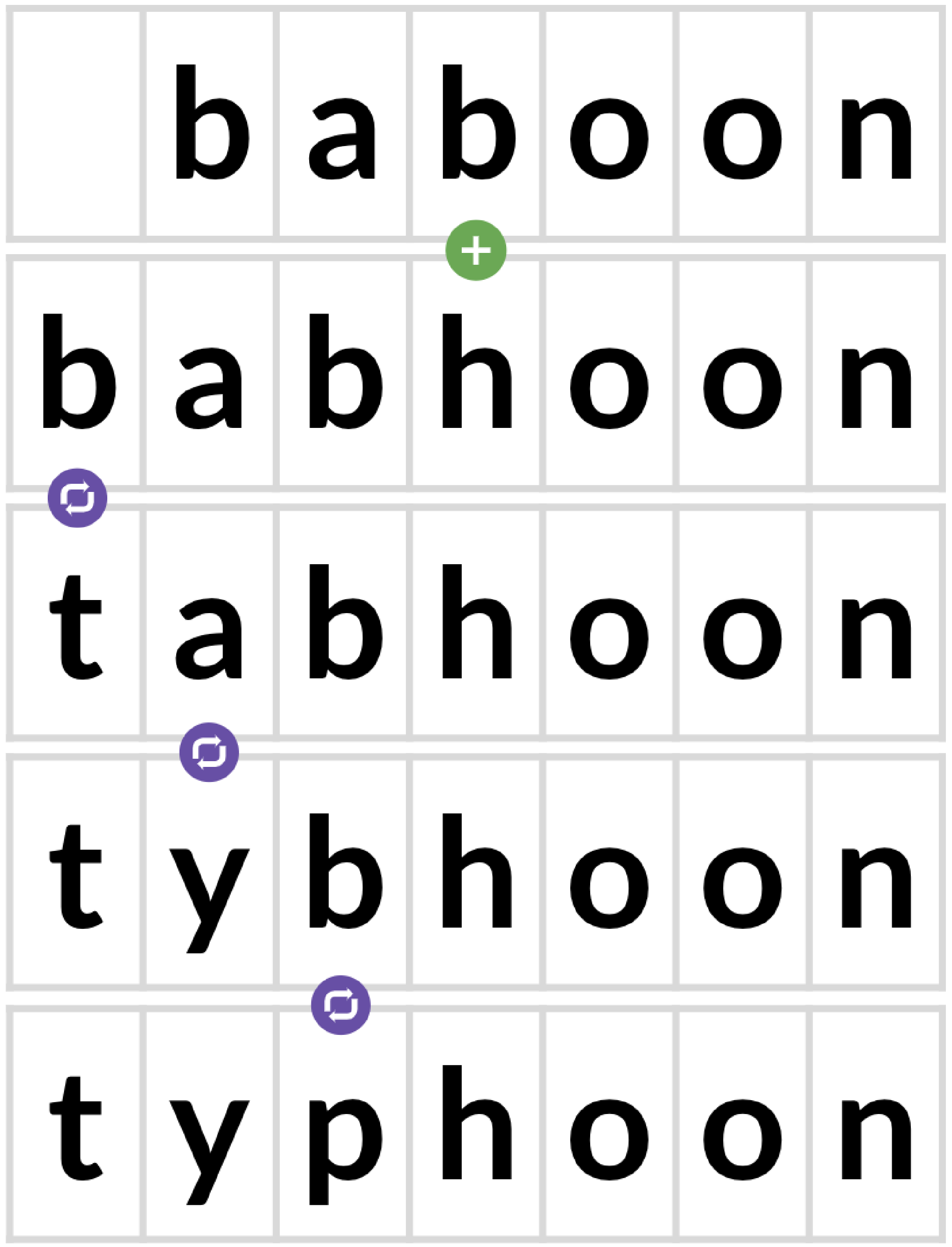


A more complex example

baboon* → *typhoon

- Insert **h**
- Substitute **b** → **t**
- Substitute **a** → **y**
- Substitute **b** → **p**

Total: 4



Types of edit distance

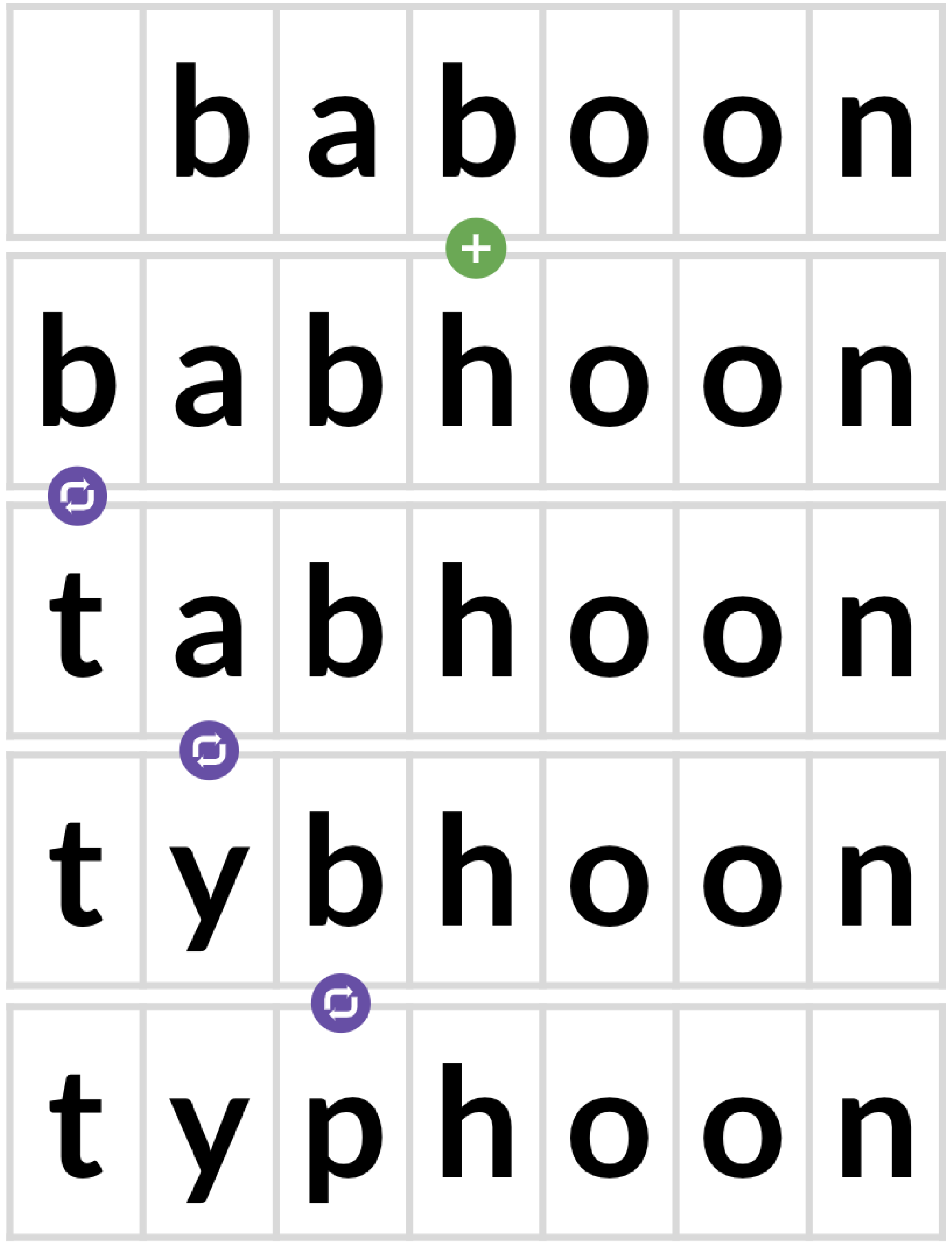
- **Damerau-Levenshtein**
  - What you just learned
- **Levenshtein**
  - Considers only substitution, insertion, and deletion
- **LCS (Longest Common Subsequence)**
  - Considers only insertion and deletion
- **Others**
  - Jaro-Winkler
  - Jaccard

Which is best?

String distance in R

```r
library(stringdist)
stringdist("baboon",
           "typhoon",
           method = "dl")
```

```
4
```



Other methods

```r
# LCS
stringdist("baboon", "typhoon",
           method = "lcs")
```

```
7
```

```r
# Jaccard
stringdist("baboon", "typhoon",
           method = "jaccard")
```

```
0.75
```

Comparing strings to clean data

- In Chapter 2:
  - `"EU"` , `"eur"` , `"Europ"` $\rightarrow$ `"Europe"`
- What if there are too many variations?
  - `"EU"` , `"eur"` , `"Europ"` , `"Europa"` , `"Erope"` , `"Evropa"` , ... $\rightarrow$ `"Europe"` ?
  - Use string distance!

Comparing strings to clean data

```
survey
```

```
         city move_score
1      chicgo          4
2  los angles          4
3     chicogo          5
4     new yrk          5
5   new yoork          2
6    seatttle          3
7  losangeles          4
8     seeatle          2
...
```

```
cities
```

```
         city
1    new york
2     chicago
3 los angeles
4     seattle
```

Remapping using string distance

```
library(fuzzyjoin)
stringdist_left_join(survey, cities, by = "city", method = "dl")
```

```
        city.x move_score      city.y
1       chicgo          4     chicago
2   los angles          4 los angeles
3      chicogo          5     chicago
4      new yrk          5    new york
5    new yoork          2    new york
6     seatttle          3     seattle
7   losangeles          4 los angeles
8      seeatle          2     seattle
9      siattle          1     seattle
...
```

Remapping using string distance

```
stringdist_left_join(survey, cities, by = "city", method = "dl", max_dist = 1)
```

```
        city.x move_score      city.y
1       chicgo          4     chicago
2   los angles          4 los angeles
3      chicogo          5     chicago
4      new yrk          5    new york
5    new yoork          2    new york
6     seatttle          3     seattle
7   losangeles          4 los angeles
8      seeatle          2        <NA>
9      siattle          1     seattle
...
```

Let's practice!

CLEANING DATA IN R

Generating and comparing pairs

CLEANING DATA IN R

Maggie Matsui
Content Developer @ DataCamp

When joins won't work

Event	Time
Houston Rockets vs Chicago Bulls	19:00
Miami Heat vs Los Angeles Lakers	19:00
Brooklyn Nets vs Orlando Magic	20:00
Denver Nuggets vs Miami Heat	21:00
San Antonio Spurs vs Atlanta Hawks	21:00

Event	Time
NBA: Nets vs Magic	8pm
NBA: Bulls vs Rockets	9pm
NBA: Heat vs Lakers	7pm
NBA: Grizzlies vs Heat	10pm
NBA: Heat vs Cavaliers	9pm

When joins won't work

Event	Time
Houston Rockets vs Chicago Bulls	19:00
Miami Heat vs Los Angeles Lakers	19:00
Brooklyn Nets vs Orlando Magic	20:00
Denver Nuggets vs Miami Heat	21:00
San Antonio Spurs vs Atlanta Hawks	21:00

Event	Time	
NBA: Nets vs Magic	8pm	
NBA: Bulls vs Rockets	9pm	
NBA: Heat vs Lakers	7pm	
NBA: Grizzlies vs Heat	10pm	✓
NBA: Heat vs Cavaliers	9pm	✓

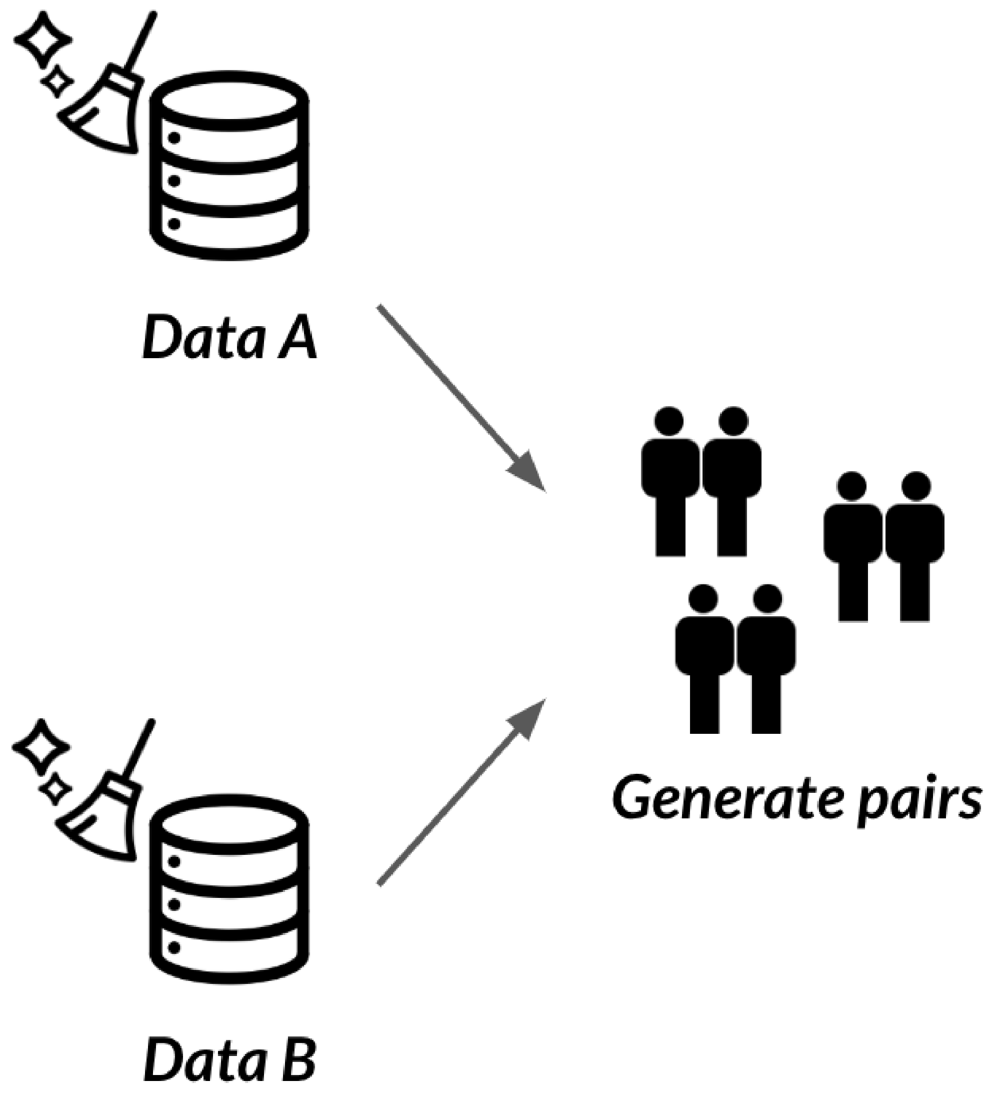
What is record linkage?

Data A

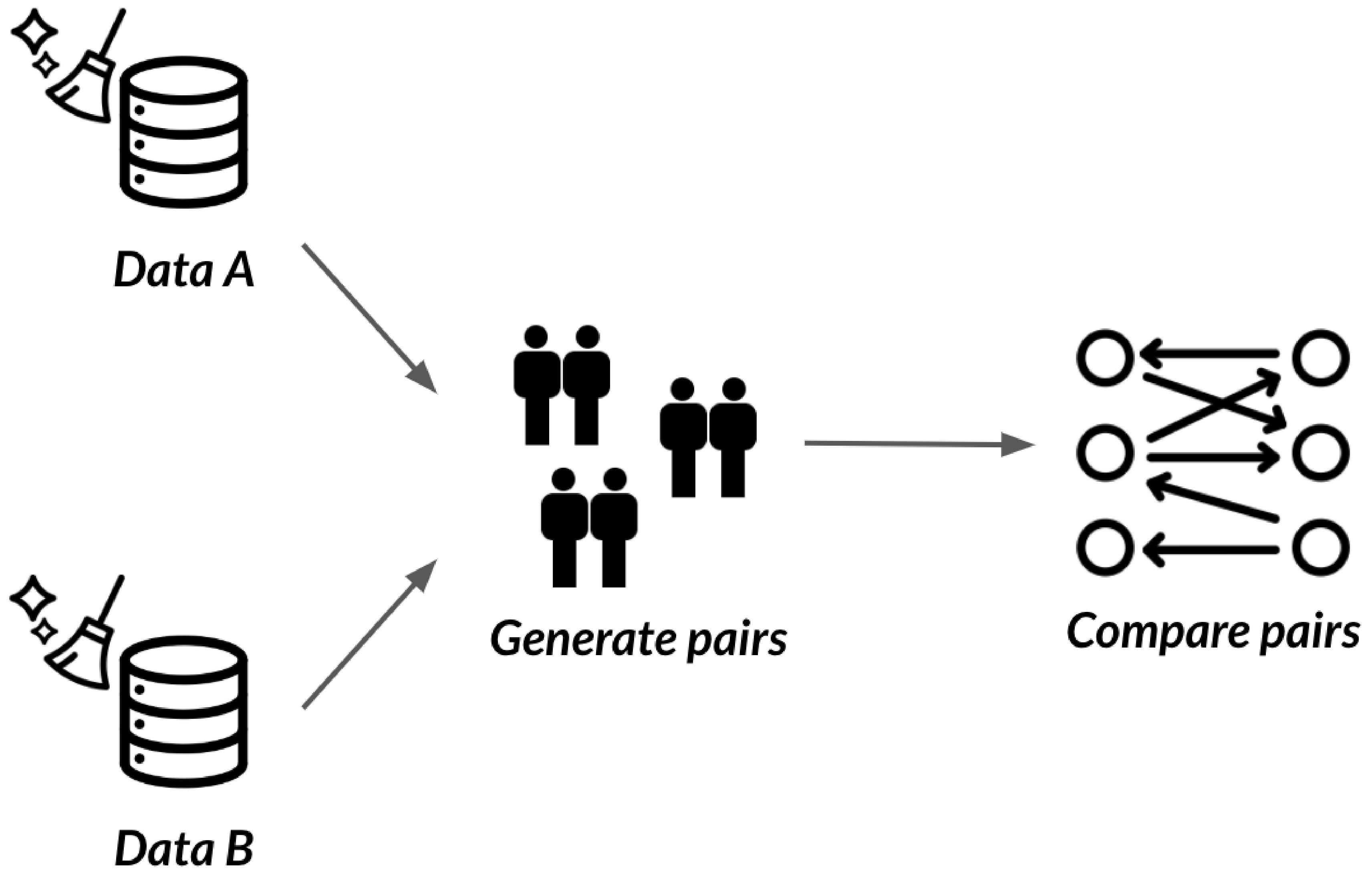
Data B

What is record linkage?

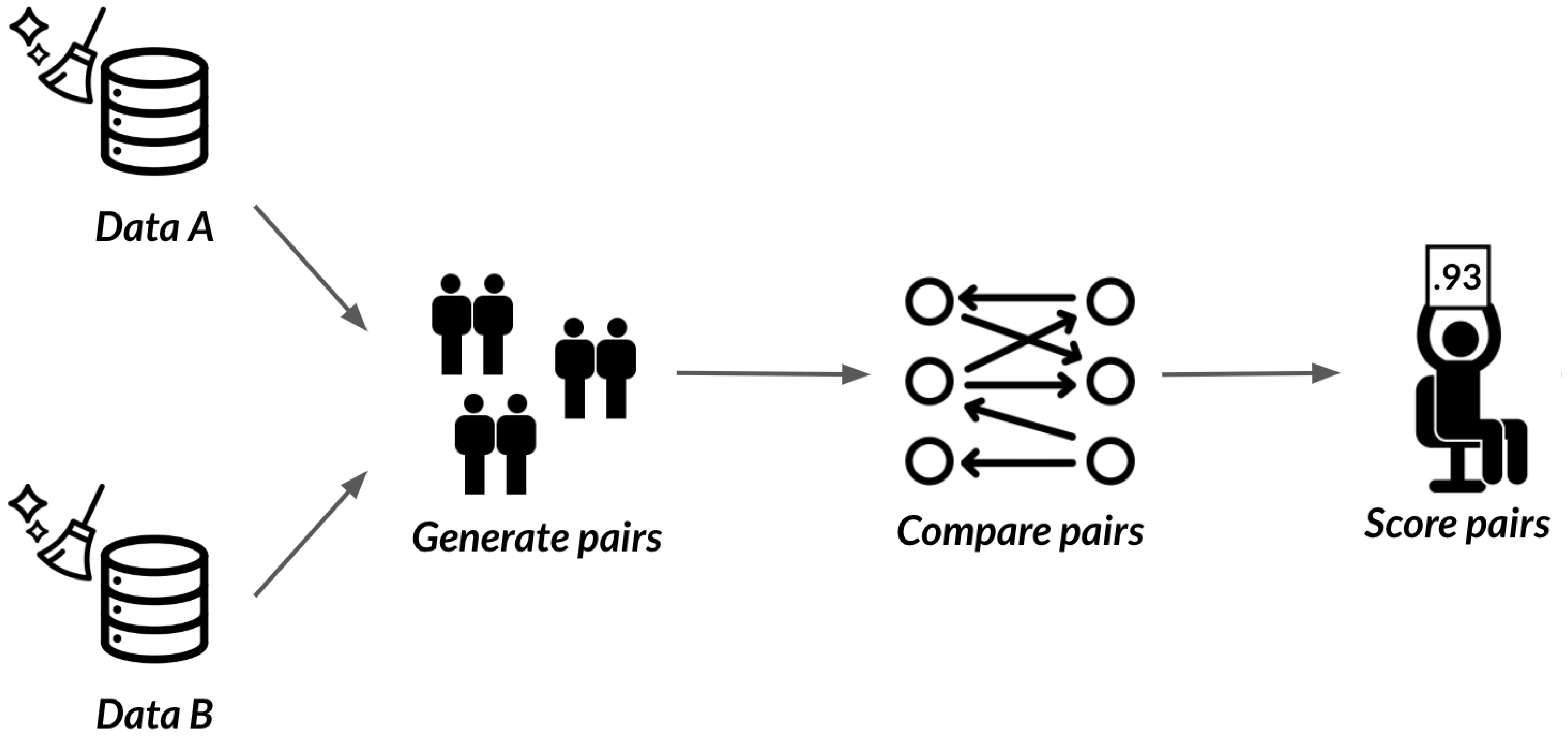
What is record linkage?



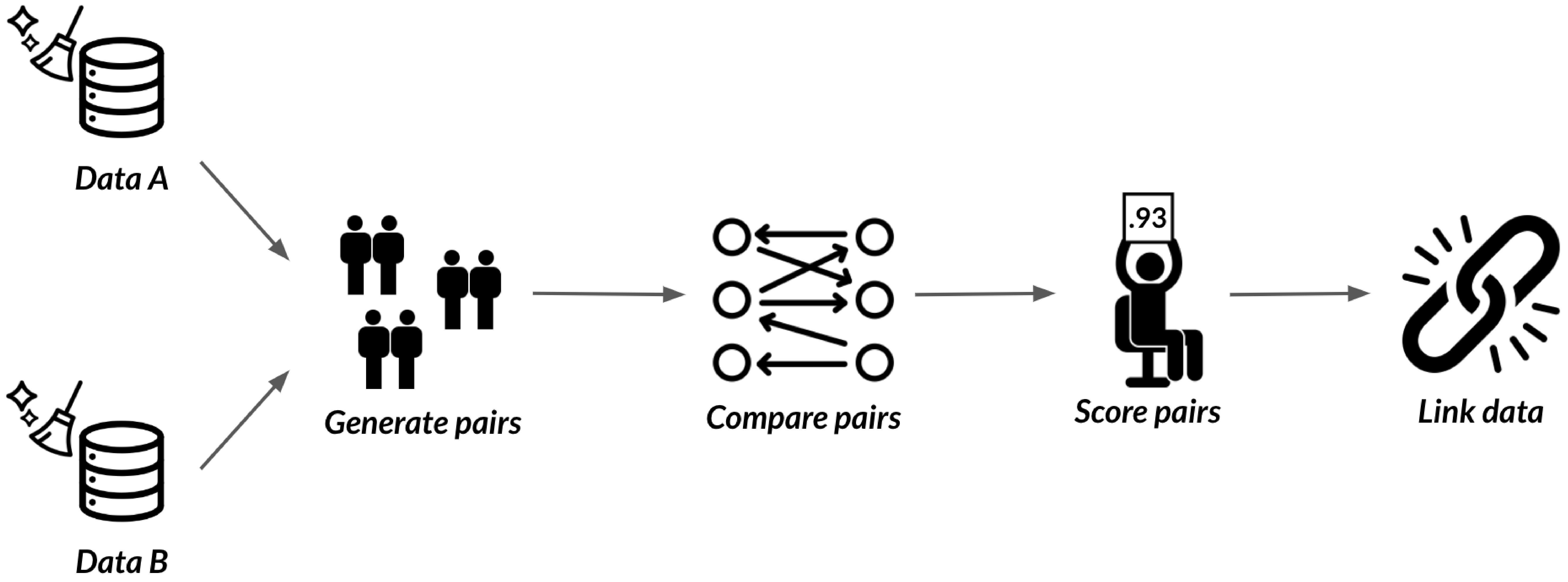
What is record linkage?



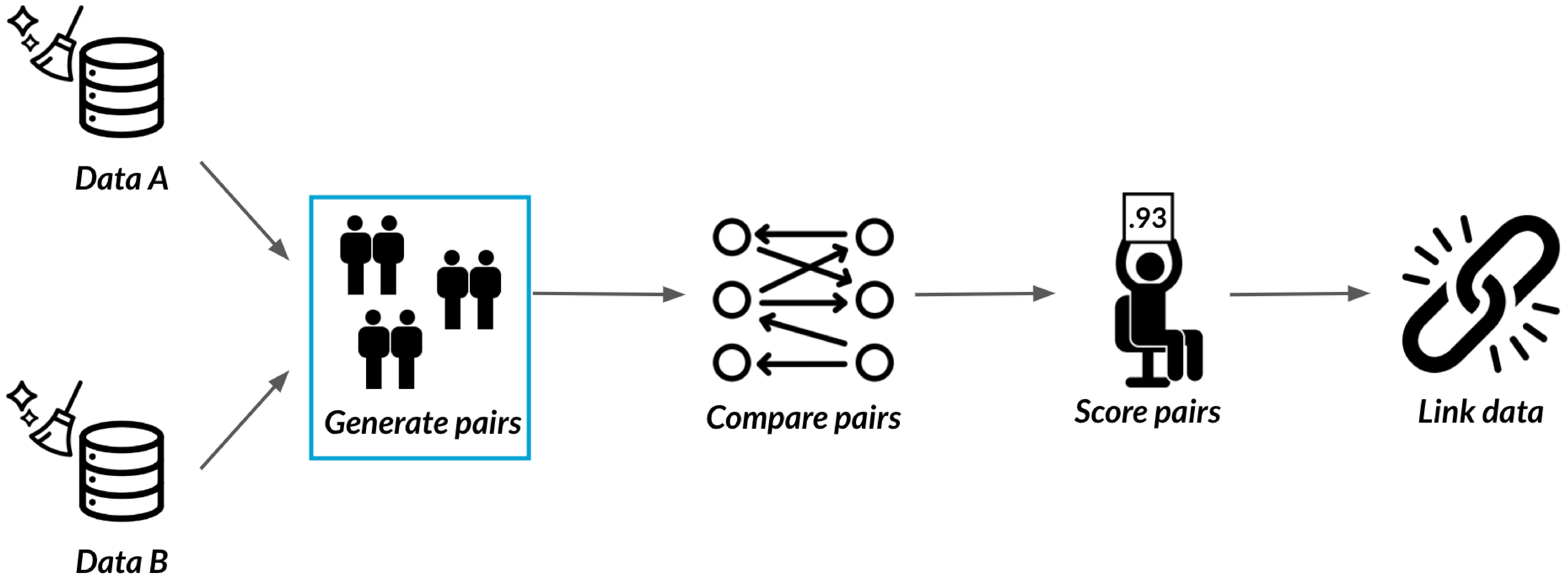
What is record linkage?



What is record linkage?



What is record linkage?



Pairs of records

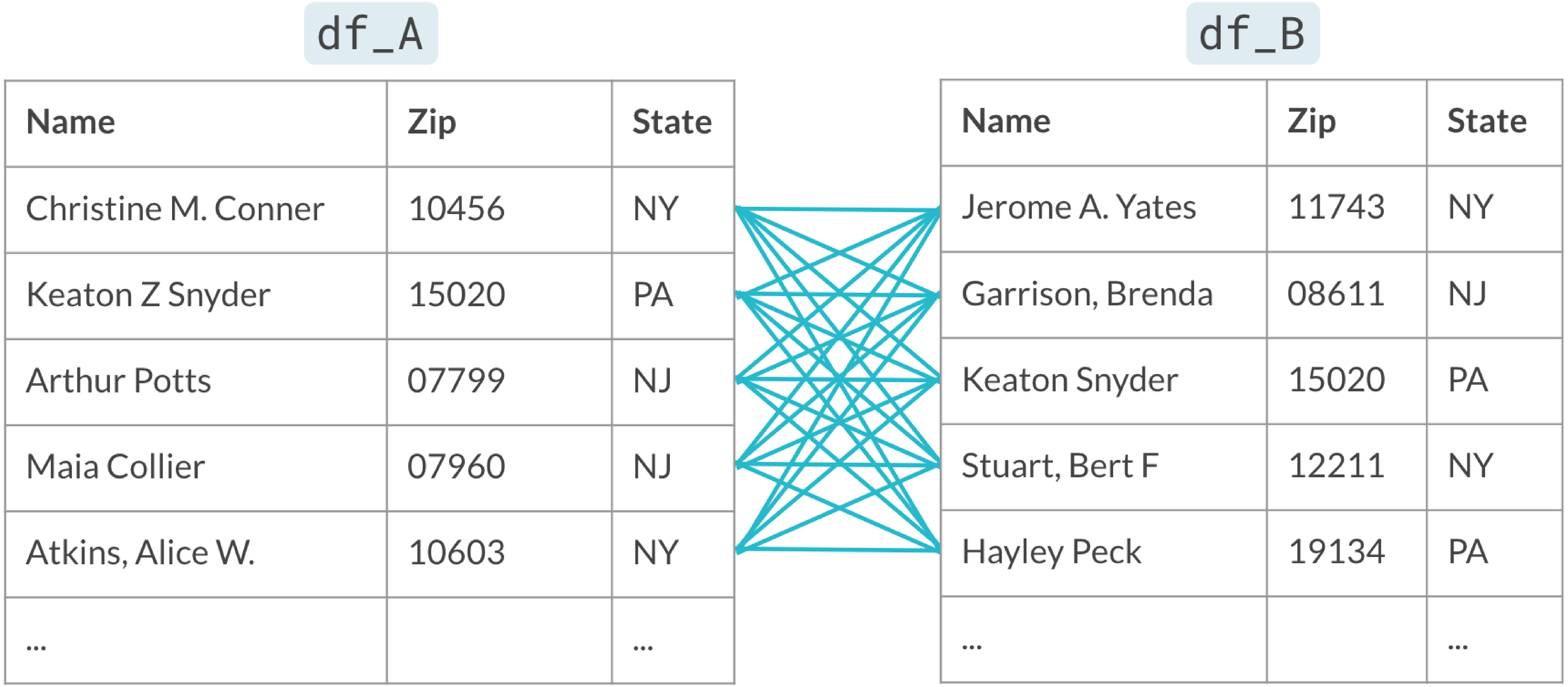
`df_A`

Name	Zip	State
Christine M. Conner	10456	NY
Keaton Z Snyder	15020	PA
Arthur Potts	07799	NJ
Maia Collier	07960	NJ
Atkins, Alice W.	10603	NY
...		...

`df_B`

Name	Zip	State
Jerome A. Yates	11743	NY
Garrison, Brenda	08611	NJ
Keaton Snyder	15020	PA
Stuart, Bert F	12211	NY
Hayley Peck	19134	PA
...		...

Generating pairs



`df_A`

Name	Zip	State
Christine M. Conner	10456	NY
Keaton Z Snyder	15020	PA
Arthur Potts	07799	NJ
Maia Collier	07960	NJ
Atkins, Alice W.	10603	NY
...		...

`df_B`

Name	Zip	State
Jerome A. Yates	11743	NY
Garrison, Brenda	08611	NJ
Keaton Snyder	15020	PA
Stuart, Bert F	12211	NY
Hayley Peck	19134	PA
...		...

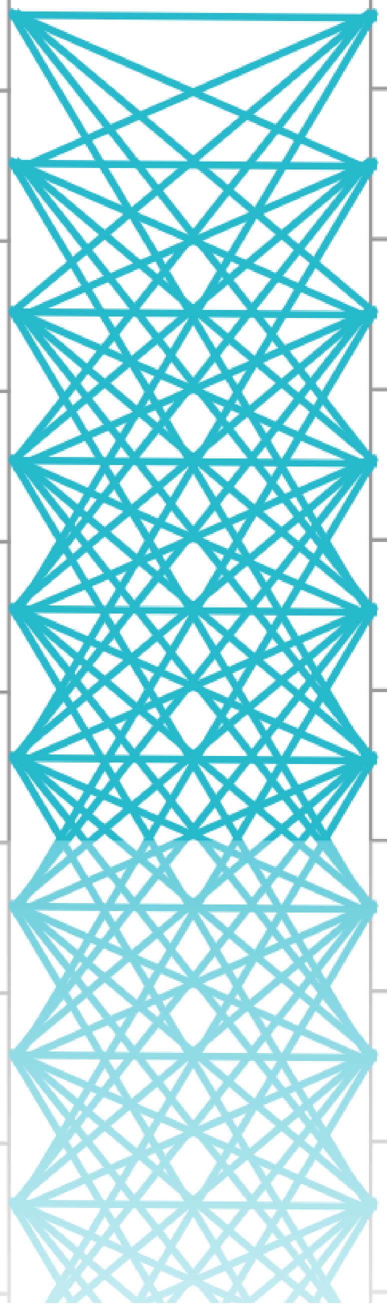
Generating pairs in R

```
library(reclin)
pair_blocking(df_A, df_B)
```

```
Simple blocking
  No blocking used.
  First data set:  5 records
  Second data set: 5 records
  Total number of pairs: 25 pairs
ldat with 25 rows and 2 columns
   x y
1  1 1
2  2 1
3  3 1
...
```

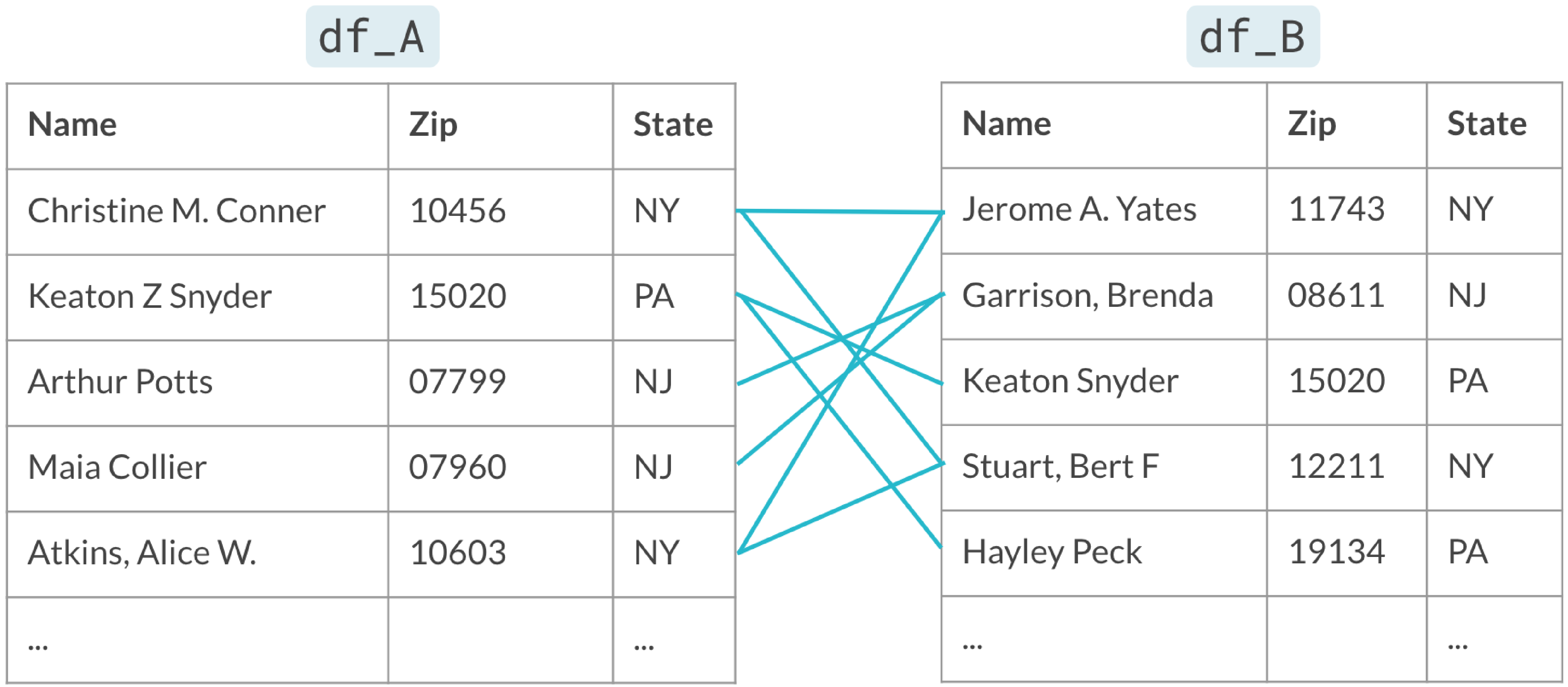
Too many pairs

Name	Zip	State
Christine M. Conner	10456	NY
Keaton Z Snyder	15020	PA
Arthur Potts	07799	NJ
Maia Collier	07960	NJ
Atkins, Alice W.	10603	NY
...	...	...
...	...	...
...	...	...



Name	Zip	State
Jerome A. Yates	11743	NY
Garrison, Brenda	08611	NJ
Keaton Snyder	15020	PA
Stuart, Bert F	12211	NY
Hayley Peck	19134	PA
...	...	...
...	...	...
...	...	...

Blocking



df_A

Name	Zip	State
Christine M. Conner	10456	NY
Keaton Z Snyder	15020	PA
Arthur Potts	07799	NJ
Maia Collier	07960	NJ
Atkins, Alice W.	10603	NY
...		...

df_B

Name	Zip	State
Jerome A. Yates	11743	NY
Garrison, Brenda	08611	NJ
Keaton Snyder	15020	PA
Stuart, Bert F	12211	NY
Hayley Peck	19134	PA
...		...

Only consider pairs when they agree on the blocking variable (State)

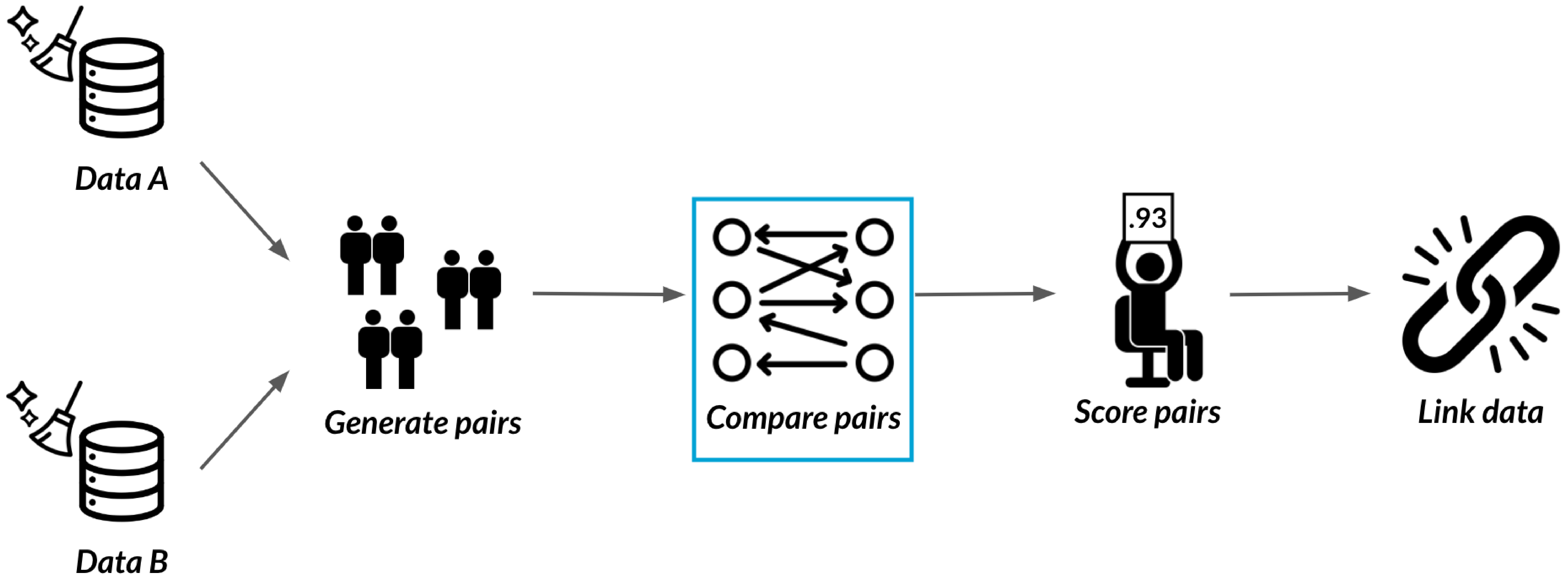
Pair blocking in R

```r
pair_blocking(df_A, df_B, blocking_var = "state")
```

```
Simple blocking
  Blocking variable(s): state
  First data set:  5 records
  Second data set: 5 records
  Total number of pairs: 8 pairs
```

```
ldat with 8 rows and 2 columns
  x y
1 1 1
2 1 4
3 2 3
4 2 5
5 3 2
6 4 2
7 5 1
8 5 4
```

Comparing pairs



Comparing pairs

```r
pair_blocking(df_A, df_B, blocking_var = "state") %>%
  compare_pairs(by = "name", default_comparator = lcs())
```

```
Compare                                     ldat with 8 rows and 3 columns
  By: name                                      x y      name
                                              1 1 1 0.3529412
Simple blocking                               2 1 4 0.3030303
  Blocking variable(s): state                 3 2 3 0.9285714
  First data set:  5 records                  4 2 5 0.2962963
  Second data set: 5 records                  ...
  Total number of pairs: 8 pairs              8 5 4 0.3333333
```

Comparing multiple columns

```r
pair_blocking(df_A, df_B, blocking_var = "state") %>%
  compare_pairs(by = c("name", "zip"), default_comparator = lcs())
```

```
Compare                                     ldat with 8 rows and 4 columns
  By: name, zip                               x y      name zip
                                            1 1 1 0.3529412 0.4
Simple blocking                             2 1 4 0.3030303 0.2
  Blocking variable(s): state               3 2 3 0.9285714 1.0
  First data set:  5 records                4 2 5 0.2962963 0.2
  Second data set: 5 records                ...
  Total number of pairs: 8 pairs            8 5 4 0.3333333 0.2
```

Different comparators

- `default_comparator = lcs()`
- `default_comparator = jaccard()`
- `default_comparator = jaro_winkler()`

Let's practice!

CLEANING DATA IN R

Scoring and linking

CLEANING DATA IN R

Maggie Matsui
Content Developer @ DataCamp

Last lesson

```
df_A
```

```
                 name   zip state
1 Christine M. Conner 10456    NY
2      Keaton Z Snyder 15020    PA
3         Arthur Potts 07799    NJ
4         Maia Collier 07960    NJ
5    Atkins, Alice W. 10603    NY
```

```
df_B
```

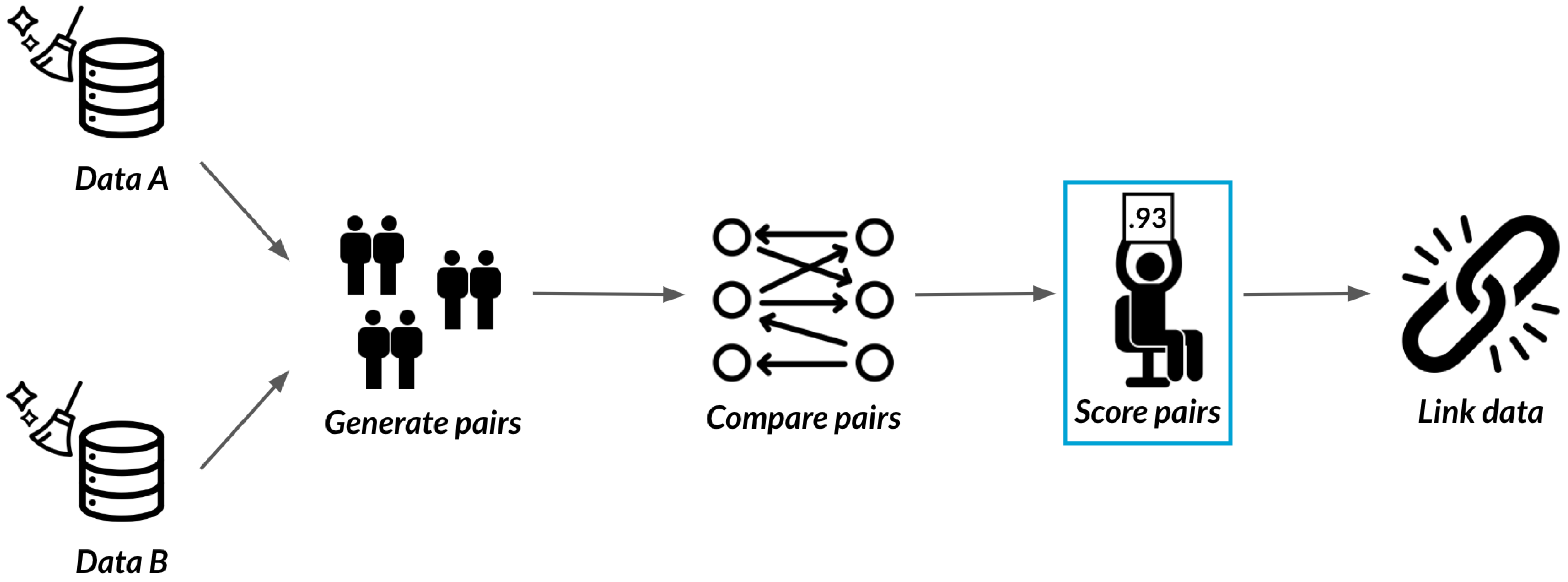
```
                 name   zip state
1   Jerome A. Yates 11743    NY
2 Garrison, Brenda 08611    NJ
3     Keaton Snyder 15020    PA
4   Stuart, Bert F 12211    NY
5       Hayley Peck 19134    PA
```

Where we left off

```r
pair_blocking(df_A, df_B, blocking_var = "state") %>%
  compare_pairs(by = c("name", "zip"), default_comparator = lcs())
```

```
  x y      name zip
1 1 1 0.3529412 0.4
2 1 4 0.3030303 0.2
3 2 3 0.9285714 1.0
4 2 5 0.2307692 0.2
5 3 2 0.2142857 0.2
6 4 2 0.2857143 0.4
7 5 1 0.1935484 0.4
8 5 4 0.3333333 0.2
```

Scoring pairs



Scoring with sums

```
  x y      name   zip
1 1 1 0.3529412 + 0.4 =
2 1 4 0.3030303 + 0.2 =
3 2 3 0.9285714 + 1.0 =
4 2 5 0.2307692 + 0.2 =
5 3 2 0.2142857 + 0.2 =
6 4 2 0.2857143 + 0.4 =
7 5 1 0.1935484 + 0.4 =
8 5 4 0.3333333 + 0.2 =
```

```r
pair_blocking(df_A, df_B, blocking_var = "state") %>%
  compare_pairs(by = c("name", "zip"), default_comparator = lcs()) %>%
  score_simsum()
```

```
  x y      name zip    simsum
1 1 1 0.3529412 0.4 0.7529412
2 1 4 0.3030303 0.2 0.5030303
3 2 3 0.9285714 1.0 1.9285714
4 2 5 0.2307692 0.2 0.4307692
5 3 2 0.2142857 0.2 0.4142857
6 4 2 0.2857143 0.4 0.6857143
7 5 1 0.1935484 0.4 0.5935484
8 5 4 0.3333333 0.2 0.5333333
```

```r
pair_blocking(df_A, df_B, blocking_var = "state") %>%
  compare_pairs(by = c("name", "zip"), default_comparator = lcs()) %>%
  score_simsum()
```

```
  x y      name zip    simsum
1 1 1 0.3529412 0.4 0.7529412
2 1 4 0.3030303 0.2 0.5030303
3 2 3 0.9285714 1.0 1.9285714  <--
4 2 5 0.2307692 0.2 0.4307692
5 3 2 0.2142857 0.2 0.4142857
6 4 2 0.2857143 0.4 0.6857143
7 5 1 0.1935484 0.4 0.5935484
8 5 4 0.3333333 0.2 0.5333333
```

Disadvantages of summing

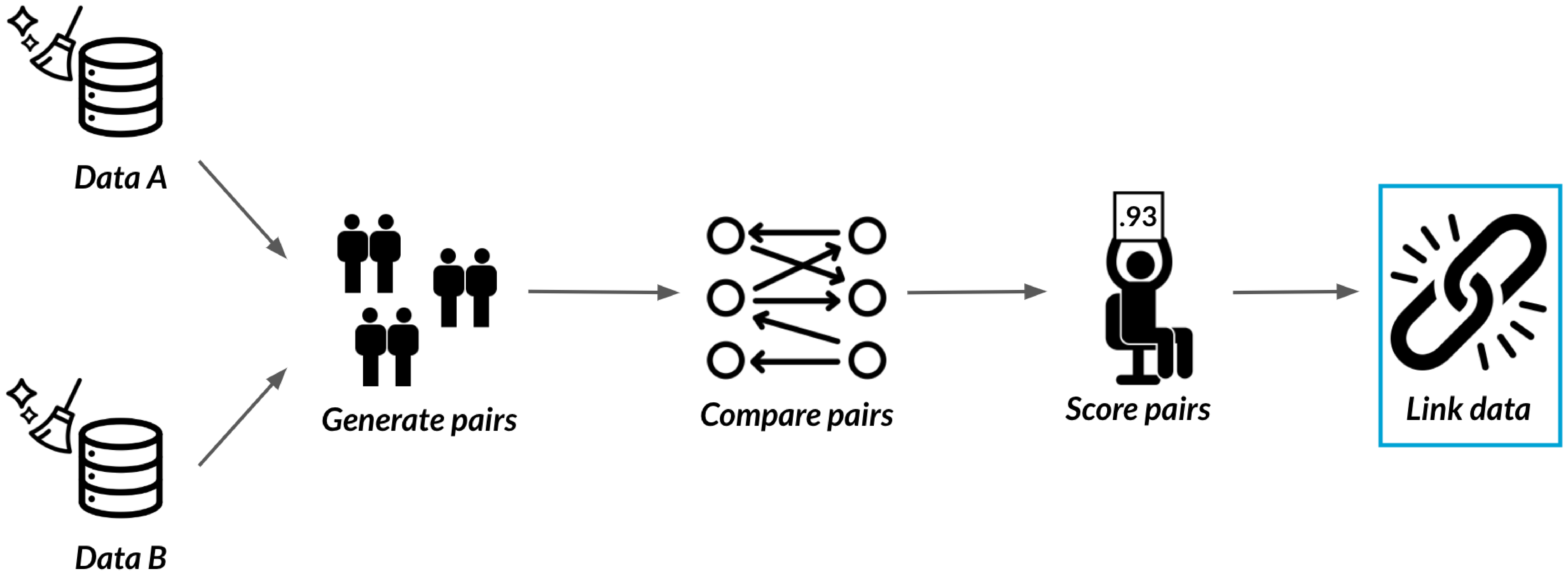
- 2 records with a similar name (Keaton Z Snyder & Keaton Snyder) are more likely to be a match
- 2 records with the same sex (Male & Male) are not as likely to be a match
- Use probabilistic scoring!

Scoring probabilistically

```
pair_blocking(df_A, df_B, blocking_var = "state") %>%
  compare_pairs(by = c("name", "zip"), default_comparator = lcs()) %>%
  score_problink()
```

```
  x y      name zip    weight
1 1 1 0.3529412 0.4 -1.011599
2 1 4 0.3030303 0.2 -2.219198
3 2 3 0.9285714 1.0 16.019278
4 2 5 0.2307692 0.2 -2.590260
5 3 2 0.2142857 0.2 -2.685570
6 4 2 0.2857143 0.4 -1.321753
7 5 1 0.1935484 0.4 -1.832576
8 5 4 0.3333333 0.2 -2.079436
```

Linking pairs



Selecting matches

```r
pair_blocking(df_A, df_B, blocking_var = "state") %>%
  compare_pairs(by = c("name", "zip"), default_comparator = lcs()) %>%
  score_problink() %>%
  select_n_to_m()
```

```
  x y      name zip    weight select
1 1 1 0.3529412 0.4 -1.011599  FALSE
2 1 4 0.3030303 0.2 -2.219198  FALSE
3 2 3 0.9285714 1.0 16.019278   TRUE
4 2 5 0.2307692 0.2 -2.590260  FALSE
5 3 2 0.2142857 0.2 -2.685570  FALSE
6 4 2 0.2857143 0.4 -1.321753  FALSE
...
```

Linking the data

```
pair_blocking(df_A, df_B, blocking_var = "state") %>%
  compare_pairs(by = c("name", "zip"), default_comparator = lcs()) %>%
  score_problink() %>%
  select_n_to_m() %>%
  link()
```

Linked data

```
                 name.x zip.x state.x            name.y zip.y state.y
1        Keaton Z Snyder 15020      PA     Keaton Snyder 15020      PA
2 Christine M. Conner 10456      NY              <NA>  <NA>    <NA>
3          Arthur Potts 07799      NJ              <NA>  <NA>    <NA>
4          Maia Collier 07960      NJ              <NA>  <NA>    <NA>
5      Atkins, Alice W. 10603      NY              <NA>  <NA>    <NA>
6                  <NA>  <NA>    <NA>  Jerome A. Yates 11743      NY
7                  <NA>  <NA>    <NA> Garrison, Brenda 08611      NJ
8                  <NA>  <NA>    <NA>   Stuart, Bert F 12211      NY
9                  <NA>  <NA>    <NA>      Hayley Peck 19134      PA
```

Let's practice!

CLEANING DATA IN R

Congratulations!

CLEANING DATA IN R

Maggie Matsui
Content Developer, DataCamp

What you learned

Diagnose dirty data

Side effects of dirty data

Clean data

Chapter 1: Common Data Problems

Data Type Constraints

Strings
Numeric data
...

Data Range Constraints

Out of range data
Out of range dates
...

Uniqueness Constraints

Finding duplicates
Treating them
...

Chapter 2: Text and Categorical Data

Membership Constraints

Finding inconsistent categories
Treating them with joins
...

Categorical Variables

Finding inconsistent categories
Collapsing them into less
...

Cleaning Text Data

Unifying formats
Finding lengths
...

Chapter 3: Advanced Data Problems

Uniformity

Unifying currency formats
Unifying date formats
...

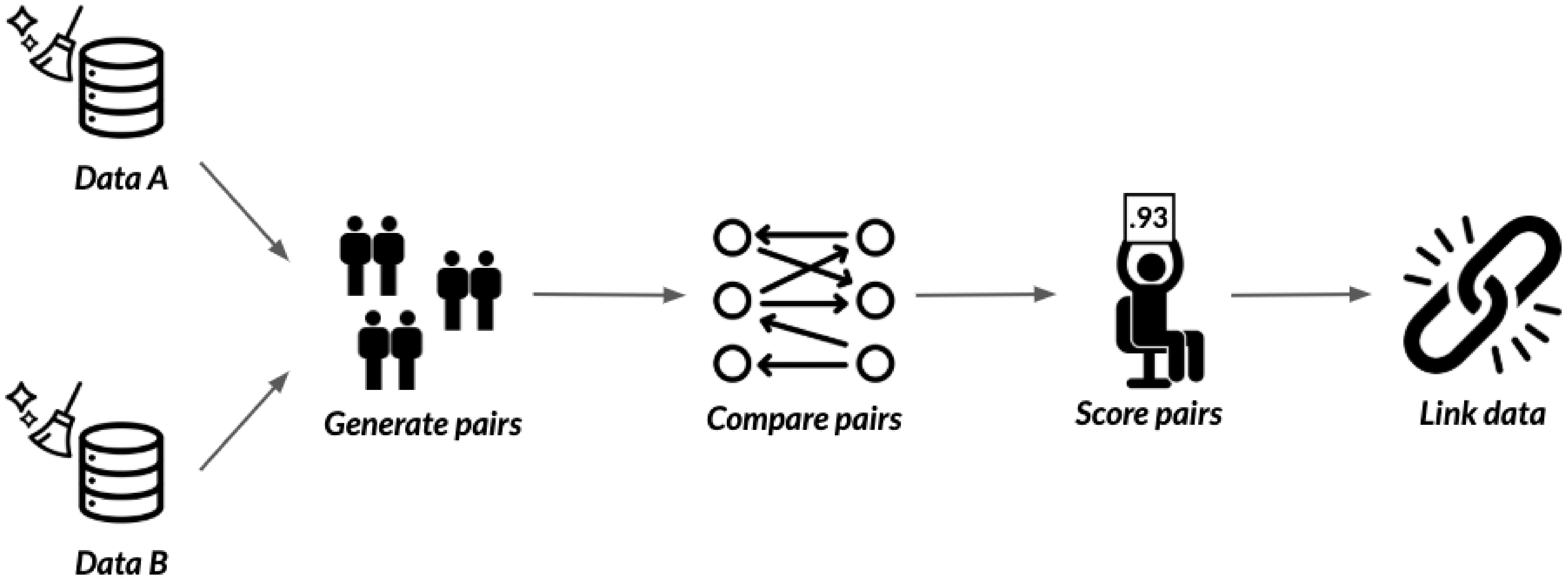
Cross field validation

Summing across rows
Validating age
...

Completeness

Finding missing data
Treating them
...

Chapter 4: Record Linkage



Expand and build upon your new skills

- Categorical Data
  - Categorical Data in the Tidyverse
- Text Data
  - String Manipulation with stringr in R
  - Intermediate Regular Expressions in R
- Writing Clean Code
  - Defensive R Programming

Congratulations!

CLEANING DATA IN R