A different type of DAG:
Data pipeline principles for epidemiology

Emily Riederer
Data Science Lead at Capital One

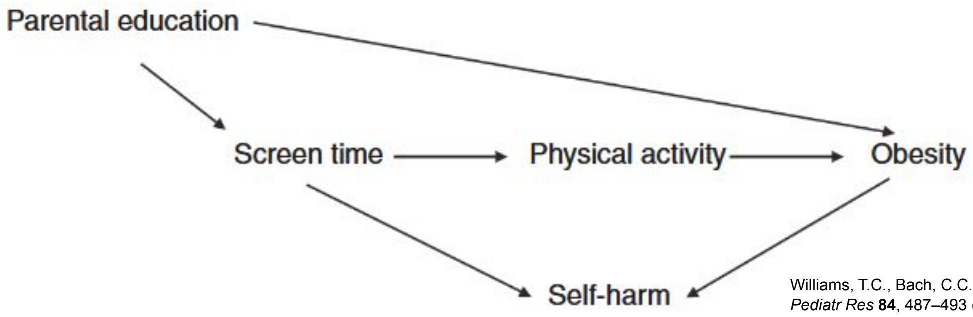
Disclosures: No relevant financial interests exist.

Industry data science can learn a lot from epidemiology

	Data	Analytical Approach	Precise Communication
Epi	Imperfection expected	Prototypical study designs	Estimands
DS	Data Management	Packaged Code	Deployed Outputs

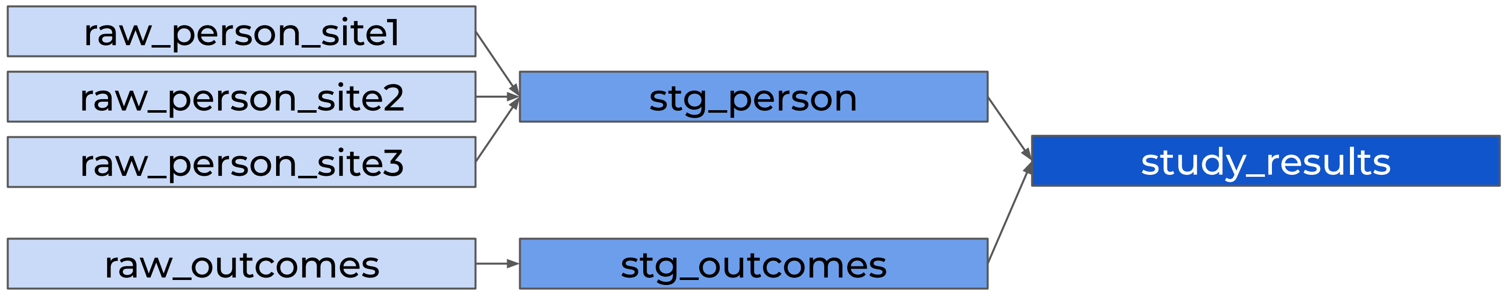
Data pipelines are a different sort of DAG that encode the computational versus conceptual data generating process

Epi



Williams, T.C., Bach, C.C., Matthiesen, N.B. *et al.* Directed acyclic graphs: a tool for causal studies in paediatrics. *Pediatr Res* **84**, 487–493 (2018). https://doi.org/10.1038/s41390-018-0071-3

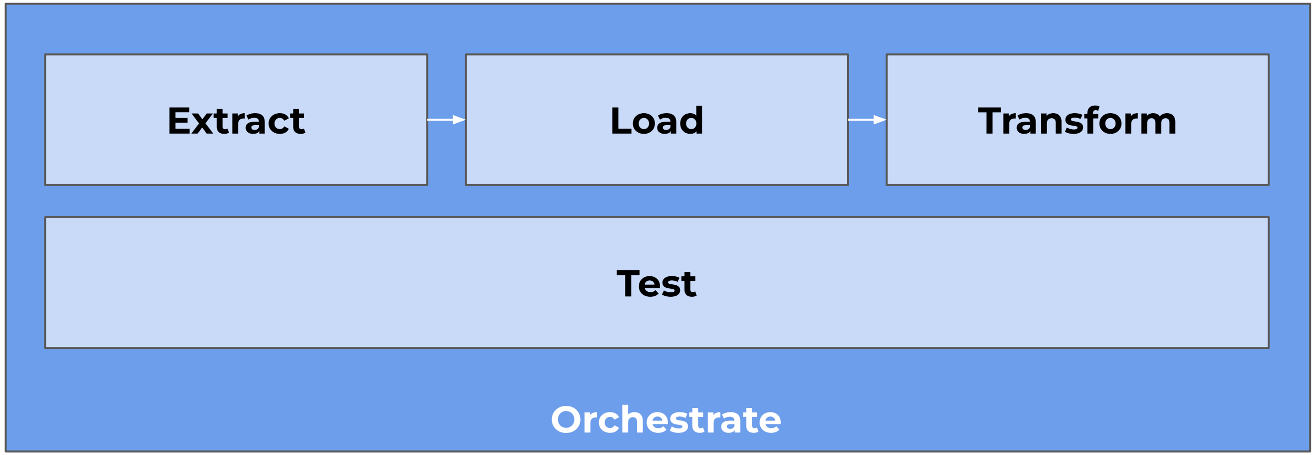
DS



Data pipelines and databases serve many needs in industry with parallels for individual analytical projects

Data Pipelines	**Project Benefit**
● Data lineage and quality control	● Reproducibility
● New data ingestion, integration, and updates	● Harmonizing disparate sources
● Sharing data artifacts across many users	● Collaboration
● Sharing data artifacts across many use cases	● Reuse across projects
● Produce complex outputs (models, reports)	● In-sync analysis

Today, we will discuss some useful tools and ideas throughout the data pipeline ecosystem

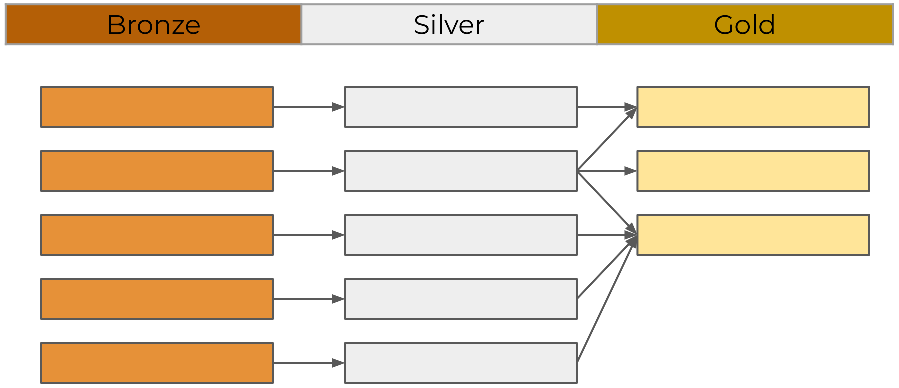


ARCHITECTURE: The same principles apply across platforms – all you need to get started is a hard drive

	Local	Shared	Cloud
Storage	📁 Local Drive	📁 Network Drive ⊟ Google Cloud Storage ⊟ Amazon S3	⊟ Google BigQuery DB
Compute	R, python, SQL + dbt, or any scripting language ⊟ duckdb		⊟ Google BigQuery SQL + dbt (data pipeline) Dagster (orchestration)

ARCHITECTURE: Pipelines preserve data artifacts at raw, standardized, and transformed stages

Medallion Architecture



Bronze (= Raw)

- Unmodified source data

Silver (= Structure)

- Standardized file structure
- Cleaned column name
- Casted data types
- Light re-encoding

Gold (= Prepared)

- Transformed and merged data
- Derived fields
- Ready for analytical use

ARCHITECTURE: Preserving artifacts at multiple stages balances reproducibility with analytical efficiency

In Database Schema

- 🗄 `data.raw`
- 🗄 `data.structured`
- 🗄 `data.prepared`

In File System

- 📁 `data/raw/`
- 📁 `data/structured/`
- 📁 `data/prepared/`

Raw

- Preserves full reproducibility
- Enables testing for upstream errors pre-transformation

Structured

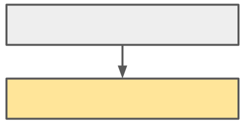
- Preserves data flexible enough to transform in different ways

Prepared

- Ready for use without rerunning expensive computations
- Ensures collaborators made same design decisions

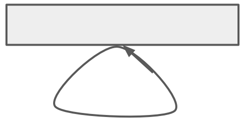
ELT
TO

ARCHITECTURE: Read/write patterns make the pipelines acyclic, idempotent, resilient to re-runs



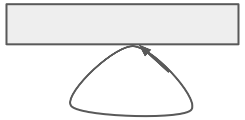
```
df = readr::read_csv('data/raw/persons.csv')

df_cleaned = dplyr::filter(df, dt_enroll < cutoff)

readr::write_csv(df_cleaned, 'data/structured/persons.csv')
```

♻



```
df = readr::read_csv('data/raw/persons.csv')

df = dplyr::filter(df, dt_enroll < cutoff)

readr::write_csv(df, 'data/raw/persons.csv')
```

X



```
df = readr::read_csv('data/raw/persons.csv')

df_cleaned = dplyr::filter(df, dt_enroll < cutoff)

readr::write_csv(df_cleaned, 'data/raw/persons.csv')
```

X

ELT
TO

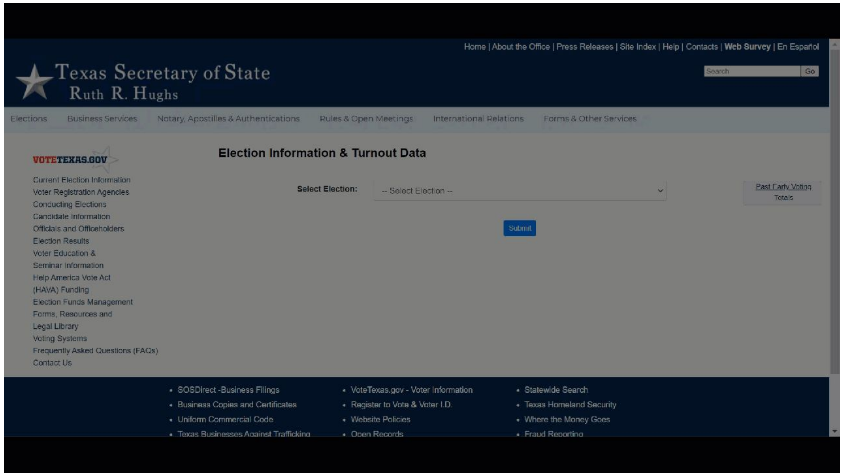
EXTRACT: Many different techniques can be used to collect data across disparate and sometimes inconvenient sources

When data is available as…	Method
Link to a standalone file, e.g. in an S3 bucket	**Download:** R {readr} or python {urllib.requests}
Exposed through a REST API	**API Call:** R {httr2} or python {requests}
Rendered server-side in the webpage's source, e.g. a static HTML table	**Scraping:** Extract content from underlying webpage structure
Rendered dynamically in a web-based UI	**Headless Browsing:** Use code to navigate a UI
Trapped within a complex UI or report	**Computer Vision:** Parse characters from an image or webpage based on visual characteristics

Increasing Complexity

EXTRACT: Automation can save manual data collection effort

Example:
Headless Browsing with Playwright



Home | About the Office | Press Releases | Site Index | Help | Contacts | **Web Survey** | En Español

Texas Secretary of State
Ruth R. Hughs

Search Go

Elections Business Services Notary, Apostilles & Authentications Rules & Open Meetings International Relations Forms & Other Services

VOTETEXAS.GOV

Current Election Information
Voter Registration Agencies
Conducting Elections
Candidate Information
Officials and Officeholders
Election Results
Voter Education &
Seminar Information
Help America Vote Act
(HAVA) Funding
Election Funds Management
Forms, Resources and
Legal Library
Voting Systems
Frequently Asked Questions (FAQs)
Contact Us

Election Information & Turnout Data

Select Election: -- Select Election --

Submit

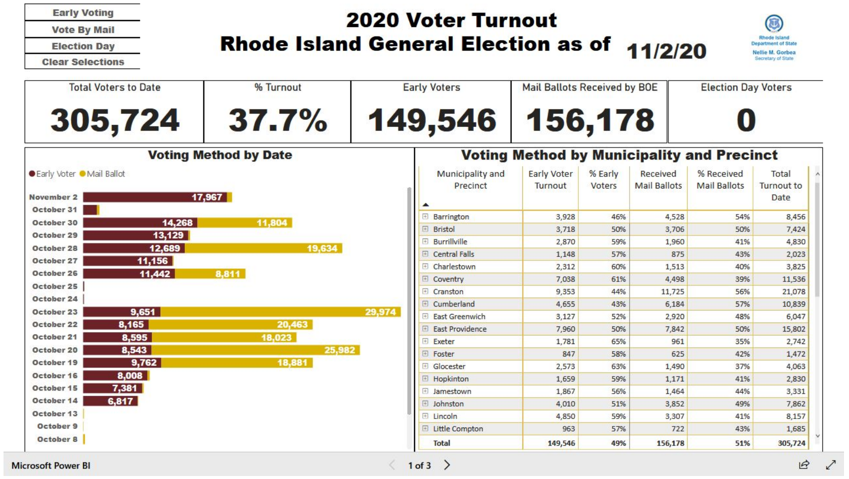
Past Early Voting Totals

- SOSDirect -Business Filings
- Business Copies and Certificates
- Uniform Commercial Code
- Texas Businesses Against Trafficking
- VoteTexas.gov - Voter Information
- Register to Vote & Voter I.D.
- Website Policies
- Open Records
- Statewide Search
- Texas Homeland Security
- Where the Money Goes
- Fraud Reporting

```
# pick selections from menu
  page.goto(url)
  page.select_option('#idElection',
    label = "2020 NOVEMBER 3RD GENERAL ELECTION")
  page.click('#electionsInfoForm button')
  page.wait_for_selector('#selectedDate')

  page.select_option('#selectedDate', value = target_date)
  page.click('#electionsInfoForm button:nth-child(2)')
  page.wait_for_selector('"Generate Statewide Report"')

# download report
  with page.expect_download() as download_info:
    page.click('"Generate Statewide Report"')
  download = download_info.value
  download.save_as(target_file)
```

EXTRACT: Automation can save manual data collection effort

Example:
Optical Character Recognition with Tesseract



Early Voting
Vote By Mail
Election Day
Clear Selections

2020 Voter Turnout
Rhode Island General Election as of 11/2/20

Rhode Island Department of State
Nellie M. Gorbea
Secretary of State

Total Voters to Date	% Turnout	Early Voters	Mail Ballots Received by BOE	Election Day Voters
305,724	37.7%	149,546	156,178	0

Voting Method by Date

Voting Method by Municipality and Precinct

Municipality and Precinct	Early Voter Turnout	% Early Voters	Received Mail Ballots	% Received Mail Ballots	Total Turnout to Date
⊞ Barrington	3,928	46%	4,528	54%	8,456
⊞ Bristol	3,718	50%	3,706	50%	7,424
⊞ Burrillville	2,870	59%	1,960	41%	4,830
⊞ Central Falls	1,148	57%	875	43%	2,023
⊞ Charlestown	2,312	60%	1,513	40%	3,825
⊞ Coventry	7,038	61%	4,498	39%	11,536
⊞ Cranston	9,353	44%	11,725	56%	21,078
⊞ Cumberland	4,655	43%	6,184	57%	10,839
⊞ East Greenwich	3,127	52%	2,920	48%	6,047
⊞ East Providence	7,960	50%	7,842	50%	15,802
⊞ Exeter	1,781	65%	961	35%	2,742
⊞ Foster	847	58%	625	42%	1,472
⊞ Glocester	2,573	63%	1,490	37%	4,063
⊞ Hopkinton	1,659	59%	1,171	41%	2,830
⊞ Jamestown	1,867	56%	1,464	44%	3,331
⊞ Johnston	4,010	51%	3,852	49%	7,862
⊞ Lincoln	4,850	59%	3,307	41%	8,157
⊞ Little Compton	963	57%	722	43%	1,685
Total	**149,546**	**49%**	**156,178**	**51%**	**305,724**

Microsoft Power BI   1 of 3

```
# take screenshot with Playwright
  page.goto(url)
  page.wait_for_load_state(state = 'networkidle')
  time.sleep(30)
  page.screenshot(path = 'ri.png')
```

```
# parse image to data with Tesseract
img = cv2.imread('ri.png')
text = pytesseract.image_to_string(img)
n_tot = re.search('Turnout\n\n(\d+)', text).group(1)
n_mail = re.search('Mail Ballots', text).group(1)
```

LOAD: Despite being the most common data storage formats, CSVs and Excel files can be brittle for computing

zip_code <str>	treated <int>	name <str>
‘06830’	1	Emily
‘27514’	1	Barret,M.

write_csv(‘persons.csv’)
read_csv(‘persons.csv’)

Wrong type inferred, leading zeros ignored

zip_code <int>	treated <int>	name <str>
6830	1	Emily
27514	1	Barret,M.

Could break file or lose data

BONUS:
DuckDB’s CSV Sniffer can help identify and quarantine bad rows that keep you from opening a file

LOAD: Parquet files are like a zipfile of data and metadata

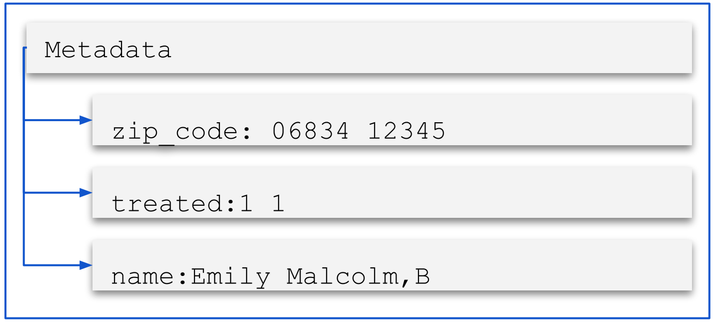
CSV

- Human readable and editable
- All data in single text file
- Row-based

```
zip_code, treated, name
06834, 1, Emily
12345, 1, Barret,M
```

Parquet

- Compressed and machine readable
- Bundles data and metadata
- Column-based



```
Metadata
  zip_code: 06834 12345
  treated:1 1
  name:Emily Malcolm,B
```

LOAD: Parquet has numerous advantages over CSV as a standard data format

1	**Similar scripting read/write**	Easily handled by R’s {readr} or {arrow}
2	**Lower risk of accidental modification**	Not easily opened in UI or modified without explicit read/write
3	**Preserves data types**	Remembers column types to avoid data corruption
4	**Storing strategies allow efficient reads**	Allows for partitioned read/writes for pre-filtering of large files
5	**Smaller file sizes**	Compressed on disk

LOAD: Partitioning improves data organization and aids in efficient reading of large files

```
arrow::write_dataset(
  data,
  path = 'data/',
  format = "parquet",
  partitioning = "year"
)
```

```
📁data/year=2021/
📁data/year=2022/
📁data/year=2023/
```

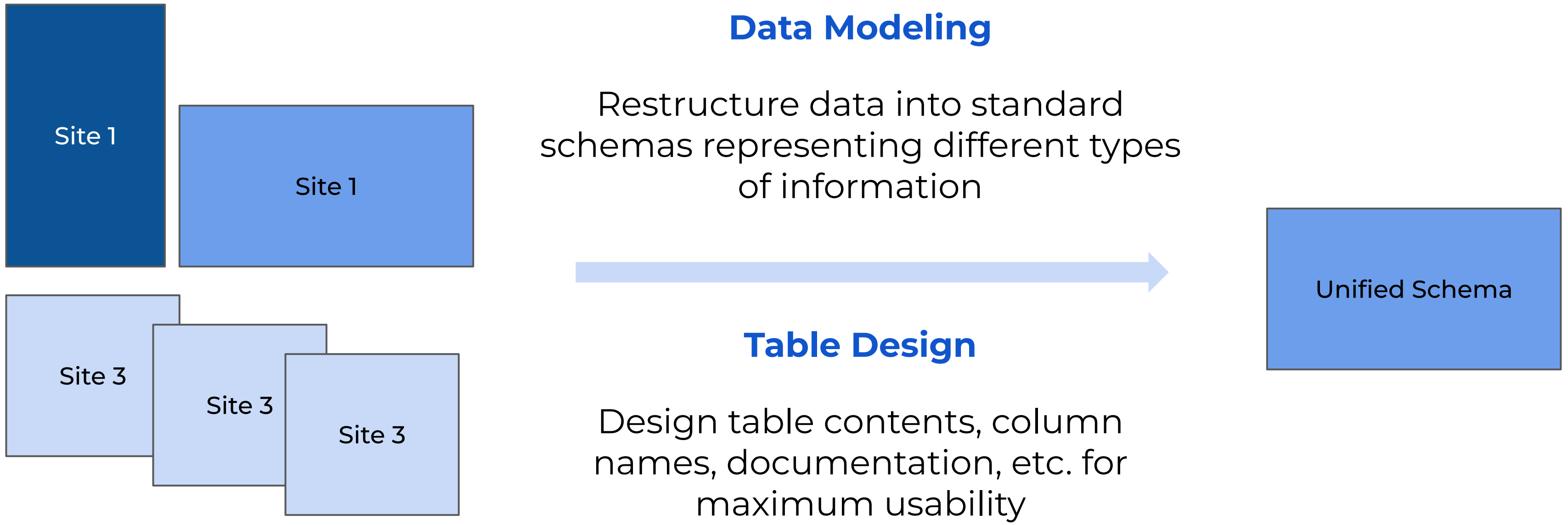
```
arrow::open_dataset(
  'data'
)
```

```
📁data/year=2021/
📁data/year=2022/
📁data/year=2023/
```

```
arrow::open_dataset(
  'data/year=2022/'
)
```

```
📁data/year=2021/
📁data/year=2022/
📁data/year=2023/
```

TRANSFORM: Data modeling provides a framework for integrating data from diverse sources and structures



Data Modeling

Restructure data into standard schemas representing different types of information

Table Design

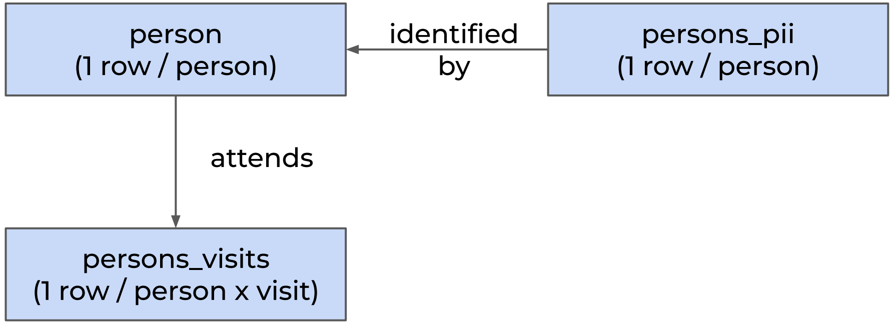
Design table contents, column names, documentation, etc. for maximum usability

TRANSFORM: Data modeling principles can help us partition information into entities and relationships

id	visit	test_result	name	age
1	2025-01-05	13	Chris	35
1	2025-01-10	1	Chris	35
1	2025-01-12	2	Chris	35
2	2025-01-05	14	Jenny	27
2	2025-02-10	0	Jenny	27

id	age
1	35
2	27

id	name
1	Chris
2	Jenny



id	visit	test_result
1	2025-01-05	13
1	2025-01-10	1
1	2025-01-12	2
2	2025-01-05	14
2	2025-02-10	0

TRANSFORM: Keep the simplest data structures as long as possible ("shift right") to preserve flexibility

Activity Schema

- 1 record per entity x event
- Long and narrow
- Flexible but may require more processing

id_person	date	event
1	2025-01-05	treatment
2	2025-01-05	treatment
3	2025-01-05	treatment
3	2025-03-14	death
4	2025-01-05	treatment
4	2025-02-10	withdraw

One Big Table (OBT)

- 1 record per entity
- Wide
- Already imposes some analytical decisions

id_person	treated	withdrew	death
1	2025-01-05		
2	2025-01-05		
3	2025-01-05		2025-03-14
4	2025-01-05	2025-02-10	

TRANSFORM: Keep the simplest data structures as long as possible (“shift right”) to preserve flexibility

id_person	date	event
1	2025-01-05	treatment
1	2025-01-12	death
2	2025-01-05	treatment
3	2025-01-05	treatment
4	2025-01-05	treatment
4	2025-02-10	withdraw

Flexible to query different inclusion/exclusion criteria

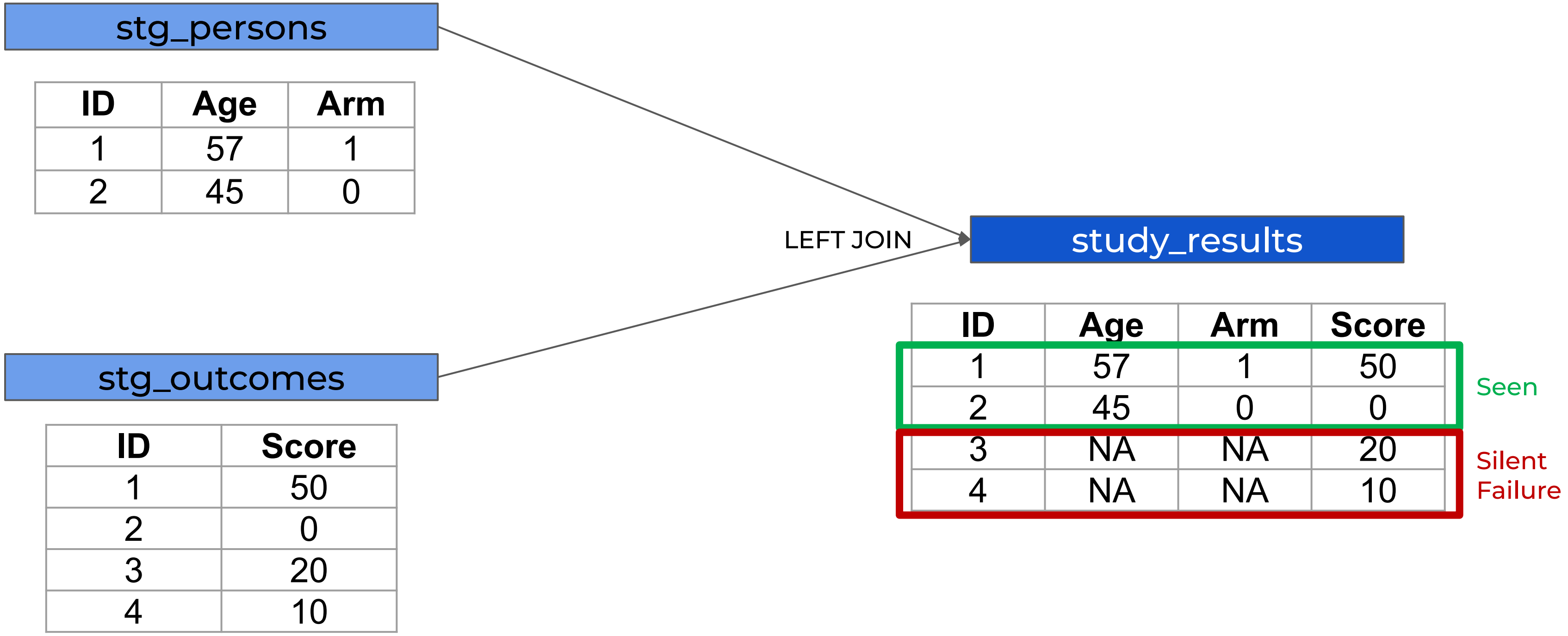
Easily updated with new records (favor appends to joins)

TRANSFORM: Consistent column naming conventions across tables aids comprehension and downstream analysis code

Stub	Semantics
ID	Unique entity identifier
IND / IS	Binary 0/1 indicator; rest of name describes 1 condition
BIN	Binary 0/1 indicator; rest of name describes 1 condition
N	Count of quantity or event occurrences
AMT	Sum-able real number amount (“denominator free”)
…	…

```
library(dplyr)

data |>

  group_by(NM_STATE) |>

  summarize(

    across(starts_with("IND"), mean),

    across(contains("_ACTL_"), sum)

  )

#> # A tibble: 51 x 4
#>   NM_STATE     IND_COUNTY_HPSA  N_CASE_ACTL  N_DEATH_ACTL
#>   <chr>                  <dbl>        <dbl>         <dbl>
#> 1 Alabama                0.149       455582          7566
#> 2 Alaska                 0.235        51338           250
#> 3 Arizona                0           753379         13098
```

TEST: A multistage pipeline allows us to conduct hypothesis-drive upstream data quality checks (“shift left”)



stg_persons

ID	Age	Arm
1	57	1
2	45	0

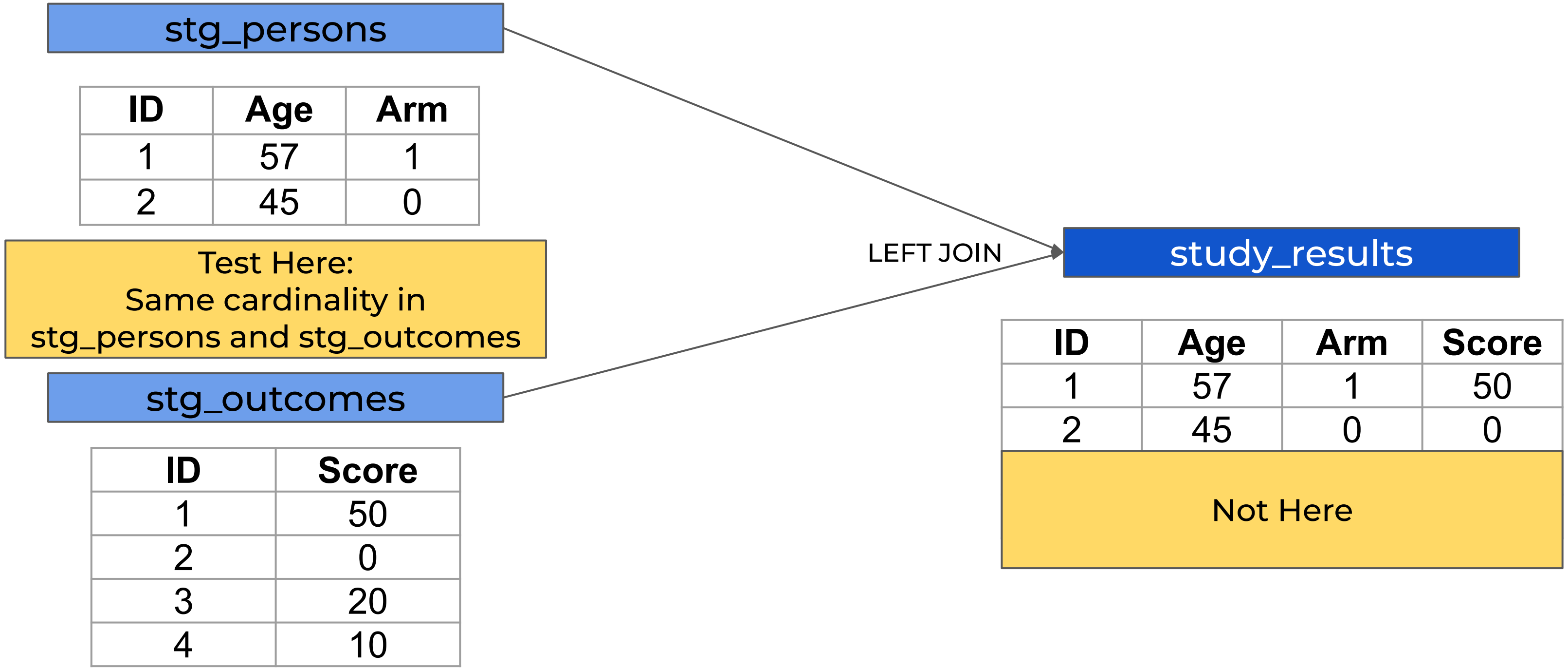
stg_outcomes

ID	Score
1	50
2	0
3	20
4	10

study_results

ID	Age	Arm	Score
1	57	1	50
2	45	0	0
3	NA	NA	20
4	NA	NA	10

TEST: A multistage pipeline allows us to conduct hypothesis-drive upstream data quality checks (“shift left”)



stg_persons

ID	Age	Arm
1	57	1
2	45	0

Test Here:
Same cardinality in
stg_persons and stg_outcomes

stg_outcomes

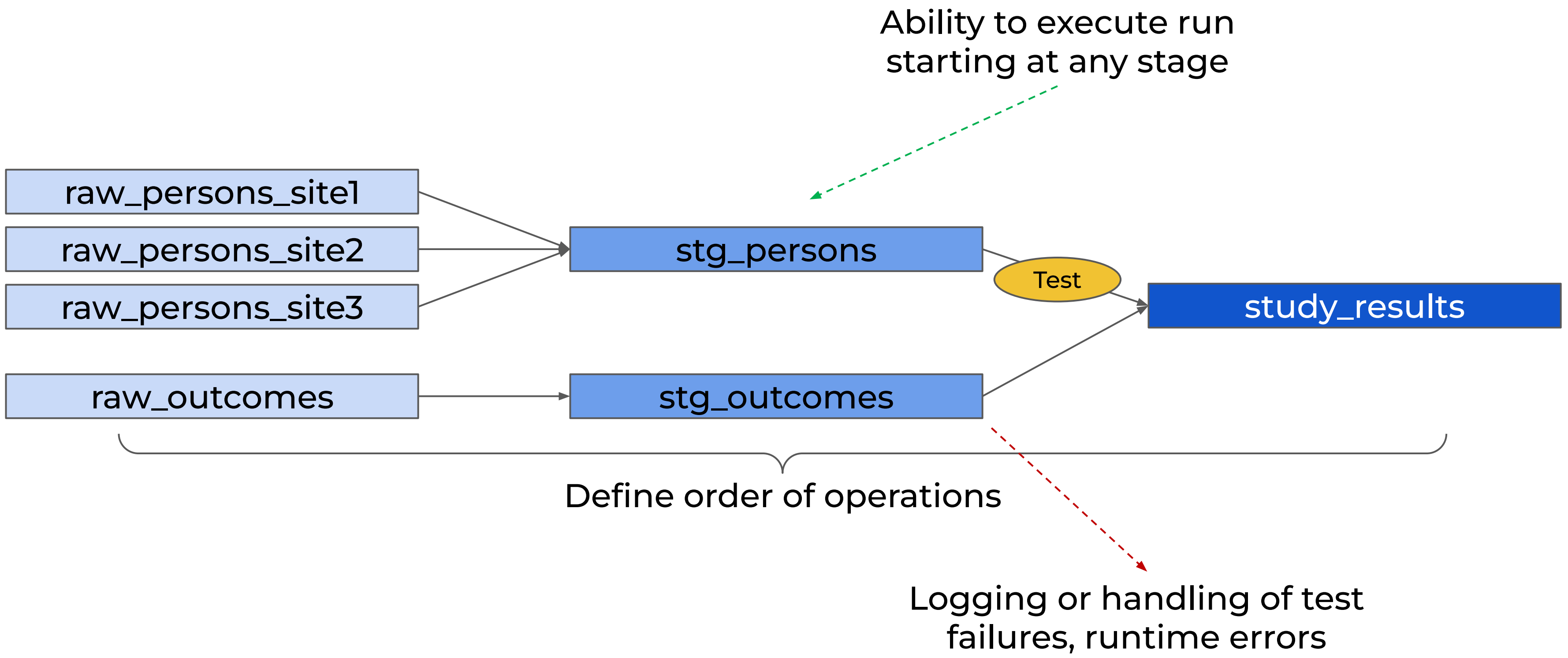
ID	Score
1	50
2	0
3	20
4	10

study_results

ID	Age	Arm	Score
1	57	1	50
2	45	0	0

Not Here

ORCHESTRATE: Orchestration tools help us run many steps of an analysis pipeline in order to keep dependencies consistent


Ability to execute run starting at any stage
raw_persons_site1
raw_persons_site2
raw_persons_site3
raw_outcomes
stg_persons
stg_outcomes
Test
study_results
Define order of operations
Logging or handling of test failures, runtime errors

ORCHESTRATE: Many different tools exist for orchestration

makefile

- General purpose config file
- Runs any shell commands based on user-defined dependencies
- Higher learning curve

```
all: ./data/clean.json

./data/raw.json: ./data/raw.json ./ingest.py
	mkdir -p data
	./ingest.py

./data/clean.json: ./data/raw.json ./clean.py
	./clean.py
```

dbt

- SQL framework that infers dependencies
- Compatible with any database including local compute (duckdb)
- Lower learning curve but less flexible

```
select id, name
from raw.persons
where dt_enroll = '2025-01-01'
```

persons.sql

```
select id, name, outcome
from {{ref('persons')}}
  left join
  {{ref('outcomes')}}
  using (id)
```

results.sql

Makefile example from https://datasciencesouth.com/blog/make/#a-make-data-pipeline-example
See also R package {targets} and Dagster

ELT
TO

Big ideas from data pipelining can help improve data handling in projects of any size of complexity

Architecture	Start with a structured read/write process to your hard drive controlled by **R, python,** or **SQL**
Extract	Automate manual extraction with tools like playwright and tesseract
Load	Store and share **Parquet** files for type-safe resilience
Transform	Apply data modeling techniques like entity-relationship mapping, Boyce-Codd normalization and the Activity schema to standardize structure
Test	Test intermediate artifacts to detect failures at their source
Orchestrate	Adopt automated tooling to run all steps in sync with make**,** dbt**,** or targets

A few more examples and resources are available online

Architecture	Build NC data pipeline using local drive, duckdb, and Arrow
Extract	Different web scraping techniques across 6 government websites
Load	
Transform	Column Names as Contracts Column Name Contracts in dbt
Test	Understanding the data (error) generating process Hypothesis-driven data testing with grouped checks
Orchestrate	

Thank you!