

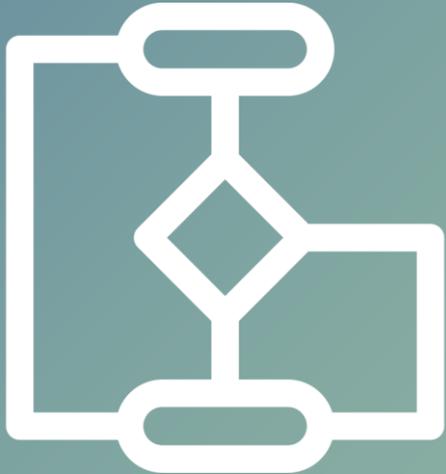


Visualizing Data Lineages

David Manley
Senior Institutional Research Analyst
Rowan University
July 17, 2024



What to Expect



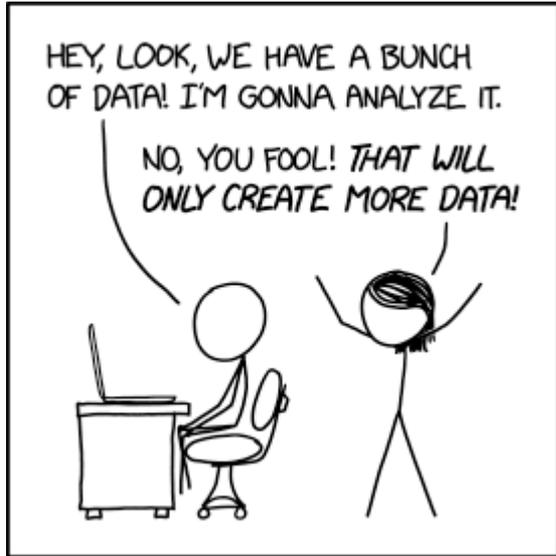
- A light survey of data lineage visualization tools (by no means exhaustive)
- A brief tutorial of how I draw data lineages (from very general to very technical)
- Comments on my experiences trying to find the “perfect” method for documenting lineages

Who Am I



- Senior Institutional Research Analyst at Rowan University
- Formerly a Statistics Instructor at Rowan University
- In a previous life I was a Quality & Reliability Engineer in manufacturing
- Data roots in:
 - Excel (with VBA to handle 65,000+)
 - JMP v3
 - Minitab v12
 - R (before Tidyverse and ggplot2)
- **In all job positions that I have held, I have used flow charts**

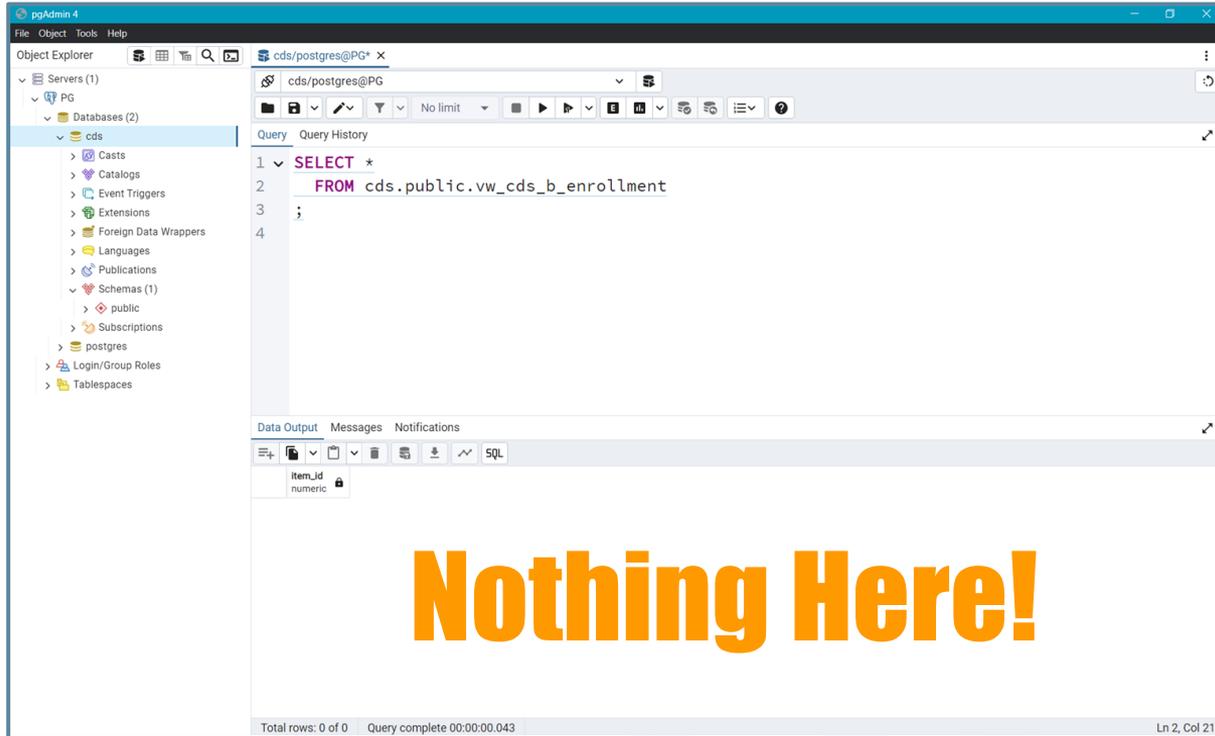
Data Trap



<https://xkcd.com/2582/>
xkcd by Randall Munroe

Don't worry
we will only discuss
METADATA
in this presentation!

In Fact, My Example Data Set Is Completely Empty



The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer shows a tree view of the database structure, including Servers, Databases, and Schemas. The 'cds' database is selected, and the 'public' schema is expanded. The main window displays a query editor with the following SQL code:

```
1 SELECT *
2 FROM cds.public.vw_cds_b_enrollment
3 ;
4
```

Below the query editor, the Data Output tab is active, showing a table with one column, 'Item_Id', of type 'numeric'. The table is currently empty. The status bar at the bottom indicates 'Total rows: 0 of 0' and 'Query complete 00:00:00.043'.

Nothing Here!

Before we go any further...

Callouts will have relevant file names for the slide

All example files used in this presentation are available on my Google Drive

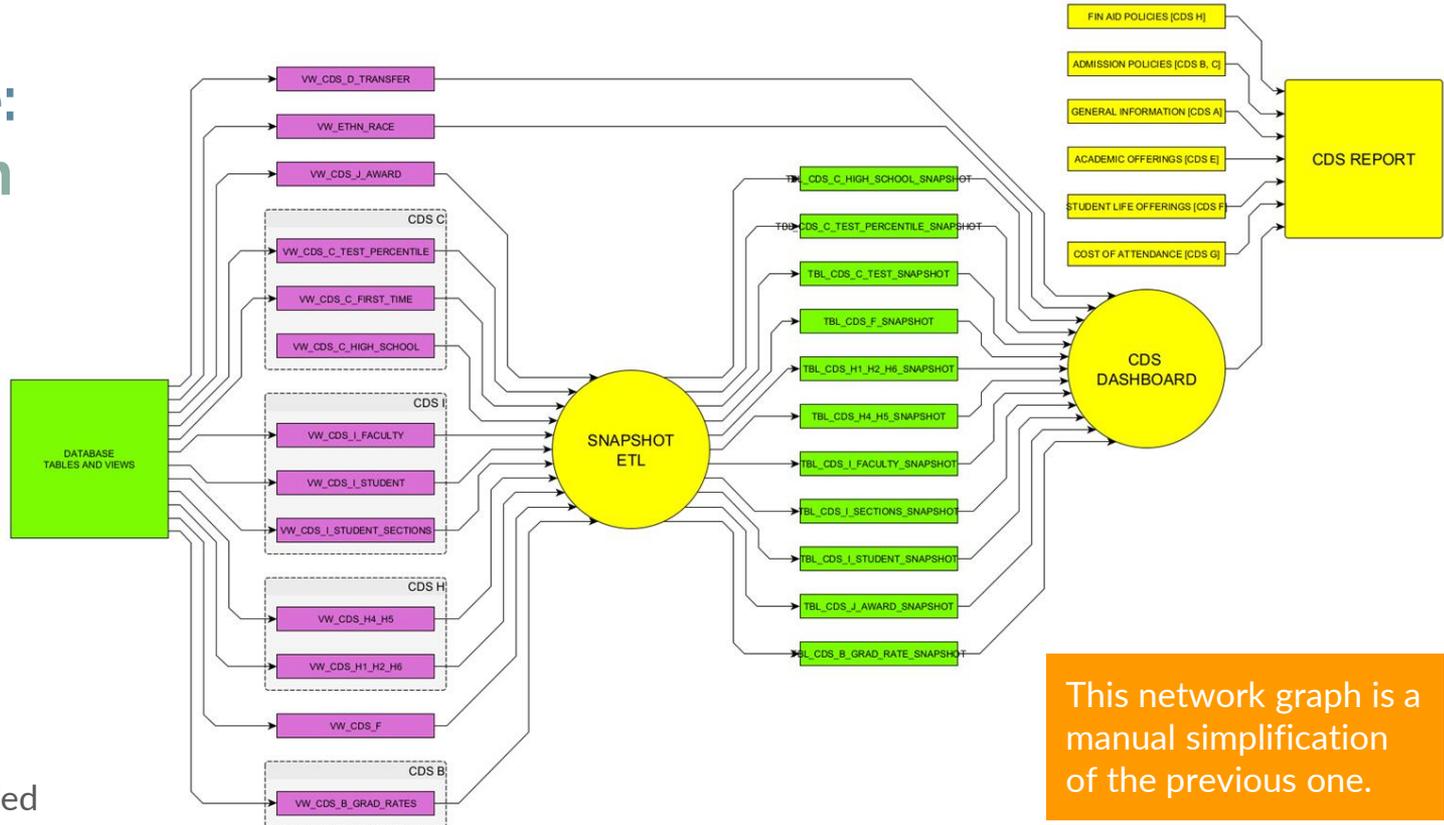
<https://drive.google.com/drive/folders/1j03hiCeEVpzQkp9p1EnrOREmXYuaGdLD?usp=sharing>



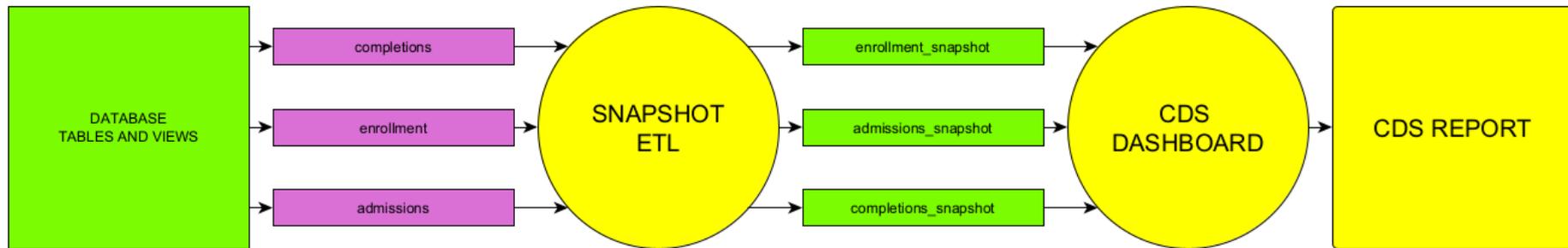
Drive

Today's Example: Common Data Set (CDS) [simpler]

- Tables (tbl_???)
- Views (vw_???)
- Manually added



Today's Example: Common Data Set (CDS) [simplest]



-  Tables (tbl_???)
-  Views (vw_???)
-  Manually added

This network graph was drawn completely manually.

cds_create_tables_view_postgres.sql
cds_create_tables_views_oracle.sql

cds_drop_views_tables_postgres.sql
cds_drop_views_tables_oracle.sql

Example CDS Database

The screenshot shows the pgAdmin 4 interface. The left pane shows the 'Object Explorer' with the 'cds' database selected. The main pane shows a query window with the following SQL:

```
1 SELECT *
2 FROM information_schema.tables
3 WHERE LOWER(table_name) NOT LIKE '%pg_%'
4 AND LOWER(table_schema) <> 'information_schema'
```

Below the query, the 'Data Output' pane displays a table with 19 rows and 10 columns. The columns are: table_catalog, table_schema, table_name, table_type, self_referencing_column_name, reference_generation, user_defined_type_catalog, and user_defined_type_name. The data shows various tables in the 'cds' database, including 'tbl_degree_status', 'tbl_degree_level', 'tbl_sure_award_crosswalk', etc.

table_catalog	table_schema	table_name	table_type	self_referencing_column_name	reference_generation	user_defined_type_catalog	user_defined_type_name
cds	public	tbl_degree_status	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_degree_level	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_sure_award_crosswalk	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_degree_type	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_cip_code	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_term_code	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_jobs	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_admissions_application	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_prior_degree	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_admissions_decision	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_admissions_census	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_enrollment_census	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_award	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_major	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_award_status	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_fafsa	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_program	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_fiscal_year	BASE TABLE	[null]	[null]	[null]	[null]
cds	public	tbl_course_schedule_credential	RASF TARI F	[null]	[null]	[null]	[null]

Total rows: 82 of 82 Query complete 00:00:00.068 Ln 4, Col 27

Contains
no data

56 empty tables
26 dummy views

PostgreSQL
and Oracle
formats
available



Scope and Goals

Why Visualize Data Lineages?

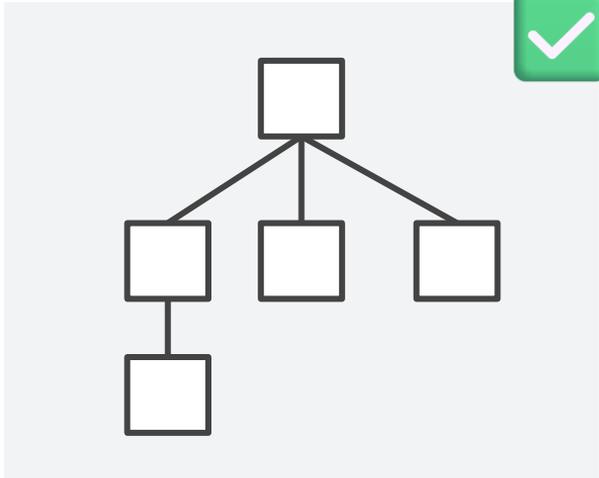
1. Design data flows (intent)
2. Monitor data systems (reality)
3. Documenting (consistency)
4. Training (speed)
5. Explaining (common understanding)

Bonus: Change of pace from bar charts, line graphs, heatmaps, etc.

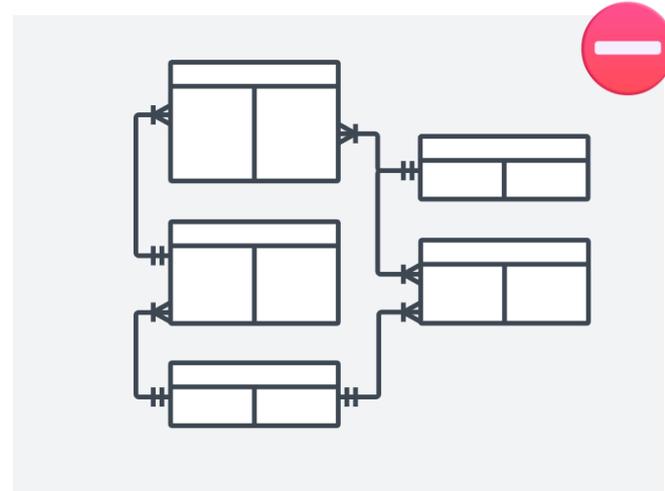


For This Presentation We Want to

Visualize Data
Flows
(Hierarchies)



Not Entity
Relationship (ER)
Diagrams



Drawing Lineages Can Be Tedious...

Rearranging a Lineage Diagram is Even Worse!





I Want to...

- ...draw a data lineage network graph quickly
 - table-level lineages
 - column-level lineages
- ...update and rearrange an existing lineage network graph effortlessly
- ...use database and data tool metadata to help draw my lineage
- ...share my my lineage diagram easily

Cost matters, but I will not consider heavily during this presentation

Disclaimer

I respect and appreciate the effort of the developers that created the tools I mention on the following slides. Thanks to them I have a variety of choices, accessible with only a few clicks. The tiers on the following slides only consider the aspects of the software related to creating data lineages and do not assess the tools in their entirety.

Thank you developers! 🙏



Tiers of Lineage Tools

My Tiers of Lineage Visualization Tools

Tier	Example Tools
5. Continuous lineage monitoring	Informatica, Collibra, Octopai
4. Automated metadata import, advanced automated layout	iGraph for R and Python, SQL Flow
3. Assisted drawing, advanced automated layout	yEd/yEd Live
2. Assisted drawing, basic automated layout	MS Visio, Draw.io, Lucidchart/LucidSpark
1. Manual drawing, manual layout	MS Office Apps, Google Apps



Tier 1
Manual drawing
Manual layout



Tier 2

Assisted drawing

Basic automated layout

Draw.io <https://draw.io/>

Can save files locally

Open source

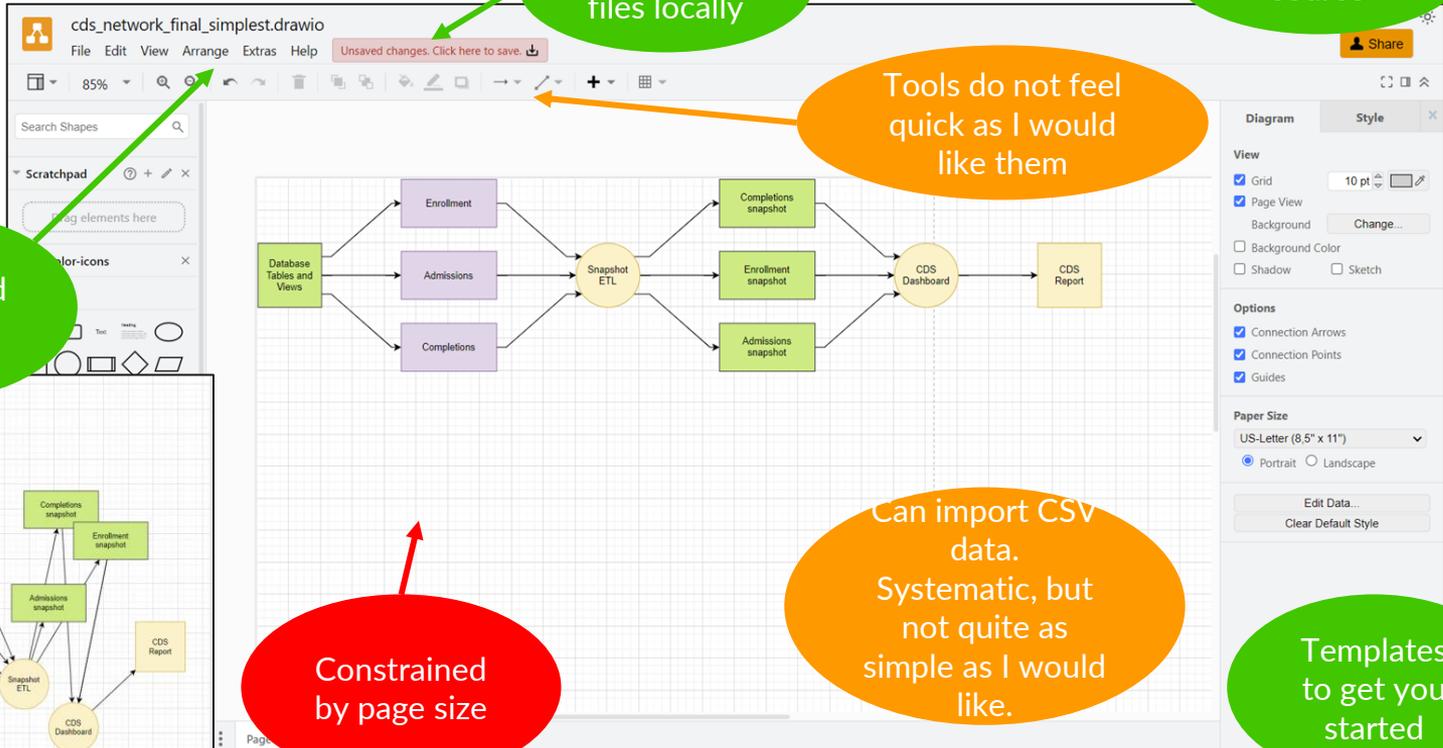
Tools do not feel quick as I would like them

Some automated layout options

Constrained by page size

Can import CSV data. Systematic, but not quite as simple as I would like.

Templates to get you started



Tier 3

Assisted drawing

Advanced automated layout

yEd Desktop/yEd Live <https://www.yworks.com/yed-live/>

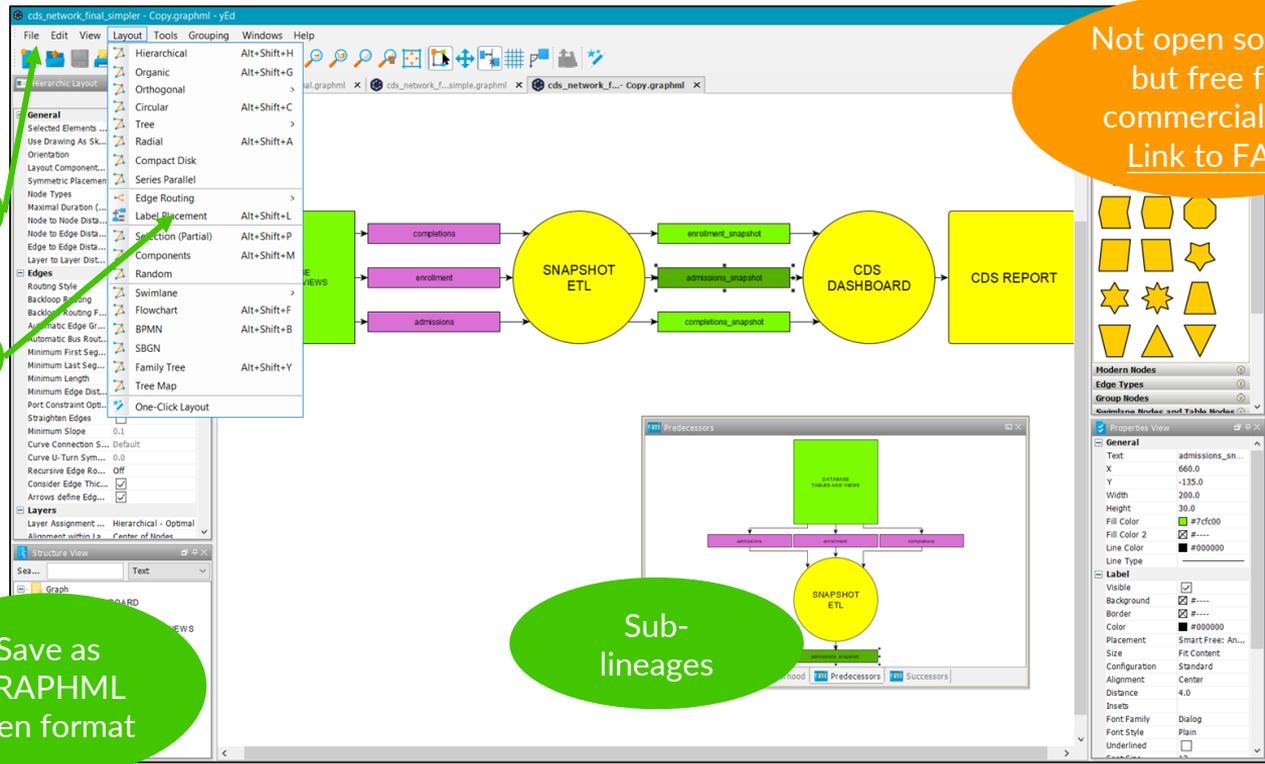
Can import metadata

Advanced layout tools

Save as GRAPHML open format

Sub-lineages

Not open source, but free for commercial use
[Link to FAQ](#)



What do I consider a simple metadata export?

1	NAME	TYPE	REFERENCED_NAME	REFERENCED_TYPE
26	VW_CDS_C_COURSES	VIEW	VW_CDS_C_COURSES	VIEW
27	VW_CDS_C_COURSES	VIEW	VW_CDS_C_COURSES	VIEW
28	VW_CDS_C_FIRST_TIME	VIEW	TBL_ADMISSIONS_APPLICATION	TABLE
29	VW_CDS_C_FIRST_TIME	VIEW	TBL_ADMISSIONS_CENSUS	TABLE
30	VW_CDS_C_FIRST_TIME	VIEW	TBL_ADMISSIONS_DECISION	TABLE
31	VW_CDS_C_FIRST_TIME	VIEW	TBL_DEGREE_TYPE	TABLE
32	VW_CDS_C_FIRST_TIME	VIEW	TBL_ENROLLMENT_CENSUS	TABLE
33	VW_CDS_C_FIRST_TIME	VIEW	TBL_PRIOR_DEGREE	TABLE
34	VW_CDS_C_HIGH_SCHOOL	VIEW	TBL_STUDENT_HIGH_SCHOOL	TABLE
35	VW_CDS_C_HIGH_SCHOOL	VIEW	VW_CDS_B_ENROLLMENT	VIEW
36	VW_CDS_C_TEST	VIEW	TBL_SAT_SCORE_CROSSWALK	TABLE
37	VW_CDS_C_TEST	VIEW	TBL_TEST_SCORE	TABLE
38	VW_CDS_C_TEST	VIEW	TBL_TEST_TYPE	TABLE
39	VW_CDS_C_TEST	VIEW	VW_CDS_B_ENROLLMENT	VIEW
40	VW_CDS_C_TEST_PERCENTILE	VIEW	VW_CDS_C_TEST	VIEW
41	VW_CDS_D_TRANSFER	VIEW	TBL_ADMISSIONS_APPLICATION	TABLE
42	VW_CDS_D_TRANSFER	VIEW	TBL_ADMISSIONS_CENSUS	TABLE
43	VW_CDS_D_TRANSFER	VIEW	TBL_ADMISSIONS_DECISION	TABLE
44	VW_CDS_D_TRANSFER	VIEW	TBL_DEGREE_TYPE	TABLE
45	VW_CDS_D_TRANSFER	VIEW	TBL_ENROLLMENT_CENSUS	TABLE

Each From-To Pair = **EDGE**
 Each Name or Ref_Name = **NODE**



Process to Draw Lineage from Metadata with yEd

Step 1

Export data from database or data tool

Step 2

Reshape metadata for yEd (nodes and edges)

Step 3

Map node and edge columns to yEd fields

Step 4

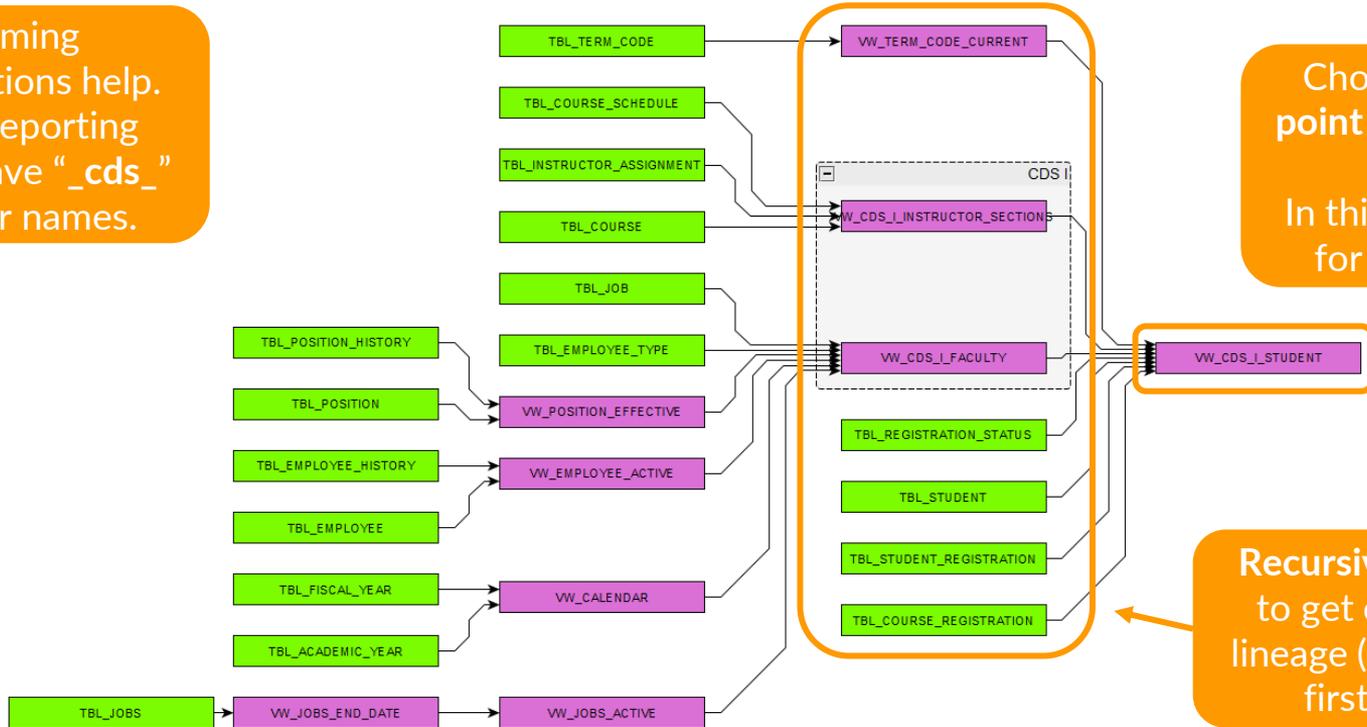
Map additional properties to yEd fields

Step 5

Layout and customize diagram

STEP 1: A Quick Note About Dependency Queries

Naming conventions help. Here reporting views have “_cds_” in their names.

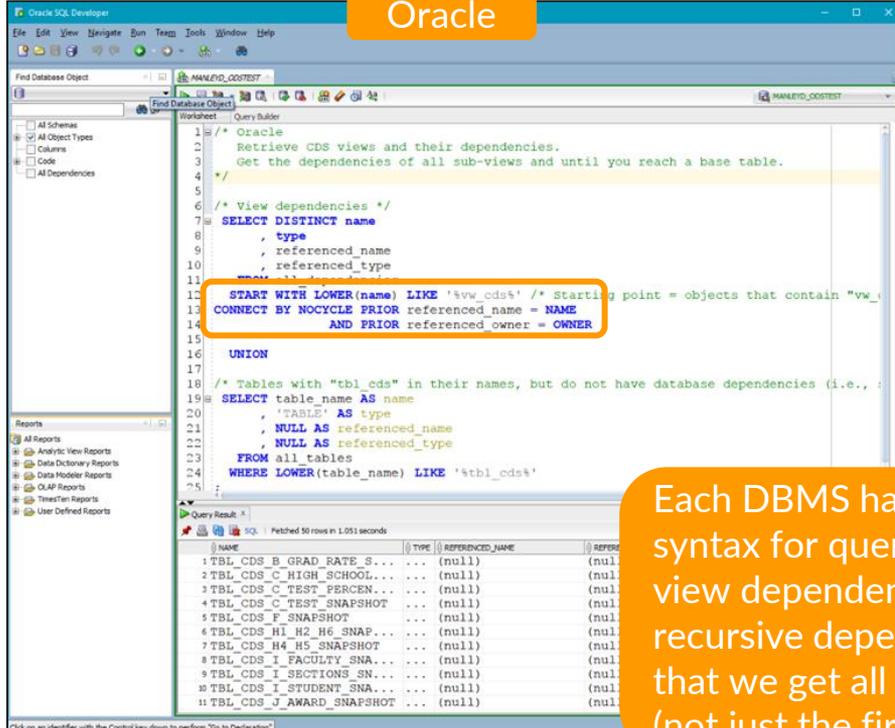


Choose **starting point** deep into the lineage. In this case search for “vw_cds%”

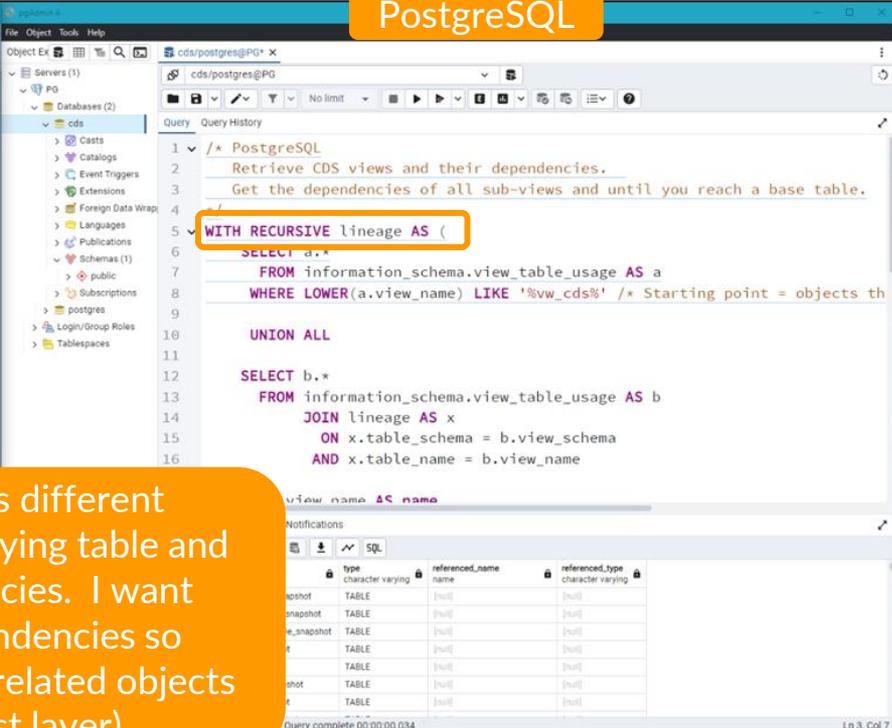
Recursively query to get complete lineage (no just the first layer)

STEP 1: Export Data from Database or Data Tool

Oracle



PostgreSQL



Each DBMS has different syntax for querying table and view dependencies. I want recursive dependencies so that we get all related objects (not just the first layer).

STEP 2: Reshape metadata for yEd

Edges

Nodes

Edges

NAME	REFERENCED_NAME
VW_CALENDAR	TBL_ACADEMIC_YEAR
VW_CALENDAR	TBL_FISCAL_YEAR
VW_CALENDAR	TBL_TERM_CODE
VW_CDS_B_ENROLLMENT	TBL_ADMISSIONS_CENSUS
VW_CDS_B_ENROLLMENT	TBL_DEGREE_TYPE
VW_CDS_B_ENROLLMENT	TBL_ENROLLMENT_CENSUS
VW_CDS_B_ENROLLMENT	TBL_FAFA
VW_CDS_B_ENROLLMENT	TBL_PROGRAM
VW_CDS_B_ENROLLMENT	TBL_TERM_CODE
VW_CDS_B_GRAD_RATES	TBL_FIN_AID_AWARD
VW_CDS_B_GRAD_RATES	TBL_PERSON
VW_CDS_B_GRAD_RATES	TBL_TERM_CODE
VW_CDS_B_GRAD_RATES	VW_CDS_B_ENROLLMENT
VW_CDS_B_GRAD_RATES	VW_CDS_J_AWARD
VW_CDS_B_GRAD_RATES	VW_TERM_CODE_CURRENT
VW_CDS_C_FIRST_TIME	TBL_ADMISSIONS_APPLICATION
VW_CDS_C_FIRST_TIME	TBL_ADMISSIONS_CENSUS
VW_CDS_C_FIRST_TIME	TBL_ADMISSIONS_DECISION
VW_CDS_C_FIRST_TIME	TBL_DEGREE_TYPE
VW_CDS_C_FIRST_TIME	TBL_ENROLLMENT_CENSUS
VW_CDS_C_FIRST_TIME	TBL_PRIOR_DEGREE
VW_CDS_C_HIGH_SCHOOL	TBL_STUDENT_HIGH_SCHOOL
VW_CDS_C_HIGH_SCHOOL	VW_CDS_B_ENROLLMENT
VW_CDS_C_TEST	TBL_SAT_SCORE_CROSSWALK
VW_CDS_C_TEST	TBL_TEST_SCORE
VW_CDS_C_TEST	TBL_TEST_TYPE
VW_CDS_C_TEST	VW_CDS_B_ENROLLMENT
VW_CDS_C_TEST_PERCENTILE	VW_CDS_C_TEST
VW_CDS_D_TRANSFER	TBL_ADMISSIONS_APPLICATION
VW_CDS_D_TRANSFER	TBL_ADMISSIONS_CENSUS
VW_CDS_D_TRANSFER	TBL_ADMISSIONS_DECISION
VW_CDS_D_TRANSFER	TBL_DEGREE_TYPE
VW_CDS_D_TRANSFER	TBL_ENROLLMENT_CENSUS
VW_CDS_D_TRANSFER	TBL_PRIOR_DEGREE
VW_CDS_F	TBL_ACTIVITY_TYPE
VW_CDS_F	TBL_STUDENT_ACTIVITY
VW_CDS_F	VW_CDS_B_ENROLLMENT
VW_CDS_H1_H2_H6	VW_CDS_H_AWARD

Prepare EDGES for Yed
1. Copied DB_OUTPUT sheet
2. Removed the TYPE columns

Nodes

NAME	TYPE	COLOR
TBL_ACADEMIC_YEAR	TABLE	#7CFC11
TBL_ACTIVITY_TYPE	TABLE	#7CFC28
TBL_ADMISSIONS_APPLICATION	TABLE	#7CFC21
TBL_ADMISSIONS_CENSUS	TABLE	#7CFC14
TBL_ADMISSIONS_DECISION	TABLE	#7CFC22
TBL_AWARD	TABLE	#7CFC43
TBL_CDS_B_GRAD_RATE_SNAPSHOT	TABLE	#7CFC00
TBL_CDS_C_HIGH_SCHOOL_SNAPSHOT	TABLE	#7CFC01
TBL_CDS_C_TEST_PERCENTILE_SNAPSHOT	TABLE	#7CFC02
TBL_CDS_C_TEST_SNAPSHOT	TABLE	#7CFC03
TBL_CDS_F_SNAPSHOT	TABLE	#7CFC04
TBL_CDS_H1_H2_H6_SNAPSHOT	TABLE	#7CFC05
TBL_CDS_H4_HS_SNAPSHOT	TABLE	#7CFC06
TBL_CDS_I_FACULTY_SNAPSHOT	TABLE	#7CFC07
TBL_CDS_I_SECTIONS_SNAPSHOT	TABLE	#7CFC08
TBL_CDS_I_STUDENT_SNAPSHOT	TABLE	#7CFC09
TBL_CDS_J_AWARD_SNAPSHOT	TABLE	#7CFC10
TBL_COHORT	TABLE	#7CFC21
TBL_COURSE	TABLE	#7CFC23
TBL_COURSE_REGISTRATION	TABLE	#7CFC28
TBL_COURSE_SCHEDULE	TABLE	#7CFC26
TBL_COURSE_SCHEDULE_CROSSLIST	TABLE	#7CFC42
TBL_DEGREE_LEVEL	TABLE	#7CFC52
TBL_DEGREE_STATUS	TABLE	#7CFC44
TBL_DEGREE_TYPE	TABLE	#7CFC15
TBL_DEMOGRAPHICS	TABLE	#7CFC49
TBL_EMPLOYEE	TABLE	#7CFC47
TBL_EMPLOYEE_HISTORY	TABLE	#7CFC48
TBL_EMPLOYEE_TYPE	TABLE	#7CFC53
TBL_ENROLLMENT_CENSUS	TABLE	#7CFC16
TBL_FAFA	TABLE	#7CFC17
TBL_FIN_AID_AWARD	TABLE	#7CFC19
TBL_FIN_AID_STUDENT	TABLE	#7CFC30
TBL_FISCAL_YEAR	TABLE	#7CFC12
TBL_INSTRUCTOR_ASSIGNMENT	TABLE	#7CFC37
TBL_JOB	TABLE	#7CFC34
TBL_JOBS	TABLE	#7CFC50
TBL_MAJOR	TABLE	#7CFC45

Prepare NODES for Yed
1. Copied DB_OUTPUT sheet
2. Stacked NAME and REFERENCED_NAME
3. Stacked TYPE and REFERENCED_TYPE
4. Added COLOR column with HTML color code for tables and views

Manually added color column to distinguish tables and views

STEP 3: Map node and edge columns to yEd fields

MS Excel Import

Data

Edge Representation: Edge List

Edge List

Data Range: EDGES!A1:B92 Adopt

Column of Source IDs: EDGES!B Adopt

Column of Target IDs: EDGES!A Adopt

Property Names in First Row:

Information: 69 Nodes, 91 Edges

Node List

Data Range: NODES!A1:C81 Adopt

Column of Node IDs: NODES!A Adopt

Column of Group Node IDs: Adopt

Property Names in First Row:

Information: 80 Nodes, 3 Node Properties

DB_OUPUT	EDGES	NODES
1	NAME	TYPE
2	TBL_AC...	TABLE #7CFC11
3	TBL_AC...	TABLE #7CFC28
4	TBL_AD...	TABLE #7CFC21
5	TBL_AD...	TABLE #7CFC14
6	TBL_AD...	TABLE #7CFC22
7	TBL_A...	TABLE #7CFC43
8	TBL_CD...	TABLE #7CFC00
9	TBL_CD...	TABLE #7CFC01
10	TBL_CD...	TABLE #7CFC02
11	TBL_CD...	TABLE #7CFC03
12	TBL_CD...	TABLE #7CFC04
13	TBL_CD...	TABLE #7CFC05
14	TBL_CD...	TABLE #7CFC06
15	TBL_CD...	TABLE #7CFC07
16	TBL_CD...	TABLE #7CFC08
17	TBL_CD...	TABLE #7CFC09
18	TBL_CD...	TABLE #7CFC10
19	TBL_CO...	TABLE #7CFC31
20	TBL_CO...	TABLE #7CFC35
21	TBL_CO...	TABLE #7CFC38
22	TBL_CO...	TABLE #7CFC36
23	TBL_CO...	TABLE #7CFC42
24	TBL_DE...	TABLE #7CFC53
25	TBL_DE...	TABLE #7CFC44

Ok Reset Cancel Help

Click File > Open and Select an Excel file to get this dialog

MS Excel Import

Data Presentation

Nodes

Use Configuration:

Configuration: New Configuration (Node)

Label Text: NAME

Fit Size to Label:

Template: Selected Template from Palette

Edges

Use Configuration:

Configuration:

Label Text: None

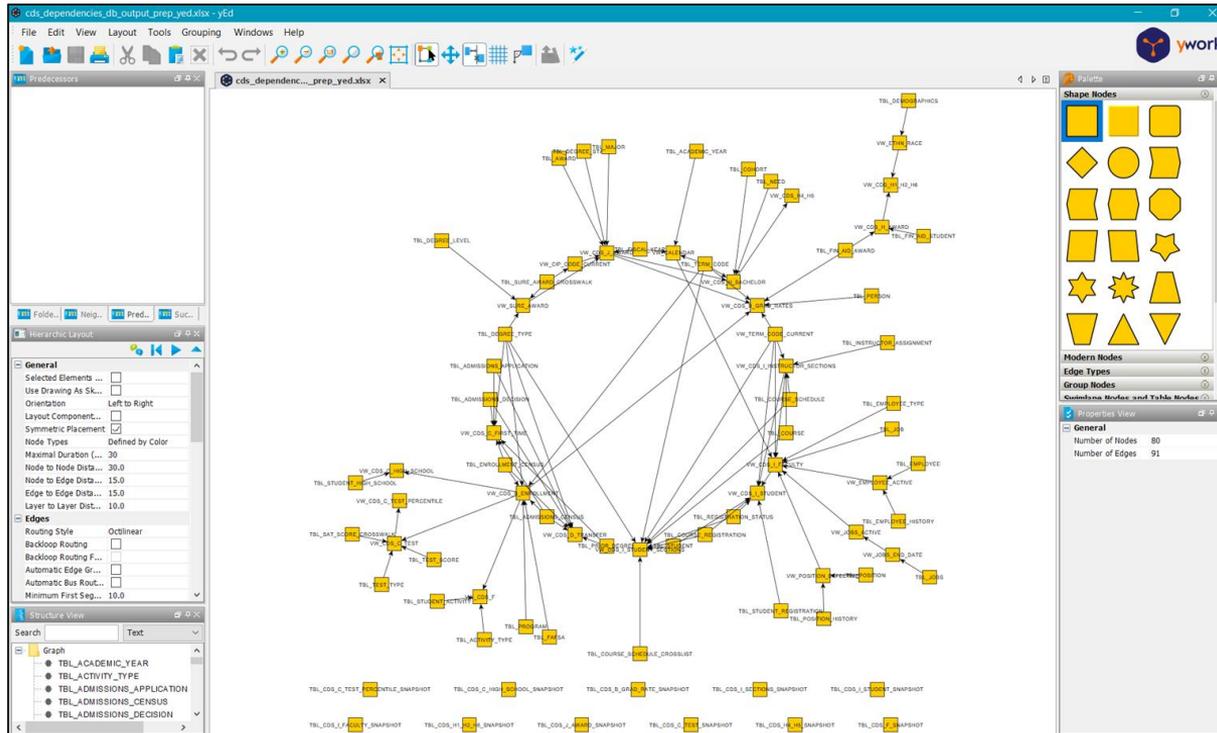
Template: Selected Template from Palette

Layout

Layout: Circular

Start with Circular layout just for fun.

STEP 3: Results So Far



STEP 4: Map additional properties to yEd fields

Properties Mapper

Configurations

Selected Configuration

Name: New Configuration

Template: None Single Multiple Icon

Mappings: Act On Selection Only

Data Source	Map To	Conversion
NAME	Label Text	automatic
COLOR	Fill Color	automatic

Map COLOR variable to Fill Color

Selected Mapping

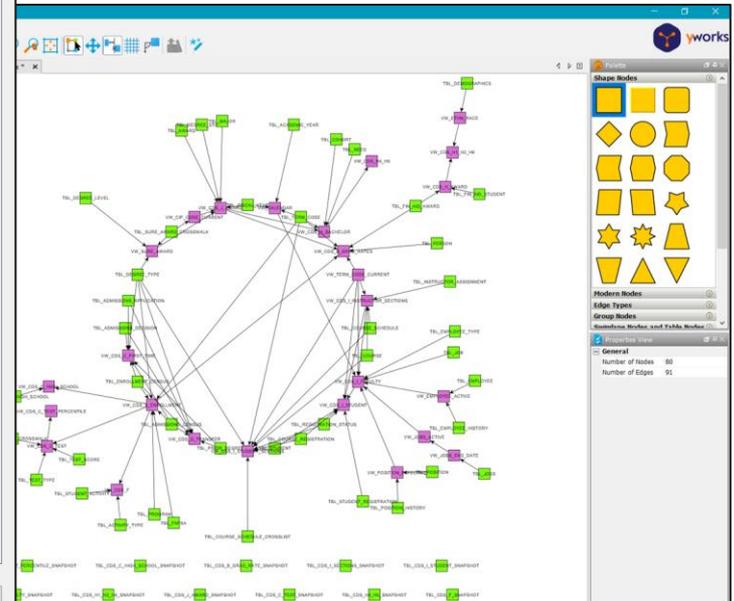
Map To Label No. 1 Replace Text Fit Node to Label

Conversion Details (NAME -> Label Text)

The selected conversion requires no configuration.

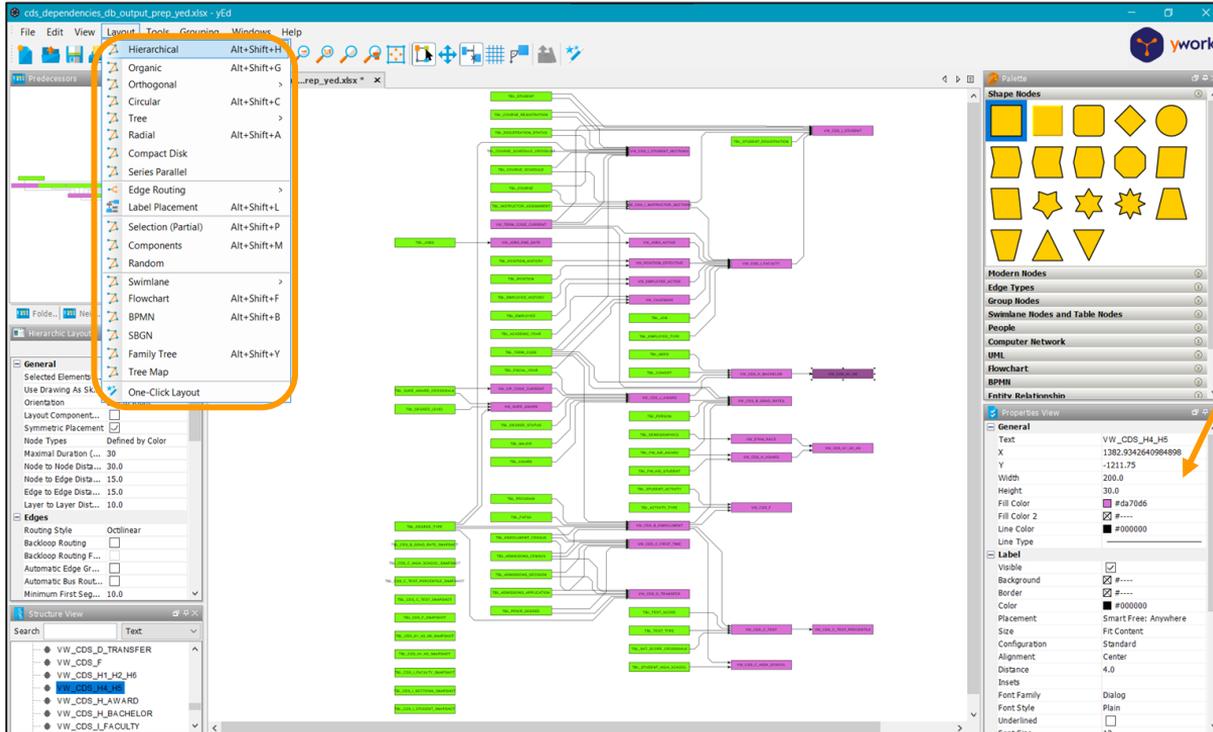
Click Edit > Properties Mapper to get this dialog

Ok Apply Cancel Help



STEP 5: Layout and customize diagram

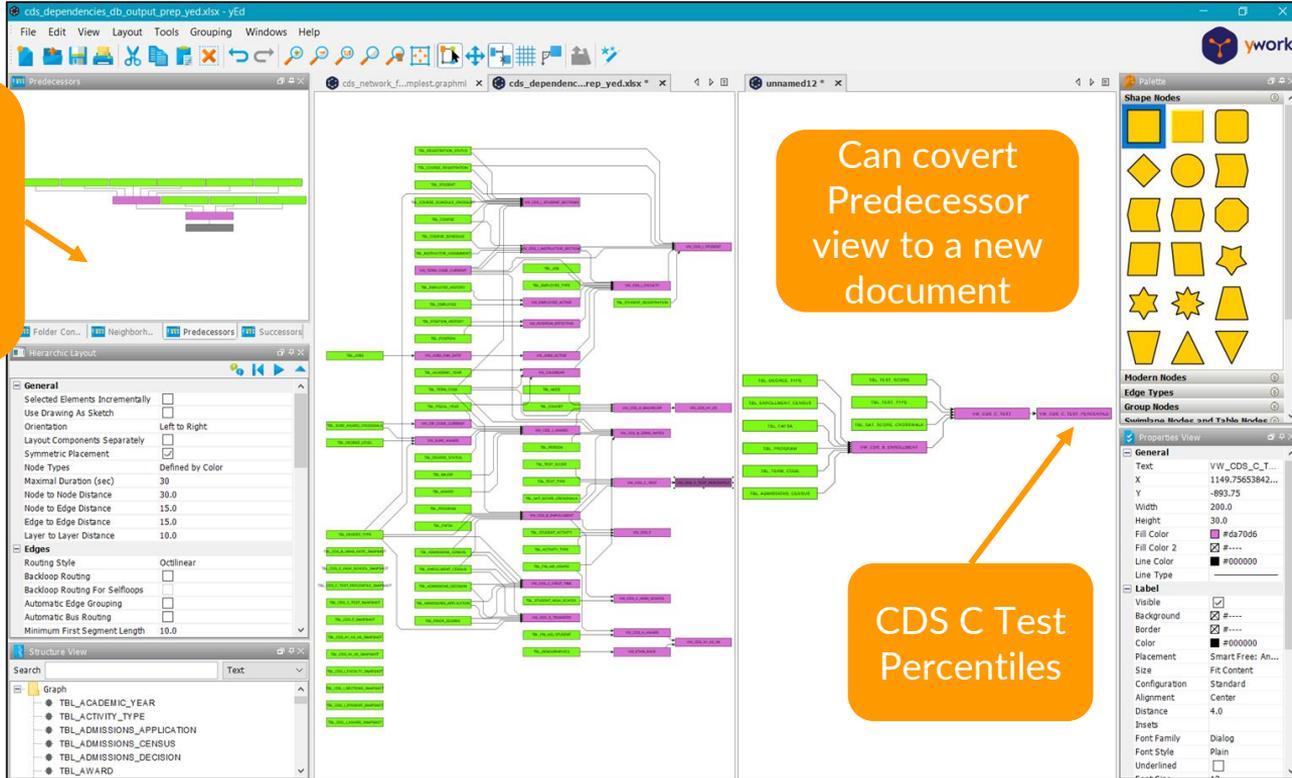
Layout menu has many choices. Used Hierarchical here.



TIP: Select a single node a press CTRL+A to select all nodes. Can then edit properties of all nodes without affecting edges.

STEP 5: Sub Lineages

Predecessors
and
Successors
view

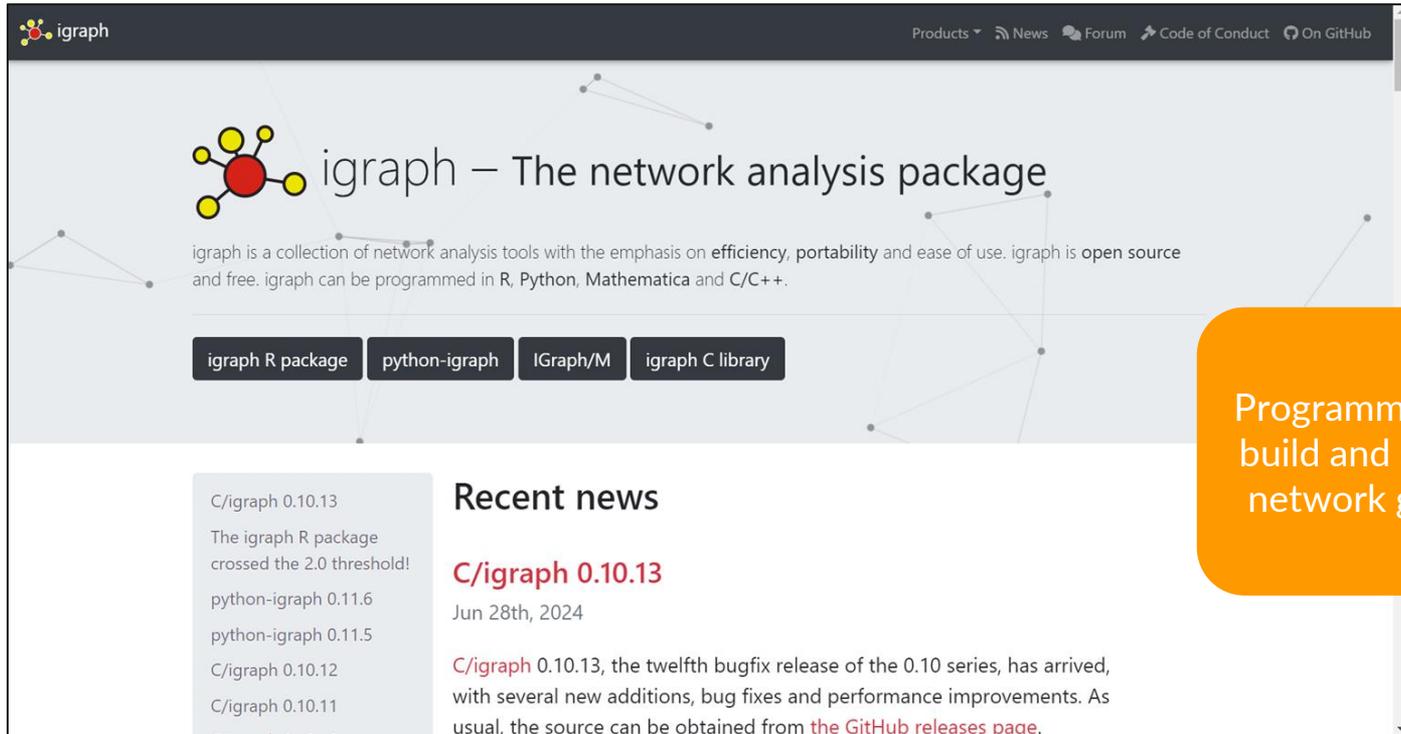


Tier 4

Automated metadata import

Advanced automated layout

iGraph <https://igraph.org/>



The screenshot shows the iGraph website homepage. At the top left is the iGraph logo, and at the top right are navigation links for Products, News, Forum, Code of Conduct, and On GitHub. The main heading is "igraph – The network analysis package". Below this is a descriptive paragraph: "igraph is a collection of network analysis tools with the emphasis on efficiency, portability and ease of use. igraph is open source and free. igraph can be programmed in R, Python, Mathematica and C/C+." Below the text are four buttons: "igraph R package", "python-igraph", "IGraph/M", and "igraph C library". The "Recent news" section features a list of updates on the left and a detailed announcement for "C/igraph 0.10.13" on the right, dated June 28th, 2024. The announcement states that this is the twelfth bugfix release of the 0.10 series, with several new additions, bug fixes, and performance improvements, and provides a link to the GitHub releases page.

igraph

Products News Forum Code of Conduct On GitHub

 igraph – The network analysis package

igraph is a collection of network analysis tools with the emphasis on **efficiency**, **portability** and ease of use. igraph is **open source** and free. igraph can be programmed in R, Python, Mathematica and C/C+.

[igraph R package](#) [python-igraph](#) [IGraph/M](#) [igraph C library](#)

Recent news

C/igraph 0.10.13
The igraph R package crossed the 2.0 threshold!
python-igraph 0.11.6
python-igraph 0.11.5
C/igraph 0.10.12
C/igraph 0.10.11

C/igraph 0.10.13
Jun 28th, 2024

C/igraph 0.10.13, the twelfth bugfix release of the 0.10 series, has arrived, with several new additions, bug fixes and performance improvements. As usual, the source can be obtained from [the GitHub releases page](#).

Programmatically
build and display
network graphs

Process to Draw Lineage from Metadata with iGraph

STEP A

R/Python Script

1. Get metadata directly from database
2. Parse metadata for edges
3. Parse metadata for nodes
4. Use iGraph package to create network
5. Colorize and plot network
6. Export to GRAPHML

STEP B

Map GRAPHML to yEd fields

STEP C

Layout and customize diagram

Why I am still using yEd?

I find it has better layouts and often I need to manually customize something.

R/Python Scripts

R

Python

```

1 library(readxl) #Import Excel files
2 library(RMySQL) #Connect to database
3 library(tidyverse) #Data manipulations
4 library(igraph) #Create the network data structure
5 library(visNetwork) #Visualize the network interactively
6
7 # 1. Read network components Excel file
8 df_components <- read_excel(file.choose())
9
10 ## OPTIONAL: 1. READ DIRECTLY FROM DATABASE
11 ## Query for network components (edges and nodes)
12 ## sql_components <- "
13 ## /* View dependencies */
14 ## SELECT DISTINCT name
15 ## , type
16 ## , referenced_name
17 ## , referenced_type
18 ## FROM all_dependencies
19 ## START WITH LOWER(name) LIKE '%vw_ods%'
20 ## CONNECT BY NOCYCLE PRIOR referenced_name = NAME
21 ## AND PRIOR referenced_owner = OWNER
22 ##
23 ## UNION
24 ##
25 ## /* Stand-alone tables */
26 ## SELECT table_name AS name
27 ## , 'TABLE' AS type
28 ## , NULL AS referenced_name
29 ## , NULL AS referenced_type
30 ## FROM all_tables
31 ## WHERE LOWER(table_name) LIKE '%tbl_ods%'
32 ## ;"
33
34 # Open database connection, query components, close database connection
35 # channel <- odbcConnect("DATABASE DSD", uid="USERID", pwd="PASSWORD")
36 # df_components <- sqlQuery(channel, sql_components)
37 # odbcClose(channel)
38
39 # 2. Parse the edges from the component data set
40 df_edge <- df_components %>%
41 filter(!is.na(REFERENCED_NAME)) %>%
42 select(REFERENCED_NAME, NAME)
43
44 # 3. Parse the nodes from the component data set
45 df_node <- df_components %>%
46 select(NAME, TYPE) %>%
47 dplyr::distinct(df_components %>%
48 filter(!is.na(REFERENCED_NAME))) %>%

```

```

1 from tkinter import *
2 import pandas as pd
3 import numpy as np
4 import igraph as ig
5 import matplotlib.pyplot as plt
6
7 # 1. Read network components Excel file
8 filename = askopenfilename()
9 df_components = pd.read_excel(filename)
10
11 # 2. Parse the edges from the component data set
12 df_edge = df_components.dropna() [{"REFERENCED_NAME", "NAME"}]
13
14 # 3. Parse the nodes from the component data set
15 df_n1 = df_components[["NAME", "TYPE"]]
16 df_n2 = df_components[["REFERENCED_NAME", "REFERENCED_TYPE"]]
17 df_n2 = df_n2.rename(columns = {"REFERENCED_NAME" : "NAME", "REFERENCED_TYPE" : "TYPE"})
18 df_n2 = df_n2.dropna()
19 df_node = pd.merge(df_n1, df_n2)
20 df_node = df_node.drop_duplicates()
21
22 # 3a. Convert TYPE variable from text to a number for each node
23 df_node["TYPE_NUMBER"] = np.where(df_node["TYPE"] == "TABLE", 1,
24 np.where(df_node["TYPE"] == "VIEW", 2,
25 np.where(df_node["TYPE"] == "FUNCTION", 3, 0)))
26
27 # 4. Create the network
28 net = ig.Graph(directed = True)
29 net.add_vertices(df_node["NAME"].tolist())
30 net.add_edges(list(zip(df_edge["REFERENCED_NAME"], df_edge["NAME"])))
31 net.vs["label"] = df_node["NAME"].tolist()
32
33 # 5. Define colors for nodes (V) and edges (E)
34 type_cols = { 1: "#3CB371",
35 2: "#4682B4",
36 3: "#FF69B4",
37 4: "#4169E1" }
38
39 cols = [type_cols[x] for x in df_node["TYPE_NUMBER"].tolist()]
40 net.vs["color"] = cols
41
42 # 5a. Plot a static image of the network
43 fig, ax = plt.subplots()
44 net.draw_networkx_static(ax, pos={}, labels=net.vs["label"], colors=cols)

```

CAUTION!
Be careful of database passwords stored in scripts!

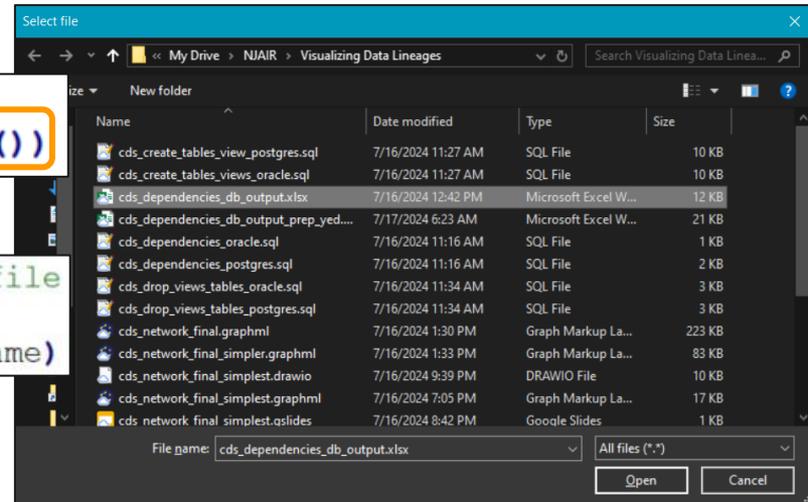
Forgive my Python...I am an R guy. 😊

STEP 1: Run the Script

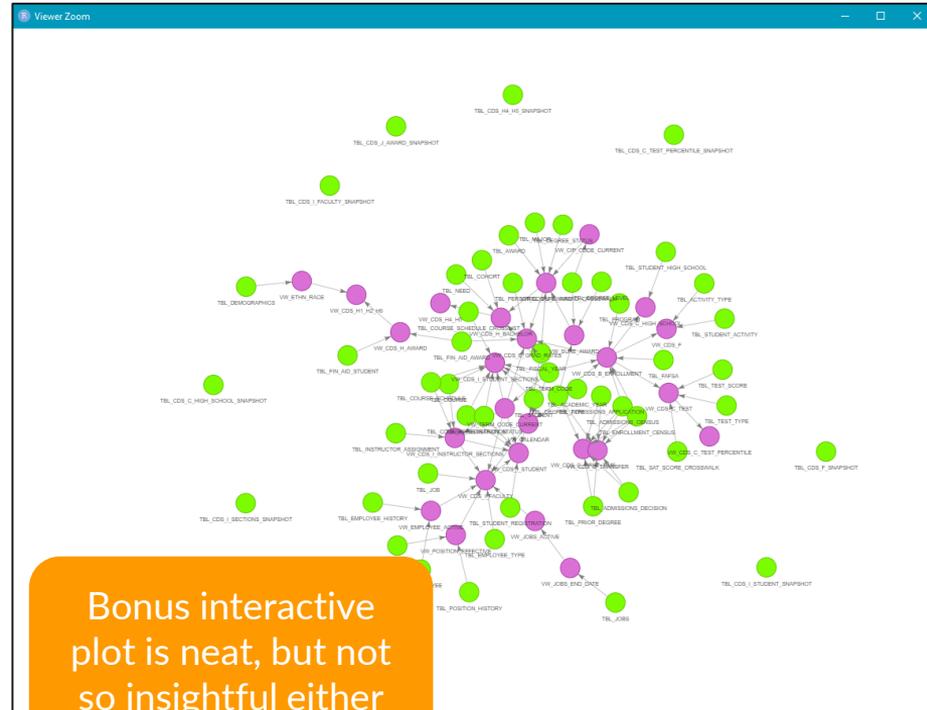
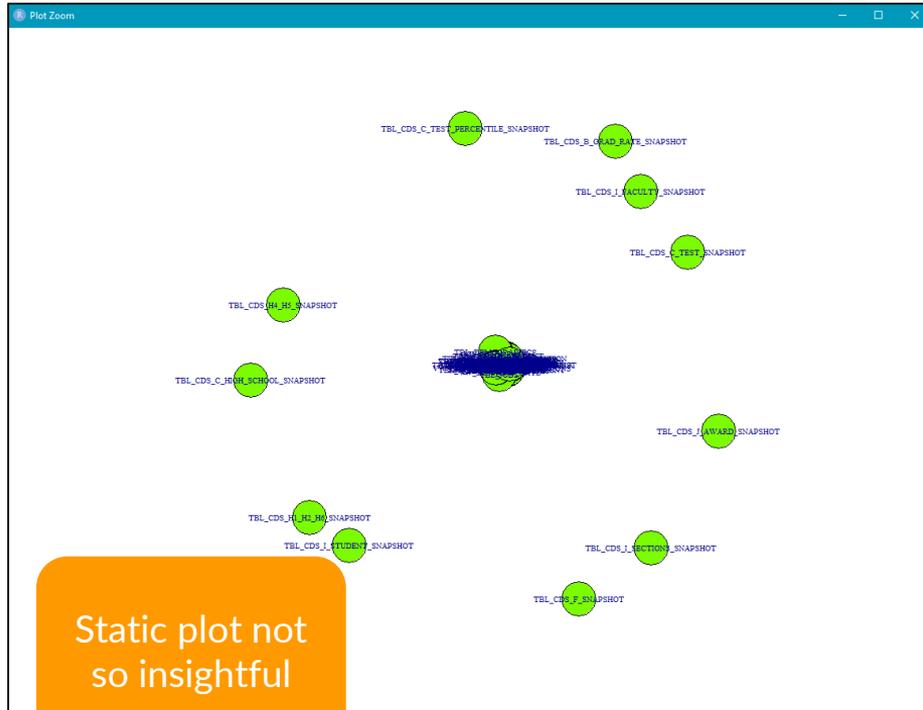
- For the sake of security, I will use Excel exports in this presentation and not db connections.
- The example scripts are meant to be **stepped through** not run all at once.
- **File dialogs** rather than file paths are coded to make file selection a more convenient.

```
# 1. Read network components Excel file  
df_components <- read_excel(file.choose())
```

```
# 1. Read network components Excel file  
filename = askopenfilename()  
df_components = pd.read_excel(filename)
```



STEP 1: Plot Output



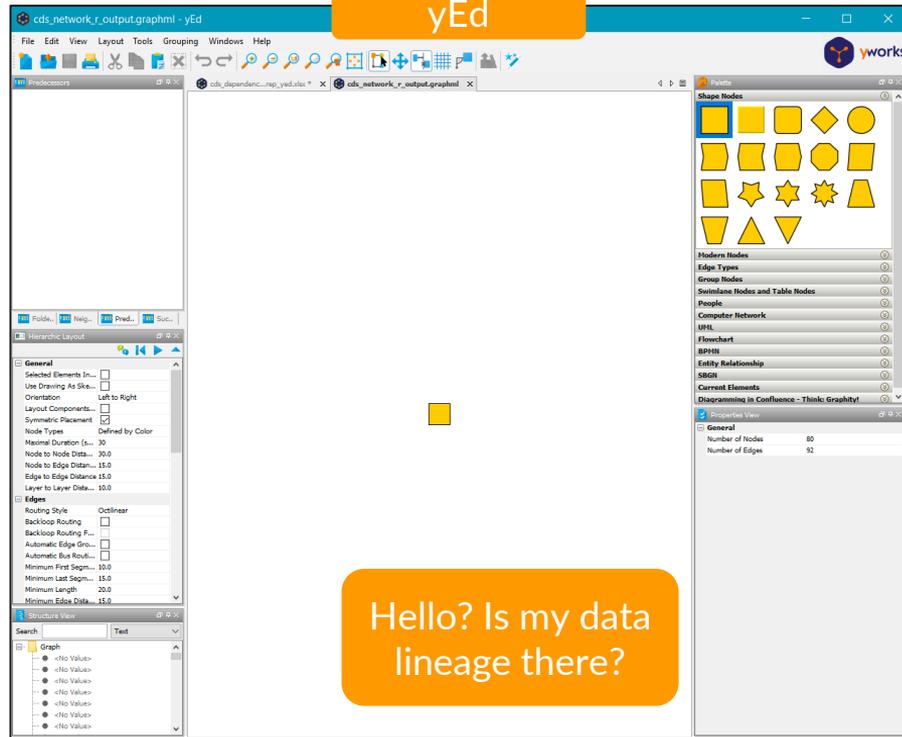
STEP 1: Send iGraph to GRAPHML File

Text Editor

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <graphml xmlns="http://graphml.graphdrawing.org/xmlns"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xsi:schemaLocation="http://graphml.graphdrawing.org/xmlns
5     http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd">
6   <!-- Created by igraph -->
7   <key id="v_name" for="node" attr.name="name" attr.type="string"/>
8   <key id="v_TYPE" for="node" attr.name="TYPE" attr.type="string"/>
9   <key id="v_TYPE_NUMBER" for="node" attr.name="TYPE_NUMBER" attr.type="double"/>
10  <key id="v_color" for="node" attr.name="color" attr.type="string"/>
11  <key id="e_color" for="edge" attr.name="color" attr.type="string"/>
12  <graph id="G" edgedefault="directed">
13    <node id="n0">
14      <data key="v_name">TBL_CDS_B_GRAD_RATE_SNAPSHOT</data>
15      <data key="v_TYPE">TABLE</data>
16      <data key="v_TYPE_NUMBER">1</data>
17      <data key="v_color">#7CFC00</data>
18    </node>
19    <node id="n1">
20      <data key="v_name">TBL_CDS_C_HIGH_SCHOOL_SNAPSHOT</data>
21      <data key="v_TYPE">TABLE</data>
22      <data key="v_TYPE_NUMBER">1</data>
23      <data key="v_color">#7CFC00</data>
24    </node>
25    <node id="n2">
26      <data key="v_name">TBL_CDS_C_TEST_PERCENTILE_SNAPSHOT</data>
27      <data key="v_TYPE">TABLE</data>
28      <data key="v_TYPE_NUMBER">1</data>
29      <data key="v_color">#7CFC00</data>
30    </node>
31    <node id="n3">
32      <data key="v_name">TBL_CDS_C_TEST_SNAPSHOT</data>
33      <data key="v_TYPE">TABLE</data>
34      <data key="v_TYPE_NUMBER">1</data>
35      <data key="v_color">#7CFC00</data>
36    </node>
37    <node id="n4">
38      <data key="v_name">TBL_CDS_F_SNAPSHOT</data>
39      <data key="v_TYPE">TABLE</data>
40    </node>
41  </graph>
42 </graphml>
```

GRAPHML is just a type of XML file

yEd



Hello? Is my data lineage there?

STEP 2: Map GRAPHML to yEd fields

Properties Mapper

Configurations

Selected Configuration

Name: New Configuration

Template: None Single Multiple Icon

Mappings: Act On Selection Only

Data Source	Map To	Conversion
name	Label Text	Automatic
color	Fill Color	Automatic

Selected Mapping

Map To Label No.: 1

Conversion Details (color -> Fill Color)

The selected conversion requires no configuration. It converts hexadecimal RGB and RGBA values to colors, for example '#ff5000' to orange.

Ok Apply Cancel Help

Map color variable to Fill Color

Click Edit > Properties Mapper to get this dialog

mi - yEd

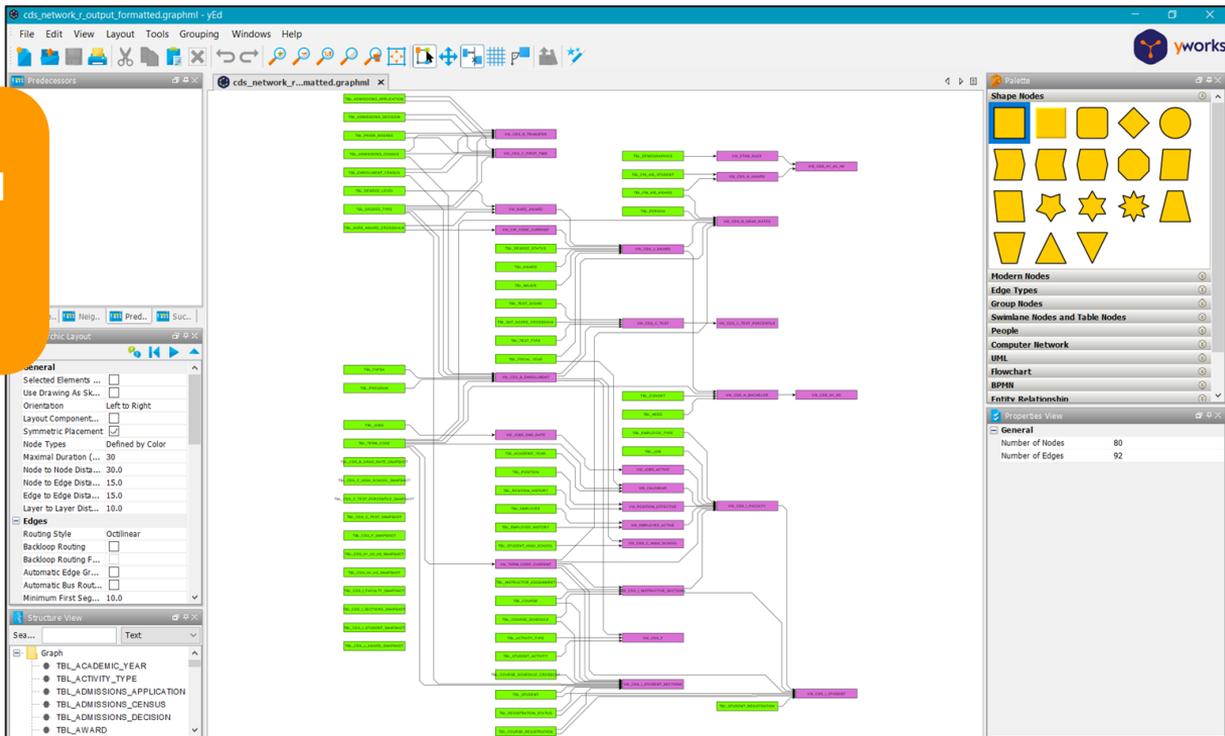
Grouping Windows Help

Nodes

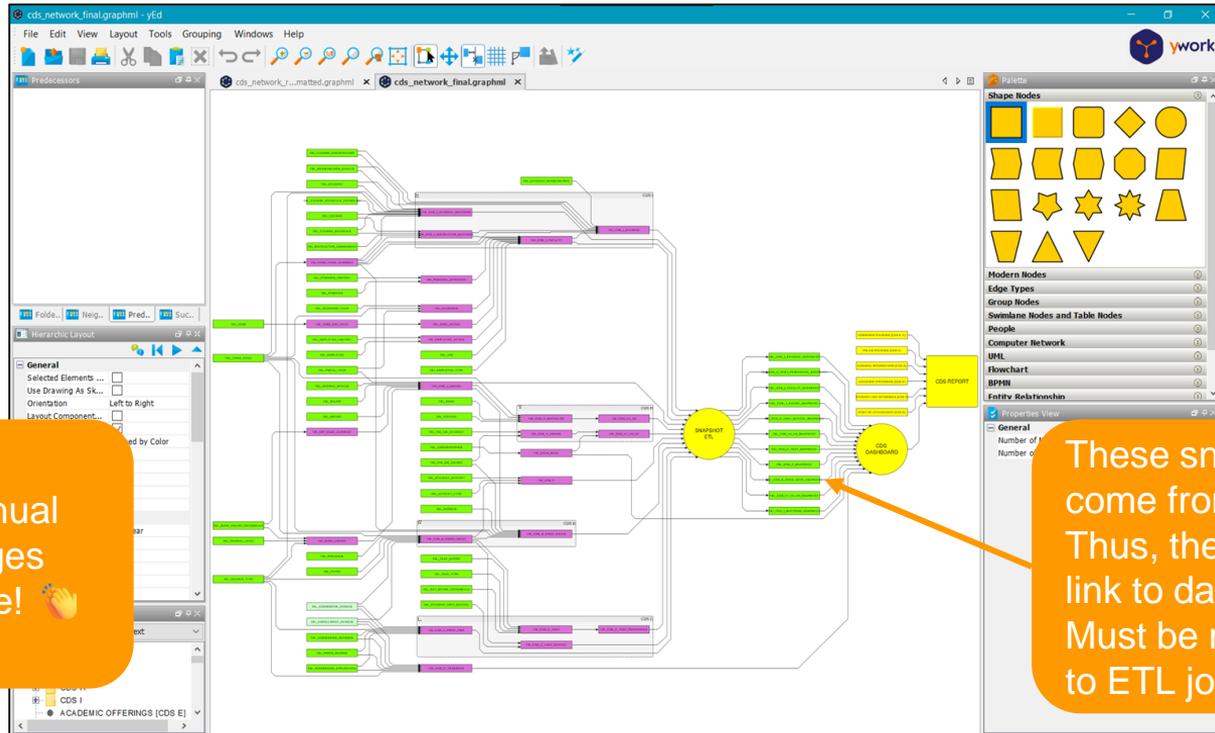
Looking promising

STEP 3: Layout and customize diagram

Chose
Hierarchical
from the
Layout
menu.



STEP 3: Layout and customize diagram



Add some manual nodes and edges and we're done! 🙌

These snapshot tables come from an ETL job. Thus, they do not auto-link to database views. Must be manually linked to ETL job node.

Before we leave Tier 4: SQL FLOW

<https://sqlflow.gudusoft.com/#/>

- There are times when I want to look at 500+ lines SQL code and understand the data lineage within that file.
- I have not found solid free or open source tools that address this situation, but SQL Flow (a commercial tool) looks like it could.
- My institution has not purchased this tool, nor does it intend to. I have no real experience with it...just passing along what I have seen on my data lineage journey.

The screenshot displays the SQLFlow application interface. On the left, a code editor shows SQL code for creating a view named 'vsal' and inserting data into various tables. The code includes subqueries for employee counts and salaries, and a complex insert statement that categorizes orders based on total amount. On the right, a data lineage diagram visualizes the execution of this SQL. It shows a central 'SELECT' operation connected to several source tables: 'employees', 'small_orders', 'medium_orders', 'large_orders', and 'special_orders'. The diagram also shows the flow of data into 'small_orders', 'medium_orders', 'large_orders', and 'special_orders' tables, and finally into a 'SELECT' operation that joins these with 'customers' and 'employees' tables. The interface includes a top navigation bar with 'SQLFlow', 'Table lineage', 'ER diagram', and 'Text output' options, and a bottom status bar with a question mark icon.

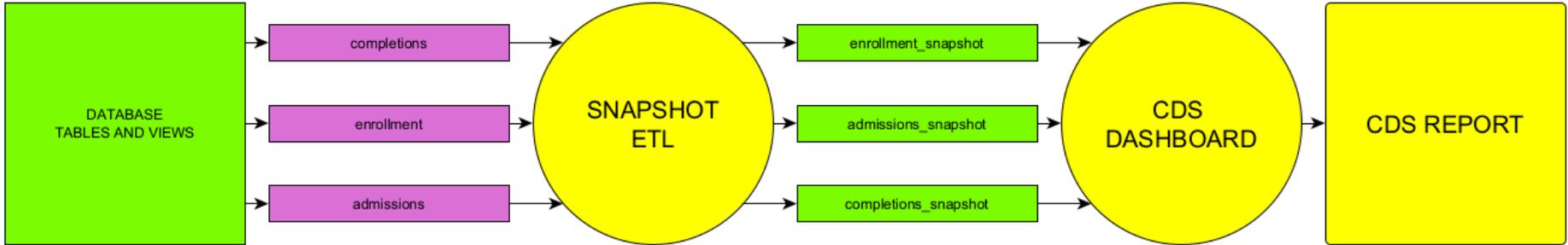


Tier 5

Continuous lineage monitoring

What Have We Done To This Point?

Essentially we have built a scanner that retrieves the metadata from a single database and displays it as a network diagram.



Can I handle multiple database and database links?



Can I parse the metadata lineage in my ETL tool?



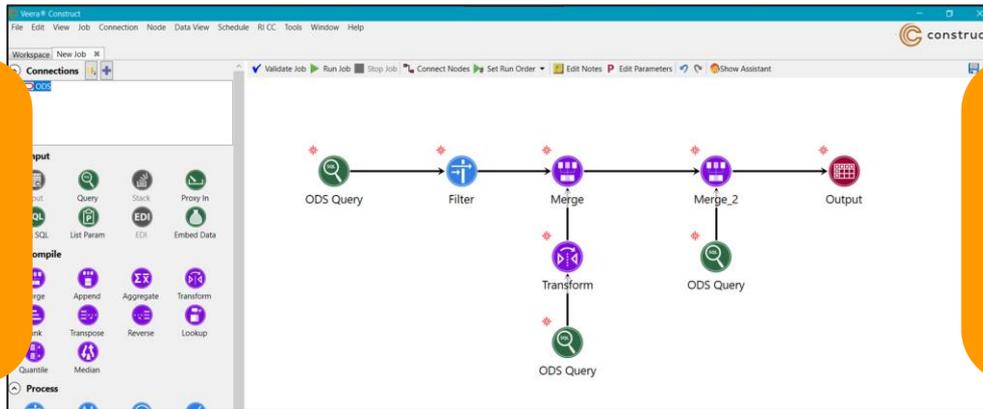
Can I parse the metadata lineage in my Dashboard?



A Question I Regularly Think About

- **At what layer in my report lineage do I make a transformation/calculation?**
 - It would be convenient if we only had one layer or one tool
 - Our ETL tool is very visual...unfortunately it has limitations

We Use Rapid
Insight Veera
Construct for ETL
<https://eab.com/solutions/rapid-insight/>



Funny note: My color scheme for tables and views in the presentation is based the colors in Veera Construct.

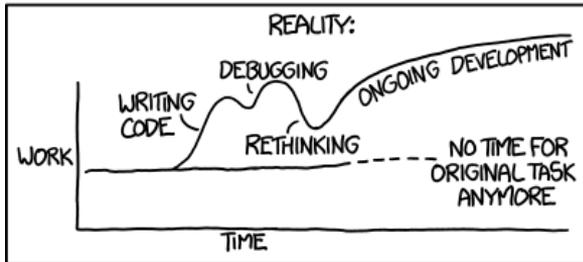
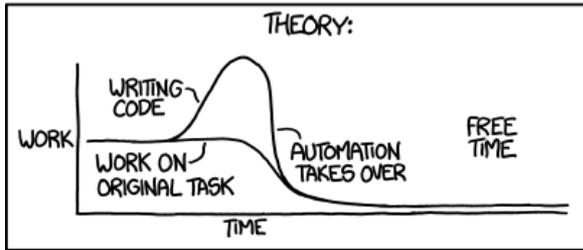


What Am I Still Missing

- Cross-platform data lineage
- Column-level data lineage
- Linking my technical data terms to business data terms

Automation

"I SPEND A LOT OF TIME ON THIS TASK.
I SHOULD WRITE A PROGRAM AUTOMATING IT!"



It can be a full-time
job picking apart

METADATA

<https://xkcd.com/1319/>

xkcd by Randall Munroe

Informatica

Google-like search bar makes finding data assets so convenient!

Rowan University began implementing Informatica in 2024 as data catalog, data governance, and data quality tool

Informatica Data Governance and Catalog

Start typing search terms for suggestions.

Rowan University

Rowan Dashboard

View Only

Help for Getting Started

- [Search & Browse for Assets](#)
- [Create Assets](#)
- [Workflows & Tickets](#)
- [Understanding Business Assets](#)
- [Understanding Technical Assets](#)
- [Data Lineage](#)
- [Bulk Import Business Assets](#)

Governance Assets

372	231	127
Business Term	Data Element Clas...	Metric
30	22	13
Process	Tableau Workbook	Domain
13	13	4
Project	Data Quality Rule ...	Data Set
3	2	1

CLAIRE Activity

powered by CLAIRE

11.4K

ASSETS WITH RECOMMENDATIONS

RECOMMENDATION TYPE

RECOMMENDATION STATUS

Technical Assets

32

Catalog Source

Business Term by CDE

Informatica: Common Data Set Asset Relationships

The screenshot displays the Informatica Data Governance and Catalog interface. The main content area shows the 'Common Data Set' asset, which is a 'BUSINESS TERM'. The 'Relationships' tab is active, showing a central 'Common Data...' node connected to four categories: 'Process' (2 items), 'Business Term' (2 items), 'Domain' (1 item), and 'Data Set' (1 item). A 'Metric' node with 47 items is also connected to the central node. On the right, a list of related data sets is displayed, including 'CDS 1-1E - Instr...', 'CDS 1-1J - Instr...', 'CDS C9 First-tim...', 'CDS 1-2 - Stude...', 'CDS C9 First-tim...', 'CDS C10-Studen...', 'CDS 1-3 - Under...', 'CDS G4 - Tuition ...', 'CDS C11 Studen...', and 'CDS 11 - Finance'.

Informatica Data Governance and Catalog

Start typing search terms for suggestions.

Rowan University

Institutional Research /

Common Data Set
BUSINESS TERM

OVERALL RATING: ★★★★★ (1) Add Rating

CRITICAL DATA ELEMENT: NO

LIFECYCLE: PUBLISHED

LAST UPDATED: Apr 15, 2024, 1:01 PM

Overview Hierarchy Relationships Data Quality Stakeholders Tickets History

Find

Process 2 Business Term 2 Domain 1 Data Set 1 Metric 47

Common Data...

CDS 1-1E - Instr...
CDS 1-1J - Instr...
CDS C9 First-tim...
CDS 1-2 - Stude...
CDS C9 First-tim...
CDS C10-Studen...
CDS 1-3 - Under...
CDS G4 - Tuition ...
CDS C11 Studen...
CDS 11 - Finance

Collapse

CDS is related to domains, terms, processes, data sets, and metrics

Informatica: CDS I2 Lineage from Tableau Dashboard

The screenshot displays the Informatica Data Governance and Catalog interface. The main view is titled "CDS I2 - Student To Faculty Ratio" and shows a data lineage diagram. The diagram starts with source data sets on the left, including "RACE", "ETHN_RACE", "DASHBOARDMGR", "CDS_I_FACULTY_SNAPSHOT", and "CDS_I_STUDENT_SNAPSHOT". These feed into a "Tableau" data set, which then feeds into a "CDS Dashboard" data set, and finally into a "Reporting - CDS Section I2 - ..." data set. The lineage ends with "Annual Report Sections H-J".

Below the lineage diagram, there is a section titled "I-2 Student to Faculty Ratio" with a description: "Report the Fall 2023 ratio of full-time equivalent students (full-time plus 1/3 part time) to full-time equivalent instructional faculty (full time plus 1/3 part time). In the ratio exclude both faculty and students in stand-alone graduate or professional programs such as medicine, law, veterinary, dentistry, social work, business, or public health in which virtually only graduate level students." Below this description, the ratio is shown as "Student to Faculty Ratio 16 to 1".

Ratio is based on number of students:	15,136
Ratio is based on number of faculty:	944

CDS I2 Student-to-faculty ratio

Informatica: CDS I2 Lineage Expanded

Yellow represents propagation of faculty assignment through lineage

CDS I2 Student-to-faculty ratio in Tableau Dashboard



Database

ETL
Lineage

Snapshot
Table

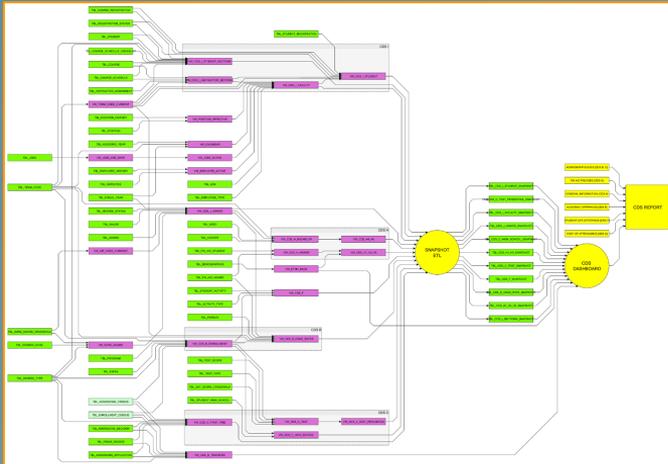
Tableau
Lineage



Informatic at Rowan University

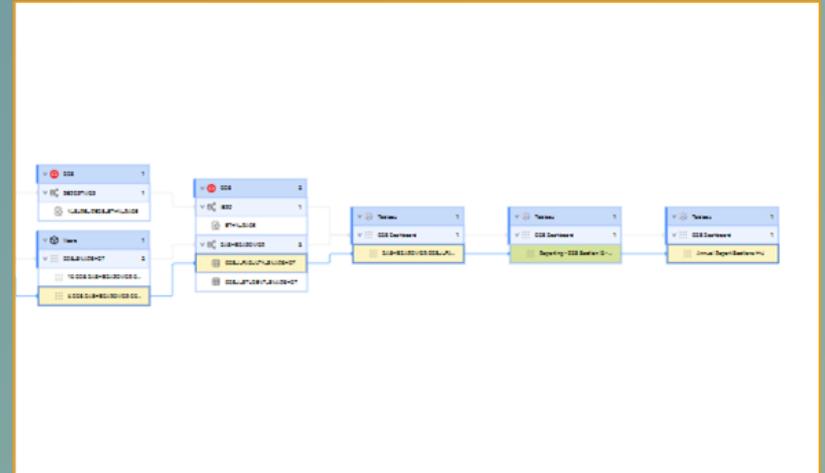
- This is just the beginning
- Still lots to learn
- Lots of potential
 - Align business terms and technical terms
 - Track and consolidate reporting data sets
 - Ensure that data quality meets specified threshold for reporting
 - Assign stakeholders to review underlying data or approve metrics for reporting

What does this mean for the non-continuous lineage we have been drawing?



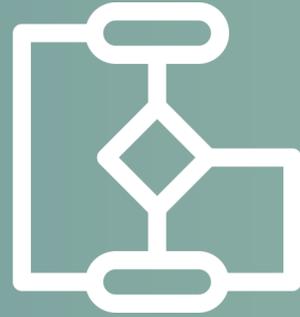
Design and Intent

VS



Monitoring Reality

Thank You



Please send questions and comments to manleyd@rowan.edu

