CHOP Analytics: R Standards

Making sustainability and collaboration easy

Overview

- Querying philosophy
- Organization with R Projects
- Code Structure
- Code Style
- Publishing to R Studio Connect

Querying Philosophy

Netezza will remain the standard tool for querying data because it is the fastest querying tool available. R is meant to manipulate, analyze, and visualize SQL-based datasets

Querying Philosophy Advantages

- There is no need to learn a new tool to query the CDW
- Minimizes the opportunity for overly complex metric building within R
- Data mart philosophy and standards will remain in tact

“

Organization with R Project

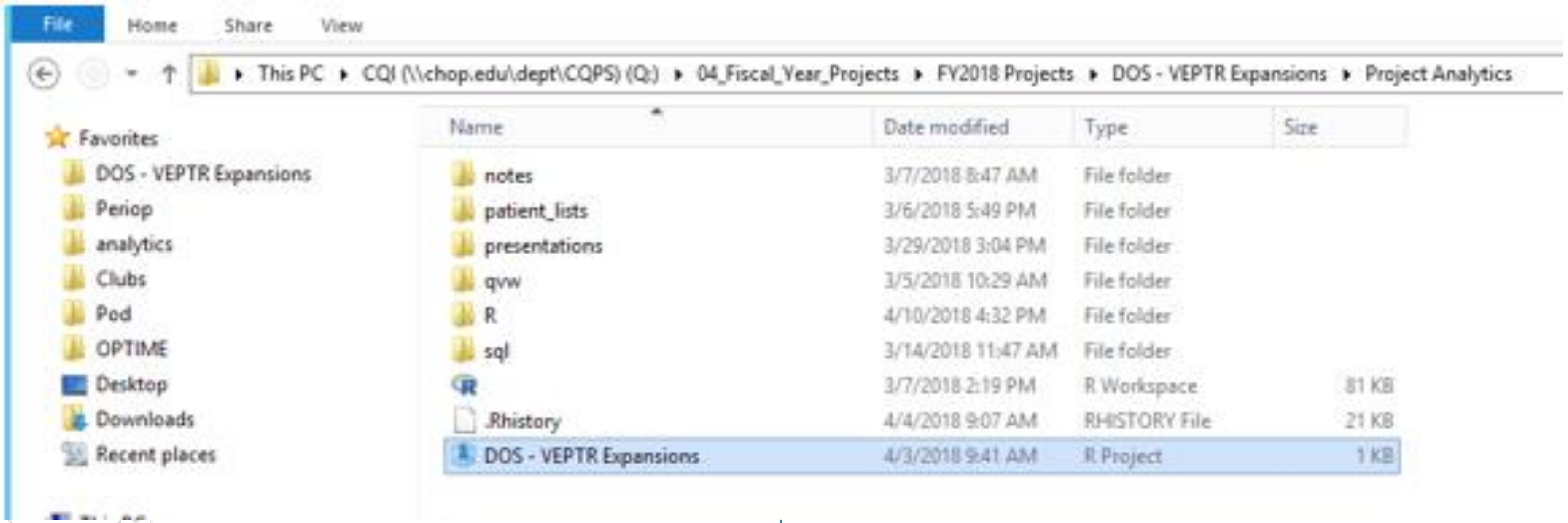
All project-based R scripts will be organized into R projects in order to make code review, project handoff, and project sustainability more efficient

R Project Advantages

- R projects store all code and outputs in a single location and eliminates the need to set a working directory
- R projects store relative file paths for better reproducibility
- They also create a clean R environment so you do not need to remove objects prior to analysis
- All of the above allow R to be more easily integrated in analysts workflow

R Project Set-Up

- In order to set-up an R project, you will need to create a directory for your project work and save an R project file at the ***top level*** of the directory



This PC ▸ CQI (\\chop.edu\dept\CQPS) (Q:) ▸ 04_Fiscal_Year_Projects ▸ FY2018 Projects ▸ DOS - VEPTR Expansions ▸ Project Analytics

Favorites
- DOS - VEPTR Expansions
- Periop
- analytics
- Clubs
- Pod
- OPTIME
- Desktop
- Downloads
- Recent places

Name	Date modified	Type	Size
notes	3/7/2018 8:47 AM	File folder	
patient_lists	3/6/2018 5:49 PM	File folder	
presentations	3/29/2018 3:04 PM	File folder	
qvw	3/5/2018 10:29 AM	File folder	
R	4/10/2018 4:32 PM	File folder	
sql	3/14/2018 11:47 AM	File folder	
	3/7/2018 2:19 PM	R Workspace	81 KB
.Rhistory	4/4/2018 9:07 AM	RHISTORY File	21 KB
DOS - VEPTR Expansions	4/3/2018 9:41 AM	R Project	1 KB

```
read.csv('patient_lists/example.csv')
```

R Project Exercise

- Create a directory and R project on your H drive that you will use throughout the chopr sessions
- Create a folder in that directory called ‘data’
- Copy and paste the blood culture dataset from the chopR repo to your ‘data’ folder

“

Code Structure

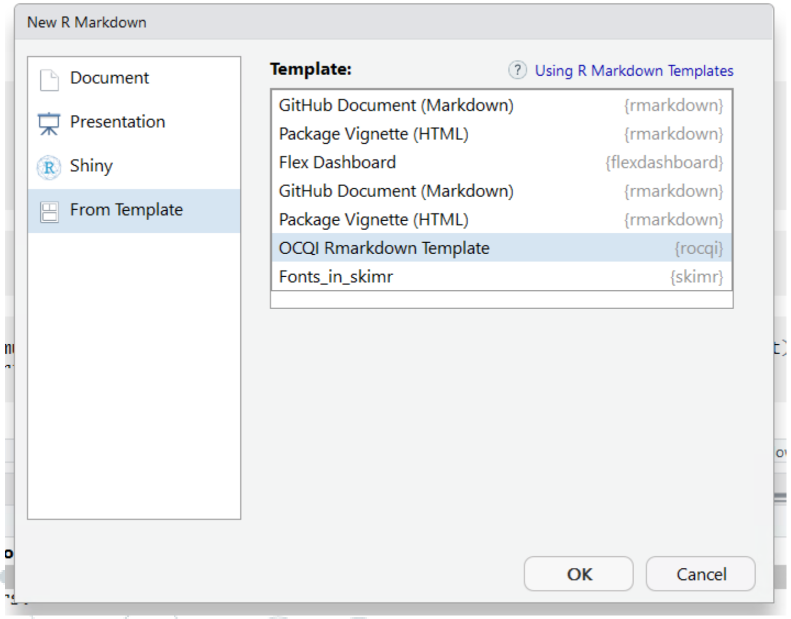
All R scripts must adhere to CHOP Analytics code structure in order to make code review, project handoff, and project sustainability more efficient

Code Structure Overview

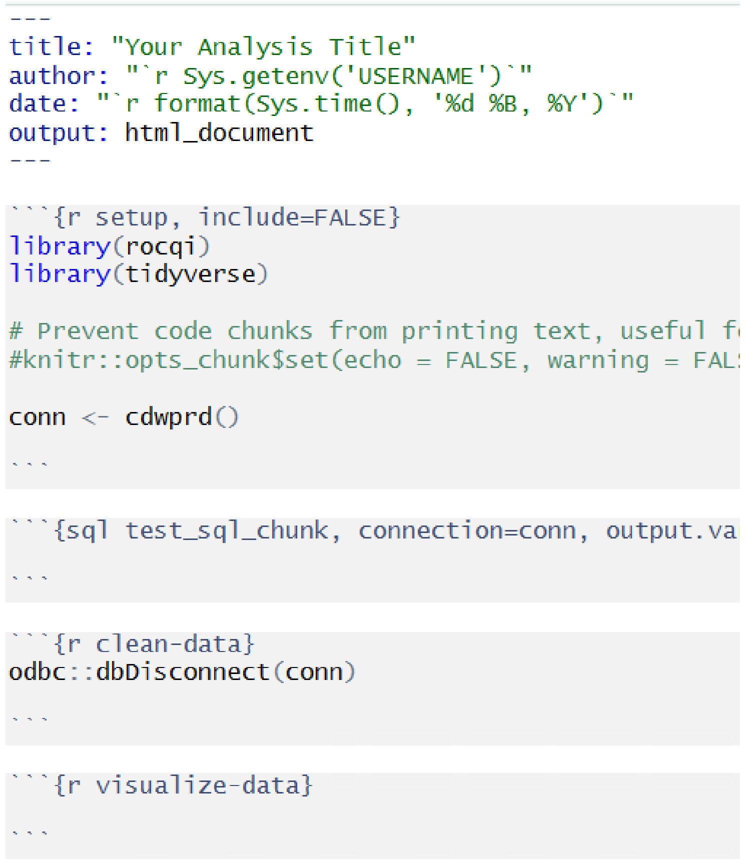
- All scripts should be organized into chunks and each chunk should be named logically
- First chunk should load all packages and set connection strings
- Second chunk should pull in all data
- Third chunk should format all data
- Subsequent chunks should be used for analyses

Code Structure is Easy with the rocqi Template

- File → New File → R Markdown → From Template


New R Markdown

GitHub Document (Markdown)	{rmarkdown}
Package Vignette (HTML)	{rmarkdown}
Flex Dashboard	{flexdashboard}
GitHub Document (Markdown)	{rmarkdown}
Package Vignette (HTML)	{rmarkdown}
OCQI Rmarkdown Template	{rocqi}
Fonts_in_skimr	{skimr}


~~~
---
title: "Your Analysis Title"
author: "`r Sys.getenv('USERNAME')`"
date: "`r format(Sys.time(), '%d %B, %Y')`"
output: html_document
---

```{r setup, include=FALSE}
library(rocqi)
library(tidyverse)

# Prevent code chunks from printing text, useful fo
#knitr::opts_chunk$set(echo = FALSE, warning = FAL

conn <- cdwprd()

~~~

```{sql test_sql_chunk, connection=conn, output.va

```

```{r clean-data}
odbc::dbDisconnect(conn)

```

```{r visualize-data}

```
```
```

Code Structure Exercise

- Open your R project (if it is closed)
- Open up the rocqi template and save it in your directory
- Outside of your SQL chunk, pull in the blood culture csv into the .Rmd

```
blood_culture<-read.csv('data/blood_culture.csv')
```

Code Style

All R scripts must adhere to CHOP Analytics style in order to make code review, project handoff, and project sustainability more efficient

Code Style Overview

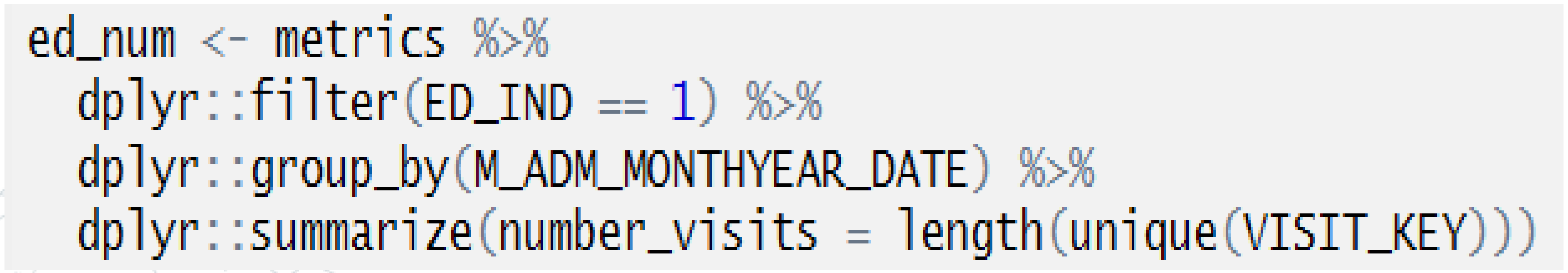
- Consistent with tidyverse R style for coding
- Standards are focused on naming, alignment, spacing, and commenting (details found on kernel)
- A list of standard packages should be used for the majority of our analyses
  - Process for adding new packages to the list of standard packages can also be found on kernel
- It is all made easy with `lintr`!

Code Style Highlights

- All function calls must be preceded with package name followed by "::"
  - reshape2::melt()

- When assigning values to new objects, '<-' should be used instead of "="

- Each function you pipe should be on its own line



```
ed_num <- metrics %>%
  dplyr::filter(ED_IND == 1) %>%
  dplyr::group_by(M_ADM_MONTHYEAR_DATE) %>%
  dplyr::summarize(number_visits = length(unique(VISIT_KEY)))
```

Code Style Exercise

- Instructor will demonstrate how to use the lintr packages to point out stylistic errors

“

Publishing with R Studio Connect

All deliverables on R Studio Connect should be sourced from a data mart in order to improve server efficiency