Datenbearbeitung mit dplyr in R

Datenpfad
datenpfad.ch

In diesem Spickzettel verwendete Datensätze

airbnb_angebot				
airbnb_id	**stadt**	**land**	**zimmer**	**jahr_gelistet**
1	Wien	Österreich	5	2018
2	Luzern	Schweiz	2	2017
3	München	Deutschland	4	2022
4	Berlin	Deutschland	3	2021

gastgeber_angebot			
gastgeber_id	**name**	**airbnb_id**	**anz_bewertung**
1	Anna Gruber	1	34
2	Reto Meier	2	55
3	Michael Weber	3	7
4	Emma Schmidt	5	12

Operatoren

R verfügt über eine Vielzahl an Operatoren. Mit arithmetischen Operatoren können Sie Rechenoperationen wie Addition und Multiplikation durchführen. Relationale Operatoren werden verwendet, um Werte miteinander zu vergleichen. Logische Operatoren werden für boolesche Operationen verwendet.

Arithmetische Operatoren

```
a + b     # Summieren zweier Variablen
a – b     # Subtrahieren zweier Variablen
a * b     # Multiplizieren zweier Variablen
a / b     # Dividieren zweier Variablen
a ^ b     # Potenzieren einer Variablen
a%%b      # Rest einer Variablen
a%/%b     # Ganzzahlige Division von Variablen
```

Relationale Operatoren

```
a == b    # Test auf Gleichheit
a != b    # Test auf Ungleichheit
a > b     # Test auf grösser als
a < b     # Test auf weniger als
a >= b    # Test auf grösser oder gleich
a <= b    # Test auf kleiner oder gleich
```

Logische Operatoren

```
!    # Logisch NICHT (NOT)
&    # Elementweise logisches UND (AND)
&&   # Logisch UND (AND)
|    # Elementweise logisches ODER (OR)
||   # Logisch ODER (OR)
```

Zuweisungsoperator

```
x <- 1    # Weist x eine Variable zu
```

Andere Operatoren

```
%in%    # Identifiziert, ob ein Element zu einem Vektor gehört
$       # Ermöglicht den Zugriff auf Objekte innerhalb eines Objekts
%>%     # Pipe, dient zur Übergabe von Objekten an Funktionen
```

Daten umwandeln

Grundlegende Spaltenoperationen

```
# Auswahl einer oder mehrerer Spalten mit select()
airbnb_angebot %>%
 select(airbnb_id, stadt)

# Spalten anhand von Startzeichen auswählen
airbnb_angebot %>%
 select(starts_with("s"))

# Spalten anhand von Endzeichen auswählen
airbnb_angebot %>%
 select(ends_with("t"))

# Alle Spalten ausser einer auswählen (z.B. airbnb_id)
airbnb_angebot %>%
 select(-airbnb_id)

# Alle Spalten innerhalb eines Bereichs auswählen
airbnb_angebot %>%
 select(land:jahr_gelistet)

# Spaltenwerte als Vektor extrahieren (nach Name oder Index)
airbnb_angebot %>%
 pull(jahr_gelistet)

# Spalten neu anordnen mit relocate()
airbnb_angebot %>%
 relocate(stadt, land)

# Spalte an die letzte Position verschieben
airbnb_angebot %>%
 relocate(stadt, .after = last_col())

# Spalte umbenennen mit rename()
airbnb_angebot %>%
 rename(jahr = jahr_gelistet)

# Spalten wählen, die regulärem Ausdruck entsprechen
airbnb_angebot %>%
 select(matches("(.a.)|(a.)"))
```

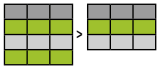
Neue Spalten erstellen

```
# Spalte zeit_am_markt erstellen, welche die Differenz von aktuellem Jahr und jahr_gelistet enthält
airbnb_angebot %>%
 mutate(zeit_am_markt = 2024 - jahr_gelistet)

# Spalte «adresse_vollständig» durch Kombination von Stadt und Land erstellen
airbnb_angebot %>%
 transmute(adr_vollständig = paste(stadt, land))

# Anzahl der Beobachtungen für eine Spalte addieren (z.B. Anzahl der Inserate pro Land)
airbnb_angebot %>%
 add_count(land)
```

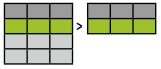
Arbeiten mit Zeilen

```
# Zeilen nach einer Bedingung filtern (z.B. Land)
airbnb_angebot %>%
 filter(land == "Schweiz")
```

```
# Zeilen nach zwei ODER mehr Bedingungen filtern
airbnb_angebot %>%
 filter(land == "Schweiz" | zimmer > 3)

# Zeilen nach zwei UND mehr Bedingungen filtern
airbnb_angebot %>%
 filter(land == "Schweiz" & zimmer < 3)

# Filtern, indem geprüft wird, ob ein Wert in einer
anderen Gruppe von Werten vorhanden ist
airbnb_angebot %>%
 filter(land %in% c("Österreich", "Deutschland"))

# Zeilen nach Zeilenindex filtern (z.B. erste 2 Zeilen)
airbnb_angebot %>%
 slice(1:2)
```

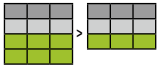


```
# Zeilen mit den höchsten Werten auswählen
airbnb_angebot %>%
 slice_max(zimmer, prop = 0.35)
```



```
# Zeilen nach Spaltenwerten aufsteigend sortieren
airbnb_angebot %>%
 arrange(zimmer)
```

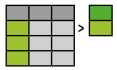
```
# Zeilen nach Spaltenwerten absteigend sortieren
airbnb_angebot %>%
 arrange(desc(stadt))

# Doppelte Zeilen im gesamten Datensatz entfernen
airbnb_angebot %>%
 distinct()
```



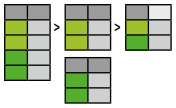
```
# Eindeutige Werte in der Spalte Land finden
airbnb_angebot %>%
 distinct(land)

# Zeilen basierend auf den Top-n-Werten einer Spalte
auswählen
airbnb_angebot %>%
 top_n(3, zimmer)
```

Datenaggregation

```
# Gruppen innerhalb einer Spalte zählen
airbnb_angebot %>%
 count(stadt)

# Gruppen innerhalb einer Spalte zählen und sortiert
zurückgeben
airbnb_angebot %>%
 count(land, sort = TRUE)

# Summenwerte einer Spalte zurückgeben (z.B. Gesamt-
zahl der Zimmer)
airbnb_angebot %>%
 summarise(total_zimmer = sum(zimmer))
```



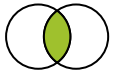
```
# Rückgabe des Mittelwertes einer Spalte (z.B. durch-
schnittliche Anzahl der Zimmer)
airbnb_angebot %>%
 summarise(avg_zimmer = mean(zimmer))

# Rückgabe einer benutzerdefinierten zusammen-
fassenden Statistik
airbnb_angebot %>%
 summarise(avg_gelistet = 2024 - mean(jahr_gelistet))

# Nach einer Variablen gruppieren und die Anzahl pro
Gruppe zurückgeben
airbnb_angebot %>%
 group_by(land) %>%
 summarise(n = n())
```



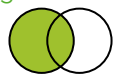
```
# Nach einer Variablen gruppieren und den Mittelwert
pro Gruppe zurückgeben
airbnb_angebot %>%
 group_by(stadt) %>%
 summarise(avg_zimmer = mean(zimmer))
```

Tabellen verbinden

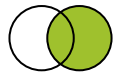
```
# Inner Join: Gibt nur Datensätze zurück, bei denen ein
Verbindungsfeld in beiden Tabellen übereinstimmt
airbnb_angebot %>%
inner_join(gastgeber_angebot, by = "airbnb_id")
```



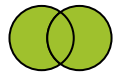
```
# Left Join: Gibt die Zeilen der linken Tabelle und die feh-
lenden Werte für alle Spalten der rechten Tabelle zurück,
für die das Verbindungsfeld keine Entsprechung findet
airbnb_angebot %>%
left_join(gastgeber_angebot, by = "airbnb_id")
```



```
# Right Join: Gibt die Zeilen der rechten Tabelle und die
fehlenden Werte für alle Spalten der linken Tabelle zu-
rück, für die das Verbindungsfeld keine Entsprechung
findet
airbnb_angebot %>%
right_join(gastgeber_angebot, by = "airbnb_id")
```



```
# Full Join: Gibt alle Datensätze aus beiden Tabellen zu-
rück, unabhängig davon, ob es eine Übereinstimmung
für das Verbindungsfeld gibt
airbnb_angebot %>%
full_join(gastgeber_angebot, by = "airbnb_id")
```

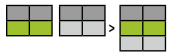


```
# Anti Join: Gibt Datensätze aus der ersten Tabelle zurück
und schliesst übereinstimmende Werte aus der zweiten
Tabelle aus
airbnb_angebot %>%
anti_join(gastgeber_angebot, by = "airbnb_id")
```

Tabellen kombinieren

```
# Tabelle rechts (horizontal) an eine andere anhängen
bind_cols(df_1, df_2)
```

```
# Tabelle unten (senkrecht) an eine andere anhängen
bind_rows(df_1, df_2)
```



```
# Kombinieren von Zeilen, die in beiden Tabellen vor-
handen sind, und Löschen von Duplikaten
union(df_1, df_2)

# Identische Spalten in beiden Tabellen finden
intersect(df_1, df_2)

# Zeilen finden, die in einer anderen Tabelle nicht vor-
handen sind
setdiff(df_1, df_2)
```