Excel tasks using R

Accomplishing common Excel data-related tasks using R

Stuart Ward
November 17, 2015

Agenda

- Context and caveats
- Computer and configurations
- Load data into R
- Look at the data
- Pivot Tables
- Pivot Charts
- VLOOKUP
- Load data into Excel

Context and caveats

- Motivation: moving from Excel to R
  - Me
  - My clients

- Vanilla

- Presentation > Conversation > Ongoing Dialogue

Computer and configurations

- Computer
  - Windows 8.1 Pro 64-bit
  - Intel Quad Core i7-3840QM 2.8GHz
  - 16 GB RAM

- R
  - R version 3.2.1 (2015-06-08) – “World-Famous Astronaut” (64-bit)
  - Revolution R Open 3.2.1 (Multithreaded BLAS/LAPACK libraries detected. Using 4 cores for math algorithms.)

Load data into R

- **From Excel**
  - **readxl package (last commit: 2015-04-23)**
    - Hadley Wickham
    - GitHub
    - Works with both xls and xlsx
    - No external dependencies
    - Tabular data only (can select specific worksheet)
    - df <- read_excel("flightdata.xls", sheet = "data")

  - **gdata package (last published: 2015-07-04)**
    - Works with both xls and xlsx files
    - Gregory R. Warnes (8 other named authors), and others
    - read.xls() – utilizes Perl; translates the named Excel file into a temporary CSV or TAB file
    - System Requirements: perl >= 5.10.0
    - df <- read.xls("flightdata.xls", perl="PathToPerl.ExeFileOnYourComputer")

Load data into R

- **From Excel**
  - **xlsx package (last published 2014-08-02)**
    - Adrian A. Dragulescu
    - Utilizes a java library from the Apache POI project
    - Read/write xlsx files
    - Control appearance (formats, fonts, colors, borders) writing to Excel
    - df <- read.xlsx2(“flightdata.xlsx", sheetName = "Sheet1")

  - **XLConnect package (last published: 2015-03-01)**
    - Mirai Solutions
    - “Comprehensive and cross-platform R package for manipulating Microsoft Excel files from within R”
    - Produce formatted reports, including graphics
    - Import specific worksheets, named ranges, individual cells
    - df <- readWorksheetFromFile(“flightdata.xls", sheet=1, startRow = 4, endCol = 2)

Load data into R

- **Use Case Scenario**
  - Government reports in Excel
    - Three workbooks; 50 sheets per workbook

Section 1 System & Claims | QR-S1 | QR-S2 | Section 2 Member Services | QR-M1 | QR-M2 | QR-M3

Measure	Claim Type			
	UB-04 (Institutional) (837 I)		CMS 1500 (Professional) (837 P)	
	In-Network	Out-Of-Network	In-Network	Out-Of-Network
Total Submitted Dollars (not paid amount)	0			
Claims Received				
Electronic	0	0	0	0
Paper	0	0	0	0
Total (calculated)	0	0	0	0
Clean Claims Adjudicated				
Paid On Time	0	0	0	0
Paid Late	0	0	0	0
Denied	0	0	0	0
Denial Rate (calculated)	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!
Claims Paid With Interest				
Total Number of Claims Paid With Interest		0		0
Total Dollar Amount of Interest Paid		$0.00		$0.00
Claims Lag				
Average number of days between the last date of service on claim and Health Plan's receipt of claim from provider.	0	0	0	0
Average number of days between the receipt date on claim and the adjudication date.	0	0	0	0
Average number of days from the adjudication date to payment (remittance advice) date.	0	0	0	0

Item No.	Data Description	Third Previous Period	Second Previous Period	Previous Period	Current Period Submission
1	Number of Member Calls Received	0	0	0	0
2	Number of Member Calls Answered	0	0	0	0
3	Number of Member Calls Answered Live Within 30 Seconds	0	0	0	0
4	Performance Measure #1: Pct in 30 Seconds				
5	Number of Abandoned Calls	0	0	0	0
6	Performance Measure #2: Pct Abandoned				
7	Number of Calls Received After Hours	0	0	0	0
	Mark an 'X' if updated from previous report version				
	Date of update resubmission				

    - Loop through data and build data frames
    - Analyze, graph, summary
    - Save to database

Load data into R

- **'Large' data (Issue #1 – time to load data)**
  - **read.csv; read.table**
  - **fread(); data.table package (last published: 2015-09-19)**
  - **readr package (last published: 2015-10-22)**

  - ***Compared to base functions, readr functions:**
    - Use a consistent naming scheme (e.g. col_names and col_types not header and colClasses)
    - Are much faster (up to 10x faster)
    - Have a helpful progress bar if loading is going to take a while
    - All functions work the same way regardless of locale. To override the US-centric defaults, use locale()

  - ***Compared to fread(), readr:**
    - Is slower (currently ~1.2-2x slower. If you want absolutely the best performance, use data.table::fread() .
    - Readr has a slightly more sophisticated parser, recognizing both doubled ("""") and backslash escapes ("\""). Readr allows you to read factors and date times directly from disk.
    - fread() saves you work by automatically guessing the delimiter, whether or not the file has a header, how many lines to skip by default and more. Readr forces you to supply these parameters.
    - The underlying designs are quite different. Readr is designed to be general, and dealing with new types of rectangular data just requires implementing a new tokenizer. fread() is designed to be as fast as possible. fread() is pure C, readr is C++ (and Rcpp).

*From Hadley Wickham

Load data into R

- **'Large' data (Issue #1 – time to load data)**
  - **Use Case Scenario**
    - Flight delay data – 5.8 million rows, 31 columns (500 MB CSV file on disk)

Loading method	Elapsed Time (sec)	RAM required (GB)
read.csv	58.3	2.8
fread	8.3	1.8
readr	42.6	4.9

    - Progress indicators

```
>
> flightTime_fread <- fread("airlines5mil.csv")
Read 84.0% of 5834763 rows
```

```
>
> flightTime_readr <- read_csv("airlines5mil.csv")
|=========================================================|  72%  419 MB
```

Load data into R

- **'Large' data: Issue #2 – data too large for memory [for R]**
  - **ff package (last published: YYY-MM-DD)**
    - File-based access to datasets that cannot fit into memory
  - **bigmemory package (last published: YYYY-MM-DD)**
    - 'big' family consists of several packages for performing tasks on large datasets
    - bigmemory implements several matrix objects (big.matrix, shared.big.matrix, filebacked.big.matrix)
    - Matrices only contain one type of data; data types dictated by C++, not R
  - **Sample data and transform (to reduce size) on-the-fly during load process?**
  - **Use Case Scenario**
    - 1 million rows; 785 columns; 2 GB CSV file
    - R – loading data exceeded 90% of available RAM; stopped process

Look at the data

- **Visual inspection of the data**

Year	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	UniqueCarrier
1987	10	16	5	1725	1725	735	735	DL
1987	10	6	2	2305	2305	556	556	DL
1987	10	13	2	1925	1925	2106	2106	DL
1987	10	26	1	1925	1925	2106	2106	DL
1987	10	1	4	1105	1105	1225	1225	DL
1987	10	2	5	1105	1105	1225	1225	DL
1987	10	23	5	1105	1105	1225	1225	DL
1987	10	3	6	1505	1505	1750	1750	DL
1987	10	13	2	1505	1505	1750	1750	DL

- **View(flights1987)**

ExcelUsingR.R × flights1987 ×

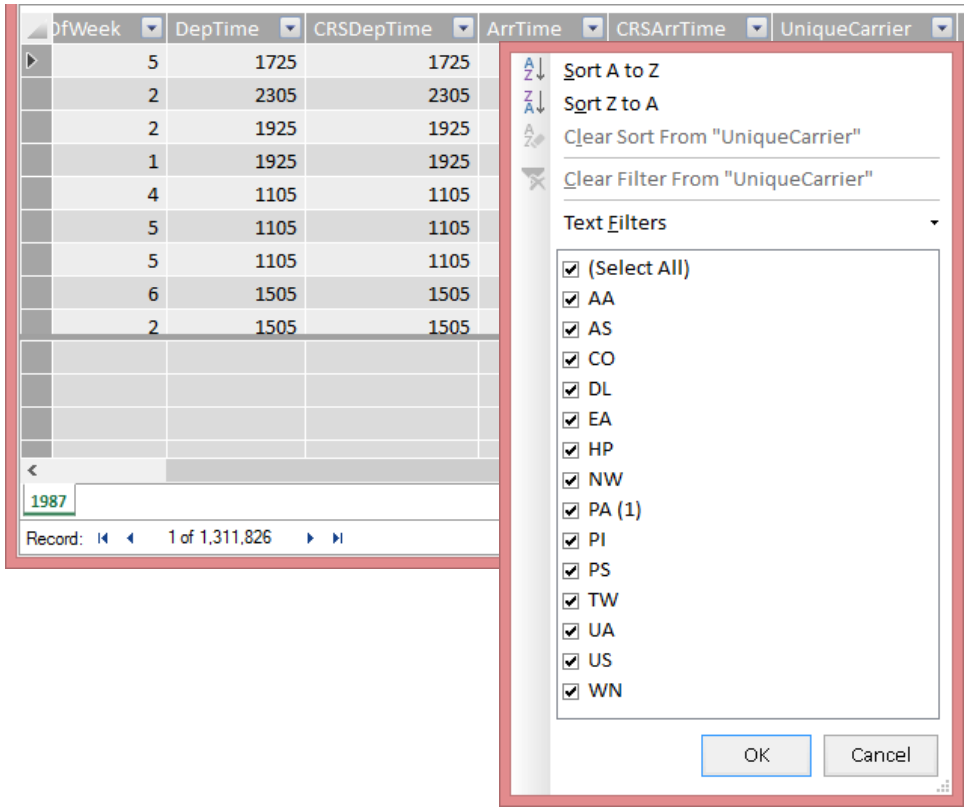
Filter

	Year	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	UniqueCarrier
1	1987	10	14	3	741	730	912	849	PS
2	1987	10	15	4	729	730	903	849	PS
3	1987	10	17	6	741	730	918	849	PS
4	1987	10	18	7	729	730	847	849	PS
5	1987	10	19	1	749	730	922	849	PS

Showing 1 to 6 of 1,311,826 entries

Look at the data

- **View unique values of each column**



- **sort(unique(flights1987$UniqueCarrier))**

```
> sort(unique(flights1987$UniqueCarrier))
 [1] AA     AS     CO     DL     EA     HP     NW     PA (1) PI     PS     TW     UA     US     WN
Levels: AA AS CO DL EA HP NW PA (1) PI PS TW UA US WN
```

Look at the data

- **Filtering and viewing results**

Year	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	UniqueCarrier
1987	10	18	7	1045	1045	1256	1207	NW
1987	10	18	7	820	820	1528	1525	NW
1987	10	18	7	615	615	844	845	NW
1987	10	18	7	900	900	1058	1108	NW
1987	10	18	7	700	700	811	833	NW
1987	10	18	7	1250	1250	1540	1517	NW
1987	10	18	7	700	700	830	839	NW
1987	10	18	7	700	700	835	842	NW
1987	10	18	7	1035	1035	1230	1211	NW

- **NWflights1987 <- subset(flights1987, UniqueCarrier == "NW")**
- **View(NWflights1987)**

ExcelUsingR.R × | flights1987 × | NWflights1987 ×

Filter

	Year	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	UniqueCarrier	Fl
141371	1987	10	1	4	905	905	1110	1100	NW	
141372	1987	10	2	5	905	905	1059	1100	NW	
141373	1987	10	3	6	911	905	1103	1100	NW	
141374	1987	10	4	7	905	905	1045	1100	NW	
141375	1987	10	5	1	905	905	1033	1100	NW	

Showing 1 to 6 of 108,273 entries

Pivot Tables

- **Excel version**

Day of Week

Airline (filtered)

Destination (filtered)

Average Arrival Delay

Average of D								
Row Lab	1	2	3	4	5	6	7	Total
DL	**8**	**21**	**13**	**10**	**9**	**5**	**12**	**11**
EWR	10	22	18	11	10	5	13	13
JFK	8	18	11	10	9	7	12	11
LGA	7	21	12	9	8	5	11	10
UA	**7**	**15**	**9**	**7**	**5**	**2**	**8**	**8**
EWR	10	18	12	8	6	2	9	10
JFK	8	15	13	10	10	10	15	12
LGA	4	11	6	6	2	-2	5	4
AA	**3**	**11**	**6**	**4**	**5**	**3**	**6**	**5**
EWR	3	13	9	6	3	3	6	6
JFK	0	6	5	1	7	5	5	4
LGA	6	13	5	4	5	1	6	6
Total	**6**	**15**	**9**	**7**	**6**	**3**	**8**	**8**

Pivot Tables

- **Utilizing stats package**
  - **aggregate() function**
  - aggregateExample <-

    aggregate(ArrDelay ~ UniqueCarrier + Dest,

    data = data_tdf, FUN=mean)

ExcelUsingR.R × aggregateExample ×

Filter

	UniqueCarrier	Dest	ArrDelay
1	AA	EWR	6.076076
2	DL	EWR	12.493385
3	UA	EWR	9.303463
4	AA	JFK	4.134506
5	DL	JFK	10.679873
6	UA	JFK	11.632780
7	AA	LGA	5.623767
8	DL	LGA	10.328615
9	UA	LGA	4.331837

- **Utilizing dplyr package**
  - **group_by() and summarise() functions**
  - groupSumExample <-

    data_tdf %>%

    group_by(UniqueCarrier) %>%

    summarise(ArrDelay=mean(ArrDelay))

ExcelUsingR.R × groupSumExample ×

Filter

	UniqueCarrier	ArrDelay
1	AA	5.303198
2	DL	10.943682
3	UA	7.567785

Pivot Tables

- **Utilizing reshape2 package**
  - **dcast() function**
  - dcastExample <-

    dcast(data_tdf,

    UniqueCarrier + Dest ~ DayOfWeek,

    value.var = "ArrDelay",

    fun.aggregate = mean)

ExcelUsingR.R × dcastExample ×

Filter

	UniqueCarrier	Dest	1	2	3	4	5	6	7
1	AA	EWR	3.3219178	12.409091	9.246429	5.702055	3.342960	2.487455	5.982877
2	AA	JFK	0.4485294	5.460432	4.768683	1.103806	6.860377	5.155303	5.330986
3	AA	LGA	5.4643963	12.698795	5.297546	4.070588	4.416938	1.038961	6.030612
4	DL	EWR	9.4486486	21.846995	17.352941	10.735751	9.511236	5.425287	12.724324
5	DL	JFK	7.5714286	18.087912	10.736264	10.000000	9.022472	6.873563	12.263736
6	DL	LGA	7.1549296	21.030516	11.443155	8.954148	7.945498	4.836228	10.730673
7	UA	EWR	9.6366279	17.175074	11.178161	8.132394	6.365559	2.394928	8.912226
8	UA	JFK	8.1798561	14.785185	13.175182	10.135135	10.240602	9.947761	15.079710
9	UA	LGA	4.0851648	10.248555	5.505682	5.521164	1.582386	-1.848101	4.625731

+ margins = c("UniqueCarrier","Dest")

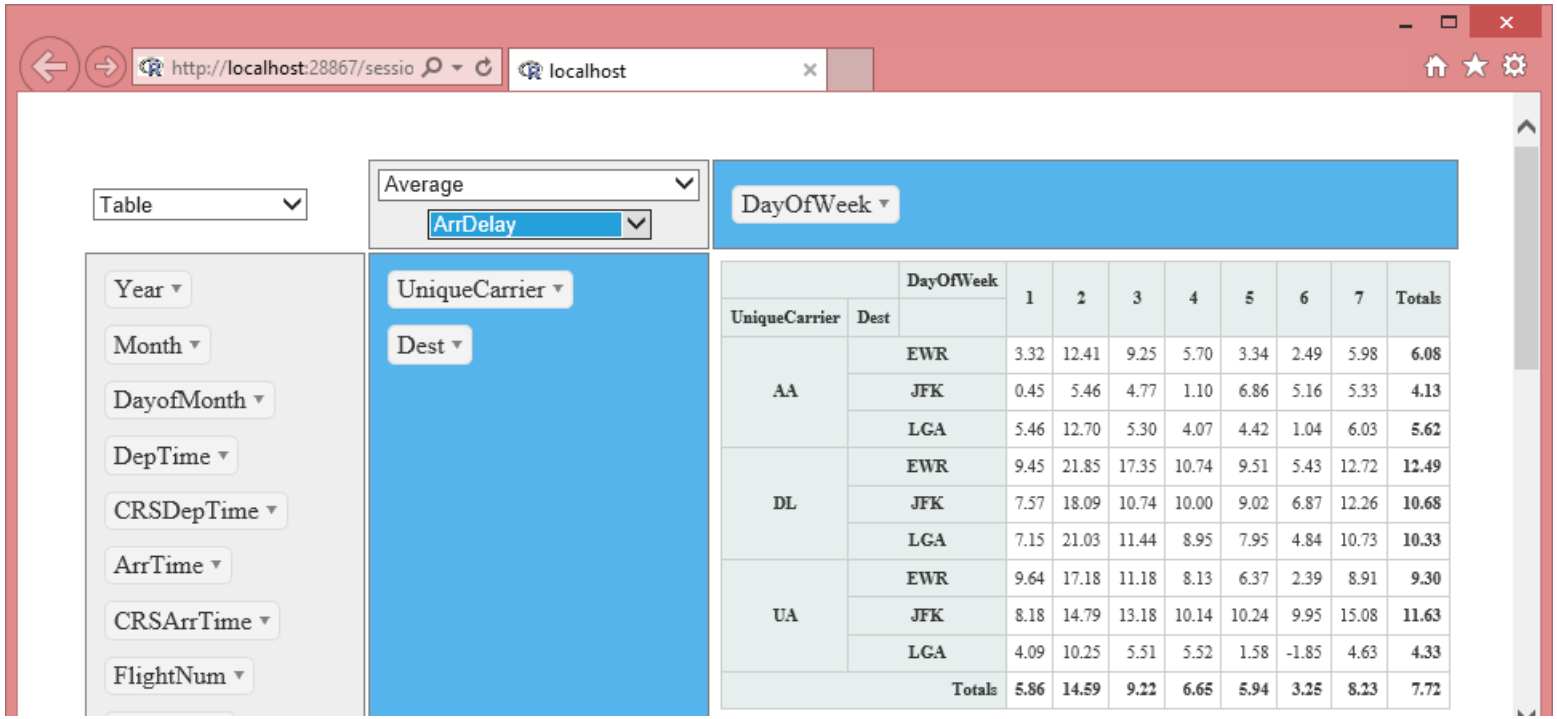
ExcelUsingR.R × dcastExample ×

Filter

	UniqueCarrier	Dest	1	2	3	4	5	6	7
1	AA	EWR	3.3219178	12.409091	9.246429	5.702055	3.342960	2.487455	5.982877
2	AA	JFK	0.4485294	5.460432	4.768683	1.103806	6.860377	5.155303	5.330986
3	AA	LGA	5.4643963	12.698795	5.297546	4.070588	4.416938	1.038961	6.030612
4	AA	(all)	3.2209696	10.360491	6.376550	3.656895	4.829211	2.790834	5.786207
5	DL	EWR	9.4486486	21.846995	17.352941	10.735751	9.511236	5.425287	12.724324
6	DL	JFK	7.5714286	18.087912	10.736264	10.000000	9.022472	6.873563	12.263736
7	DL	LGA	7.1549296	21.030516	11.443155	8.954148	7.945498	4.836228	10.730673
8	DL	(all)	7.8133903	20.861429	12.911142	9.545822	8.489115	5.257530	11.481536
9	UA	EWR	9.6366279	17.175074	11.178161	8.132394	6.365559	2.394928	8.912226
10	UA	JFK	8.1798561	14.785185	13.175182	10.135135	10.240602	9.947761	15.079710
11	UA	LGA	4.0851648	10.248555	5.505682	5.521164	1.582386	-1.848101	4.625731
12	UA	(all)	7.0118064	13.850856	9.119474	7.348468	4.933824	1.942149	8.142678
13	(all)	(all)	5.8624795	14.588235	9.224414	6.652909	5.936703	3.246765	8.232310

Pivot Tables

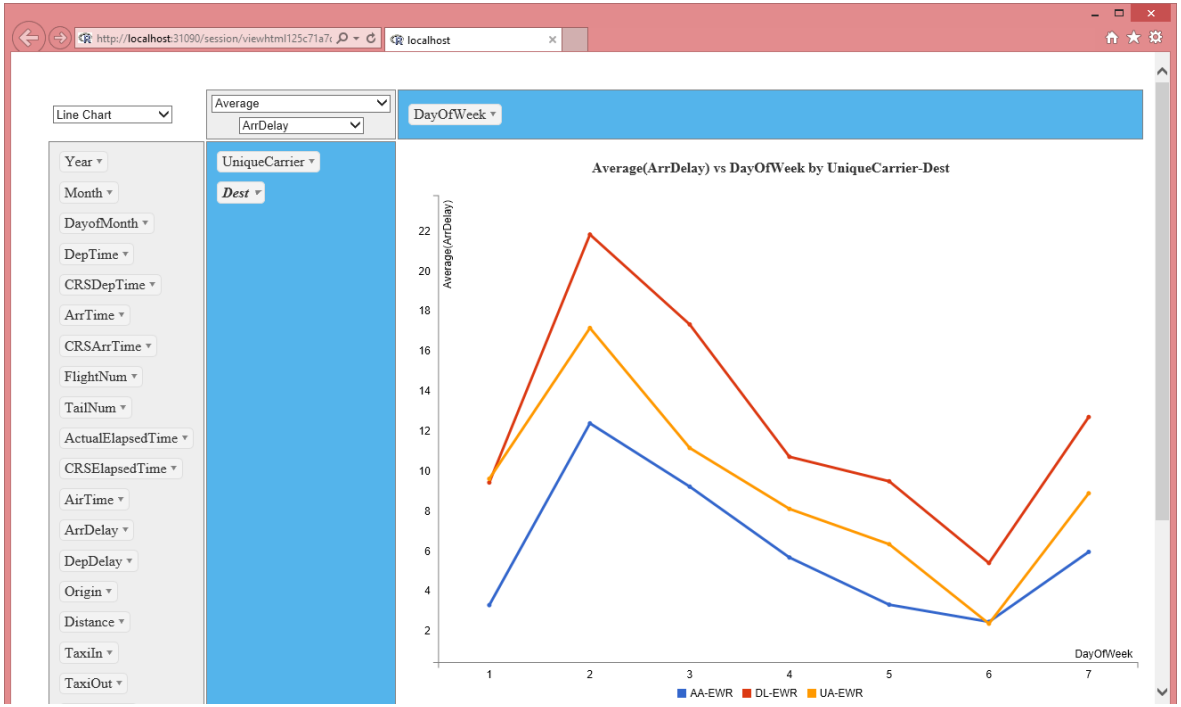
- **Utilizing rpivotTable package**
  - devtools::install_github(c("ramnathv/htmlwidgets", "smartinsightsfromdata/rpivotTable"))
  - library(rpivotTable)
  - rpivotTable(data_tdf)
  - rpivotTable(data = data_tdf, rows=c("UniqueCarrier","Dest"), col="DayOfWeek",
    aggregatorName="Average", vals="ArrDelay", rendererName="Table")



UniqueCarrier	Dest	DayOfWeek 1	2	3	4	5	6	7	Totals
AA	EWR	3.32	12.41	9.25	5.70	3.34	2.49	5.98	**6.08**
	JFK	0.45	5.46	4.77	1.10	6.86	5.16	5.33	**4.13**
	LGA	5.46	12.70	5.30	4.07	4.42	1.04	6.03	**5.62**
DL	EWR	9.45	21.85	17.35	10.74	9.51	5.43	12.72	**12.49**
	JFK	7.57	18.09	10.74	10.00	9.02	6.87	12.26	**10.68**
	LGA	7.15	21.03	11.44	8.95	7.95	4.84	10.73	**10.33**
UA	EWR	9.64	17.18	11.18	8.13	6.37	2.39	8.91	**9.30**
	JFK	8.18	14.79	13.18	10.14	10.24	9.95	15.08	**11.63**
	LGA	4.09	10.25	5.51	5.52	1.58	-1.85	4.63	**4.33**
	Totals	**5.86**	**14.59**	**9.22**	**6.65**	**5.94**	**3.25**	**8.23**	**7.72**

Pivot Charts

- **Utilizing rpivotTable package**



- **Utilizing ggplot2**
  - **?**
- **Utilizing ggvis**
  - **?**

VLOOKUP

- **merge()**
  - airportCodes <- read.csv("AirportCodes.csv")

  - names(airportCodes)[names(airportCodes)=="AirportCode"] <- "Dest"
  - usingMerge <- merge(data_tdfAC, airportCodes)
  - usingMerge <- merge(data_tdfAC, airportCodes, by.x="Dest", by.y="AirportCode")

  - **Merge scenarios**

Scenario	SQL Equiv.	Merge Parameters
Keep rows where there's a match in both	INNER JOIN	N/A. This is default
Keep all rows from x, regardless of match in y	LEFT JOIN	set all.x = TRUE
Keep all rows from y, regardless of match in x	RIGHT JOIN	set all.y = TRUE
Keep all rows from x AND from y	OUTER JOIN	set all = TRUE

  - usingMergeAllX <- merge(data_tdfAC, airportCodes, all.x = TRUE)

VLOOKUP

- **join() in dplyr package**
  - usingJoin <- inner_join(data_tdfAC, airportCodes)
  - usingJoinLeft <- left_join(data_tdfAC, airportCodes)

  - **Join scenarios**
    - **inner_join**
    - **semi_join**
    - **left_join**
    - **anti_join**
    - **inner_join**
    - **semi_join**
    - **left_join**
    - **anti_join**
    - **full_join**

VLOOKUP

- **Compare merge() vs. join()**
  - **Speed**
  - **Resulting table**
    - **merge()**

ExcelUsingR.R × usingMergeAllX ×

Filter

	Dest	Year	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	UniqueCarrier	FlightNum	AirportName
1	ABE	1987	10	3	6	642	644	922	924	UA	980	Lehigh Valley International
2	ABE	1987	11	8	7	657	659	932	936	UA	980	Lehigh Valley International
3	ABE	1987	12	23	3	1127	1130	1147	1153	US	167	Lehigh Valley International
4	ABE	1987	10	11	7	1222	1205	1347	1322	NW	258	Lehigh Valley International
5	ABE	1987	12	31	4	1741	1741	1813	1805	EA	735	Lehigh Valley International
6	ABE	1987	12	9	3	1127	1130	1149	1153	US	167	Lehigh Valley International
7	ABE	1987	11	1	7	1128	1130	1150	1153	US	167	Lehigh Valley International

    - **join()**

ExcelUsingR.R × usingJoin ×

Filter

	Year	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	UniqueCarrier	FlightNum	Dest	AirportName
1	1987	10	14	3	741	730	912	849	PS	1451	SFO	San Francisco International
2	1987	10	15	4	729	730	903	849	PS	1451	SFO	San Francisco International
3	1987	10	17	6	741	730	918	849	PS	1451	SFO	San Francisco International
4	1987	10	18	7	729	730	847	849	PS	1451	SFO	San Francisco International
5	1987	10	19	1	749	730	922	849	PS	1451	SFO	San Francisco International
6	1987	10	21	3	728	730	848	849	PS	1451	SFO	San Francisco International
7	1987	10	22	4	728	730	852	849	PS	1451	SFO	San Francisco International

VLOOKUP – complex scenario

- **Match exact on one column; match between on two columns**

Member		
MemberID	VisitDate	VisitType
A123	8/10/2014	PROF
A123	9/23/2014	PROF
A123	10/11/2014	HOSP
A124	6/3/2014	PROF
A125	7/8/2014	HOSP

Match exactly on MemberID

AND

VisitDate must be between Begin/End date

Return only Coverage column value

Enrollment			
MemberID	BeginDate	EndDate	Coverage
A123	2/1/2014	9/30/2014	MK
A124	1/1/2014	12/31/2014	MC
A125	8/1/2014	12/31/2014	CO

VLOOKUP – complex scenario

- **sqldf() in sqldf package**
  - member <- read.csv("member.csv")
  - enrollment <- read.csv("enrollment.csv")

  - member$VisitDate <- as.Date(member$VisitDate, "%m/%d/%Y")
  - enrollment$BeginDate <- as.Date(enrollment$BeginDate, "%m/%d/%Y")
  - enrollment$EndDate <- as.Date(enrollment$EndDate, "%m/%d/%Y")

  - library(sqldf)

  - sqldfJoin <- sqldf("SELECT member.MemberID, member.VisitDate, member.VisitType, enrollment.Coverage
    FROM member
    LEFT OUTER JOIN enrollment ON member.MemberID = enrollment.MemberID
    AND (member.VisitDate > enrollment.BeginDate AND member.VisitDate < enrollment.EndDate)")

ExcelUsingR.R* × | sqldfJoin ×

Filter

	MemberID	VisitDate	VisitType	Coverage
1	A123	2014-08-10	PROF	MK
2	A123	2014-09-23	PROF	MK
3	A123	2014-10-11	HOSP	NA
4	A124	2014-06-03	PROF	MC
5	A125	2014-07-08	HOSP	NA

Load data into Excel from R

- **Utilizing XLConnect package**
  - **Example code to write a data frame to an Excel file**
    - library(XLConnect)
    - wb <- loadWorkbook("WriteToExcel.xlsx", create=TRUE)
    - createSheet(wb,"MemberEnrollment")
    - writeWorksheet(wb, sqldfJoin, "MemberEnrollment")
    - saveWorkbook(wb)

MemberID	VisitDate	VisitType	Coverage
A123	08/09/2014 20:00:00	PROF	MK
A123	09/22/2014 20:00:00	PROF	MK
A123	10/10/2014 20:00:00	HOSP	
A124	06/02/2014 20:00:00	PROF	MC
A125	07/07/2014 20:00:00	HOSP	

  - **From the XLConnect vignette**
    - Question: How can I style my output - set fonts, colors etc?
    - Answer: XLConnect does not currently allow direct access to low-level formatting options

Other Excel tasks?

- **?**
  - **?**
    - ?

Q&A and Next Steps