



Joins I

Intro to Joins



- Read Chapter 10
- Usually, Multiple Tables of Data are Used in Analysis
- Data Must Be Merged Prior to Analysis
- Requires Attention to Detail
- Fundamental Concept in Data Science

Sample Data



- Transaction Data

| Name | Purchase | Day | Month | ID |
|--------|----------|-----|-------|------|
| Harry | 6.99 | 1 | 3 | 1001 |
| Harry | 12.99 | 2 | 3 | 1023 |
| Billy | 8.99 | 2 | 3 | 1027 |
| Fred | 14.99 | 2 | 3 | 1039 |
| Billy | 13.99 | 3 | 3 | 1042 |
| George | 12.99 | 3 | 3 | 1043 |
| George | 12.99 | 3 | 3 | 1048 |
| George | 9.99 | 3 | 3 | 1051 |
| Harry | 10.99 | 4 | 3 | 1063 |
| Billy | 9.99 | 4 | 3 | 1072 |

- Sales Data

| Day | Month | Sales |
|-----|-------|-------|
| 1 | 3 | 45.05 |
| 2 | 3 | 43.83 |
| 3 | 3 | 53.71 |
| 4 | 3 | 42.92 |

Sample Data



- Survey Data

| Name | Age | Overall | Service | Food |
|--------|-----|---------|---------|------|
| Harry | 35 | 3 | 4 | 5 |
| Billy | 43 | 5 | 3 | 4 |
| George | 61 | 2 | 1 | 1 |
| Merri | 52 | 5 | 5 | 5 |

- Order Data (Preview)

| ID | Coupon | GiftCard | Item |
|------|--------|----------|---------|
| 1001 | 1 | 0 | Veggie |
| 1002 | 0 | 0 | Pork |
| 1003 | 1 | 0 | Veggie |
| 1004 | 1 | 0 | Pork |
| 1005 | 1 | 0 | Poultry |
| 1006 | 0 | 0 | Poultry |
| 1007 | 1 | 0 | Seafood |
| 1008 | 1 | 0 | Seafood |
| 1009 | 1 | 1 | Beef |
| 1010 | 0 | 1 | Pork |

Sample Data



- Scenario: Restaurant Owner
- Why Connect the Data?
- What Questions Can We Answer?
- What Insights Might We Learn?

Keys



- The Variable(s) That Uniquely Identify an Observation
- Two Types:
 - Primary = Uniquely Identifies an Observation in Its Own Table
 - Foreign = Uniquely Identifies an Observation in Another Table

Keys



- Identifying the Primary Keys
 - ID is a Primary Key for Both Transaction and Order Data
 - Day + Month is a Primary Key for Sales Data
 - Name is a Primary Key for Survey Data

Keys



- Verifying the Primary Keys

```
Transaction %>%  
  count(ID) %>%  
  filter(n>1)
```

```
## # A tibble: 0 x 2  
## # ... with 2 variables: ID <int>, n <int>
```

```
Transaction %>%  
  count(Name) %>%  
  filter(n>1)
```

```
## # A tibble: 3 x 2  
##   Name      n  
##   <chr> <int>  
## 1 Billy     3  
## 2 George    3  
## 3 Harry     3
```

```
identical(unique(Transaction$ID),Transaction$ID)
```

```
## [1] TRUE
```

```
identical(unique(Transaction$Name),Transaction$Name)
```

```
## [1] FALSE
```


Keys



- Verifying the Primary Keys

```
Sales %>%  
  count (Month)
```

```
## # A tibble: 1 x 2  
##   Month      n  
##   <int> <int>  
## 1       3     4
```

```
Sales %>%  
  count (Day,Month)
```

```
## # A tibble: 4 x 3  
##   Day Month      n  
##   <int> <int> <int>  
## 1     1     3     1  
## 2     2     3     1  
## 3     3     3     1  
## 4     4     3     1
```

Mutating Joins



- Inner Joins
 - Matches Observations When Their Keys are Equal
 - Equivalent to `> merge(x,y)`
 - Example: Survey + Transaction

```
unique(Survey$Name)
```

```
## [1] "Harry" "Billy" "George" "Merri"
```

```
unique(Transaction$Name)
```

```
## [1] "Harry" "Billy" "Fred" "George"
```

Mutating Joins



- Inner Joins
- Example: Survey + Transaction

```
Survey %>%  
  count (Name)
```

```
## # A tibble: 4 x 2  
##   Name      n  
##   <chr> <int>  
## 1 Billy     1  
## 2 George    1  
## 3 Harry     1  
## 4 Merri     1
```

```
Transaction %>%  
  count (Name)
```

```
## # A tibble: 4 x 2  
##   Name      n  
##   <chr> <int>  
## 1 Billy     3  
## 2 Fred      1  
## 3 George    3  
## 4 Harry     3
```

Mutating Joins

- Inner Joins

- Example: Survey + Transaction

```
SurveyTrans=inner_join(Survey,Transaction,by="Name")  
SurveyTrans
```

```
## # A tibble: 9 x 9  
##   Name      Age Overall Service  Food Purchase   Day Month   ID  
##   <chr> <int>   <int>   <int> <int>   <dbl> <int> <int> <int>  
## 1 Harry    35      3      4      5     6.99      1      3  1001  
## 2 Harry    35      3      4      5    13.0      2      3  1023  
## 3 Harry    35      3      4      5    11.0      4      3  1063  
## 4 Billy    43      5      3      4     8.99      2      3  1027  
## 5 Billy    43      5      3      4    14.0      3      3  1042  
## 6 Billy    43      5      3      4     9.99      4      3  1072  
## 7 George   61      2      1      1    13.0      3      3  1043  
## 8 George   61      2      1      1    13.0      3      3  1048  
## 9 George   61      2      1      1     9.99      3      3  1051
```



Mutating Joins



- Outer Joins
 - Left-Join
 - Keeps All Observations in Left Dataset
 - Equivalent to

```
> merge(x,y,all.x=TRUE)
```

Mutating Joins

- Outer Joins
 - Left-Join
- Example: Survey + Trans.



```
SurveyTrans2=left_join(Survey,Transaction,by="Name")  
SurveyTrans2
```

```
## # A tibble: 10 x 9  
##   Name      Age Overall Service  Food Purchase  Day Month   ID  
##   <chr>  <int>   <int>   <int> <int>   <dbl> <int> <int> <int>  
## 1 Harry    35      3      4      5     6.99      1      3 1001  
## 2 Harry    35      3      4      5    13.0      2      3 1023  
## 3 Harry    35      3      4      5    11.0      4      3 1063  
## 4 Billy    43      5      3      4     8.99      2      3 1027  
## 5 Billy    43      5      3      4    14.0      3      3 1042  
## 6 Billy    43      5      3      4     9.99      4      3 1072  
## 7 George   61      2      1      1    13.0      3      3 1043  
## 8 George   61      2      1      1    13.0      3      3 1048  
## 9 George   61      2      1      1     9.99      3      3 1051  
## 10 Merri   52      5      5      5     NA      NA      NA  NA
```

Mutating Joins



- Outer Joins
 - Right-Join
 - Keeps All Observations in Right Dataset
 - Equivalent to

```
> merge(x,y,all.y=TRUE)
```

Mutating Joins

- Outer Joins
 - Right-Join
 - Example: Survey + Trans.



```
SurveyTrans3=right_join(Survey,Transaction,by="Name")
SurveyTrans3
```

```
## # A tibble: 10 x 9
##   Name      Age Overall Service  Food Purchase  Day Month   ID
##   <chr>  <int>   <int>   <int> <int>   <dbl> <int> <int> <int>
## 1 Harry    35      3      4      5     6.99     1     3  1001
## 2 Harry    35      3      4      5    13.0     2     3  1023
## 3 Billy    43      5      3      4     8.99     2     3  1027
## 4 Fred     NA     NA     NA     NA    15.0     2     3  1039
## 5 Billy    43      5      3      4    14.0     3     3  1042
## 6 George   61      2      1      1    13.0     3     3  1043
## 7 George   61      2      1      1    13.0     3     3  1048
## 8 George   61      2      1      1     9.99     3     3  1051
## 9 Harry    35      3      4      5    11.0     4     3  1063
## 10 Billy   43      5      3      4     9.99     4     3  1072
```


Mutating Joins



- Outer Joins
 - Full-Join
 - Keeps All Observations in Both Datasets
 - Equivalent to

```
> merge(x,y,all.x=TRUE,all.y=TRUE)
```

Mutating Joins

- Outer Joins
 - Full-Join
 - Example: Survey + Trans.



```
SurveyTrans4=full_join(Survey,Transaction,by="Name")
SurveyTrans4
```

```
## # A tibble: 11 x 9
##   Name      Age Overall Service  Food Purchase  Day Month   ID
##   <chr>  <int>   <int>   <int> <int>   <dbl> <int> <int> <int>
## 1 Harry    35       3       4     5     6.99     1     3  1001
## 2 Harry    35       3       4     5    13.0     2     3  1023
## 3 Harry    35       3       4     5    11.0     4     3  1063
## 4 Billy    43       5       3     4     8.99     2     3  1027
## 5 Billy    43       5       3     4    14.0     3     3  1042
## 6 Billy    43       5       3     4     9.99     4     3  1072
## 7 George   61       2       1     1    13.0     3     3  1043
## 8 George   61       2       1     1    13.0     3     3  1048
## 9 George   61       2       1     1     9.99     3     3  1051
## 10 Merri   52       5       5     5     NA      NA     NA   NA
## 11 Fred    NA      NA      NA     NA    15.0     2     3  1039
```

Mutating Joins



- Duplicate Keys
 - All Examples Illustrate the Scenario When Keys Repeat
 - One to Many Relationship
 - “Usually” Indicates Error
 - Identify Your Most Important Dataset
 - Summarize then Merge

Mutating Joins

- Duplicate Keys
- Example

```
SurveyTrans5 = Transaction %>%  
  group_by(Name) %>%  
  summarize(n=n(), Avg.Purchase=mean(Purchase)) %>%  
  inner_join(Survey, by="Name")
```

SurveyTrans5

```
## # A tibble: 3 x 7  
##   Name      n Avg.Purchase   Age Overall Service  Food  
##   <chr> <int>      <dbl> <int>   <int>   <int> <int>  
## 1 Billy     3      11.0    43      5       3     4  
## 2 George    3      12.0    61      2       1     1  
## 3 Harry     3      10.3    35      3       4     5
```



Mutating Joins



- Defining the Key Columns
 - Default: Uses All Variables that Appear in Both Tables

```
SalesTrans = inner_join(Sales, Transaction)
```

```
## Joining, by = c("Day", "Month")
```

```
SalesTrans
```

```
## # A tibble: 10 x 6
##   Day Month Sales Name Purchase ID
##   <int> <int> <dbl> <chr>    <dbl> <int>
## 1     1     3  50.7 Harry     6.99  1001
## 2     2     3  49.9 Harry    13.0  1023
## 3     2     3  49.9 Billy     8.99  1027
## 4     2     3  49.9 Fred    15.0  1039
## 5     3     3  49.9 Billy    14.0  1042
## 6     3     3  49.9 George   13.0  1043
## 7     3     3  49.9 George   13.0  1048
## 8     3     3  49.9 George    9.99  1051
## 9     4     3  38.4 Harry    11.0  1063
## 10    4     3  38.4 Billy     9.99  1072
```

Mutating Joins

- Defining the Key Columns
 - Keys Based on Multiple Variables
 - Key Names Can Be Different



```
Sales2 = Sales %>%
  rename(D=Day,M=Month)
Trans2 = Transaction %>%
  group_by(Day,Month,Name) %>%
  summarize(sumPurchase=sum(Purchase)) %>%
  ungroup()

SalesTrans2=left_join(Trans2, Sales2,
  by=c("Day"="D", "Month"="M")) %>%
  transmute(Day=Day,Month=Month,Name=Name,
    perSales=sumPurchase/Sales)
```

Mutating Joins



- Defining the Key Columns

| Day | Month | Name | perSales |
|-----|-------|--------|----------|
| 1 | 3 | Harry | 0.14 |
| 2 | 3 | Billy | 0.18 |
| 2 | 3 | Fred | 0.30 |
| 2 | 3 | Harry | 0.26 |
| 3 | 3 | Billy | 0.28 |
| 3 | 3 | George | 0.72 |
| 4 | 3 | Billy | 0.26 |
| 4 | 3 | Harry | 0.29 |

Filtering Joins



- Semi-Join
 - `> semi_join(x,y)`
 - Keeps All Observations in Left Dataset That Have a Match in Right Dataset
 - Primary Data = Left
 - Scenario: Want All Order Data Only For Select Customers

Filtering Joins



- Semi-Join

```
semi_join(Order, Transaction)
```

```
## Joining, by = "ID"
```

```
## # A tibble: 9 x 4
```

```
##       ID Coupon GiftCard Item
##   <int> <int>    <int> <chr>
## 1  1001      1        0 Poultry
## 2  1023      1        0 Beef
## 3  1027      0        0 Beef
## 4  1039      0        0 Poultry
## 5  1042      1        1 Beef
## 6  1043      0        0 Poultry
## 7  1048      0        0 Poultry
## 8  1051      0        0 Veggie
## 9  1063      0        0 Pork
```

Filtering Joins



- Anti-Join
 - `> anti_join(x,y)`
 - Drops All Observations in Left Dataset That Have a Match in Right Dataset
 - Primary Data = Left
 - Scenario: Want All Order Data Except For Select Customers

Filtering Joins



- Anti-Join

```
anti_join(Order, Transaction)
```

```
## Joining, by = "ID"
```

```
## # A tibble: 54 x 4
```

```
##       ID Coupon GiftCard Item
##   <int> <int>    <int> <chr>
## 1  1002      0      0 Poultry
## 2  1003      1      0 Seafood
## 3  1004      1      0 Seafood
## 4  1005      1      1 Beef
## 5  1006      0      1 Pork
## 6  1007      0      0 Beef
## 7  1008      0      0 Pork
## 8  1009      1      0 Poultry
## 9  1010      1      0 Pork
## 10 1011      1      1 Veggie
## # ... with 44 more rows
```

Closing



Disperse
and Make
Reasonable
Decisions