Building an end-to-end open source data platform for clinical data

Alberto Labarga

PyData London 2023

bsc-health-data / **pydatalondon23-modern-data-stack** Public

alabarga Update README.md `cd3d156` 5 minutes ago **50** commits

database	Add database DOckerfile	7 months ago
datahub	Add datahub	7 months ago
dremio	Add files	7 months ago
meltano	Create meltano.yml	2 days ago
obevo	Add obevo schemas	7 months ago
slides	Add files via upload	5 months ago
synthea	Update README.md	2 days ago
.gitattributes	Add obevo	7 months ago
.gitignore	Initial commit	7 months ago
LICENSE	Initial commit	7 months ago
README.md	Update README.md	5 minutes ago
TUTORIAL.md	Update TUTORIAL.md	2 days ago

https://github.com/bsc-health-data/pydatalondon23-modern-data-stack

Telecommunications Engineer
Working as Head of Health Data @BSC
More than 20 years teaching

open data - open source – open science

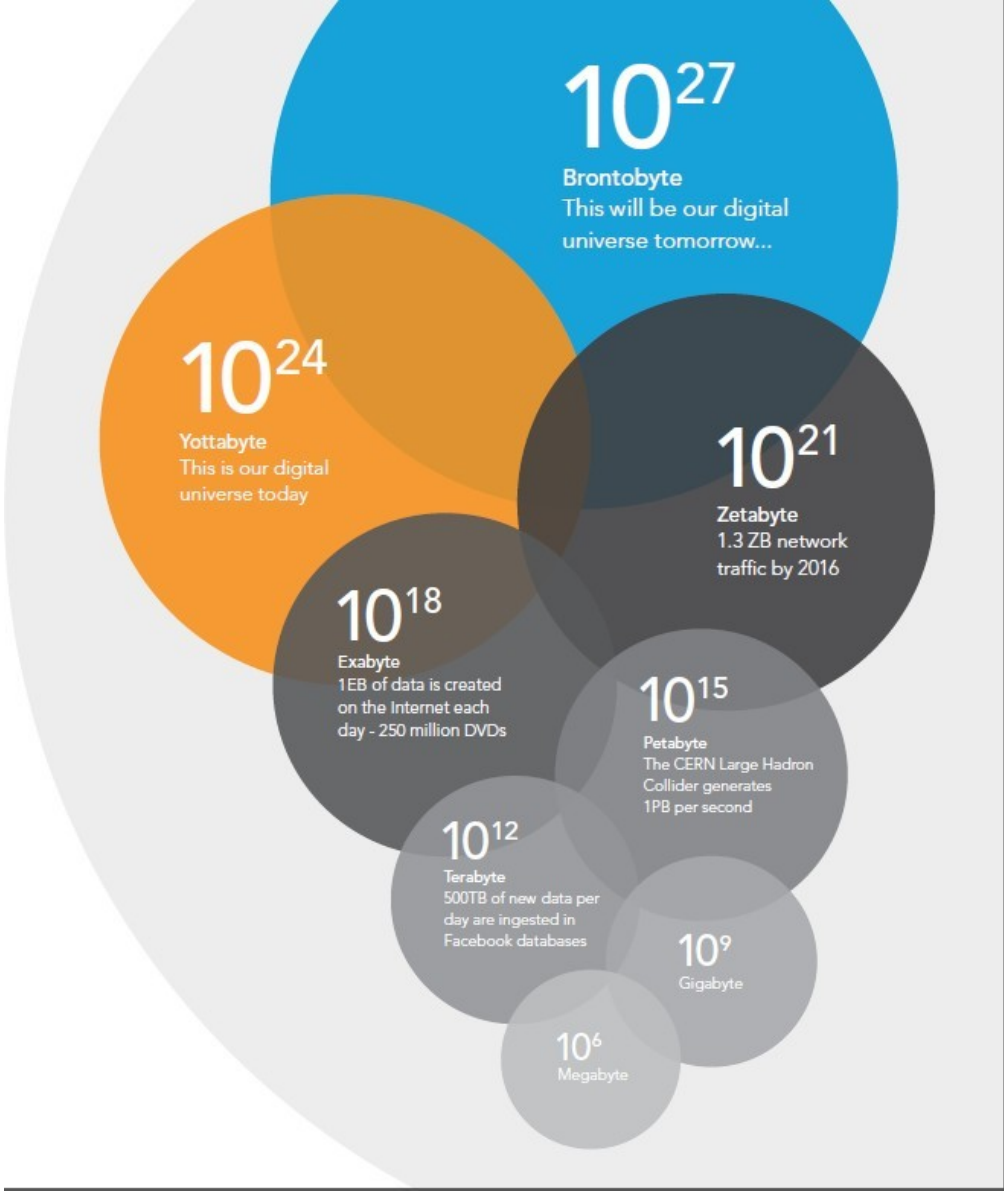
I am Alberto

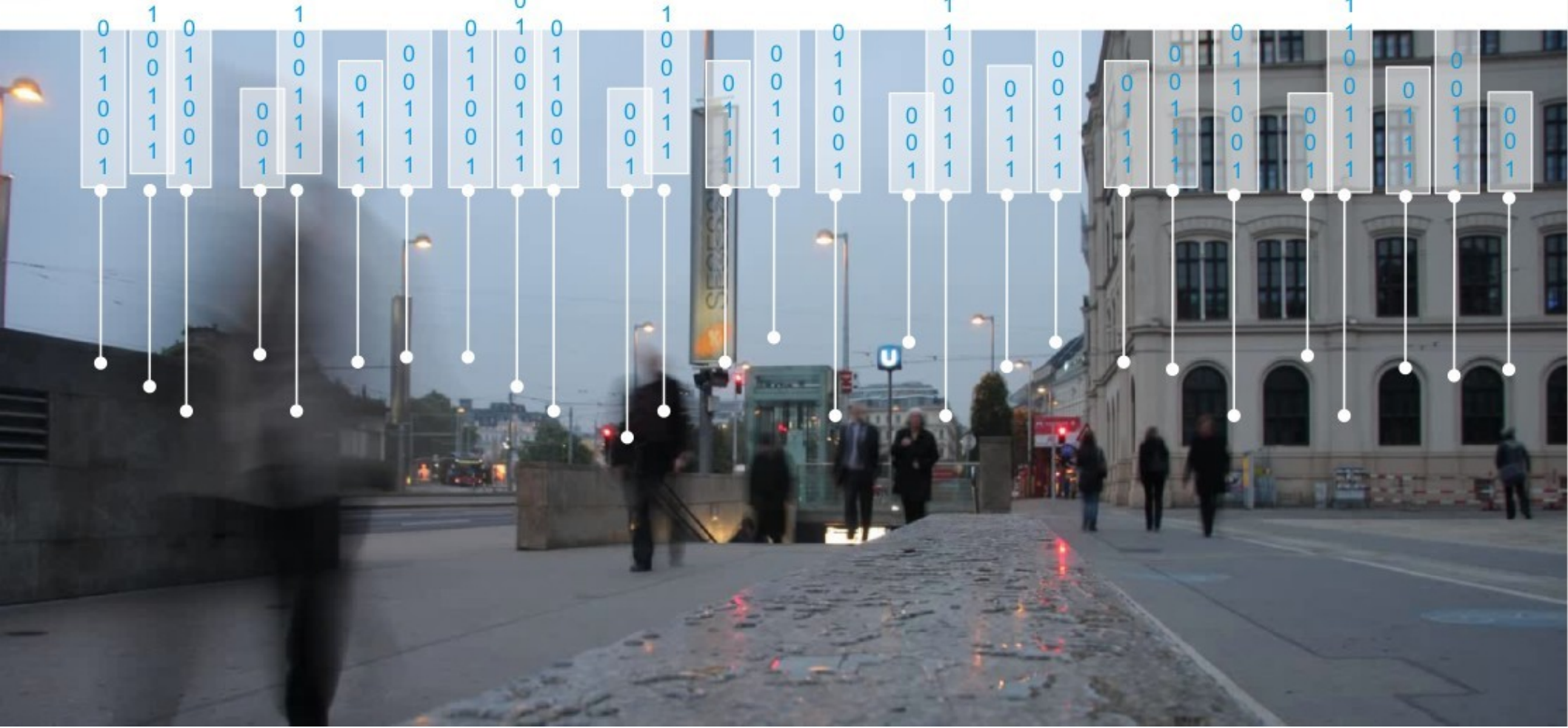
alabarga
alabarga
/in/albertolabarga

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Data is big

My God, it’s full of data!


10^{27}
Brontobyte
This will be our digital universe tomorrow...
10^{24}
Yottabyte
This is our digital universe today
10^{21}
Zetabyte
1.3 ZB network traffic by 2016
10^{18}
Exabyte
1EB of data is created on the Internet each day - 250 million DVDs
10^{15}
Petabyte
The CERN Large Hadron Collider generates 1PB per second
10^{12}
Terabyte
500TB of new data per day are ingested in Facebook databases
10^{9}
Gigabyte
10^{6}
Megabyte



A
B

CLOSED

OPEN

FAIR

What is FAIR DATA?

Data and supplementary materials have sufficiently rich metadata and a unique and persistent identifier.

FINDABLE

Metadata and data are understandable to humans and machines. Data is deposited in a trusted repository.

ACCESSIBLE

Metadata use a formal, accessible, shared, and broadly applicable language for knowledge representation.

INTEROPERABLE

Data and collections have a clear usage licenses and provide accurate information on provenance.

REUSABLE

Data is the new oil

Supreme
PWS
Regular
93
89
87

Data Scientist
VS
Data Engineer
VS
Data Analyst

Modern data stack

Modern Data Stack

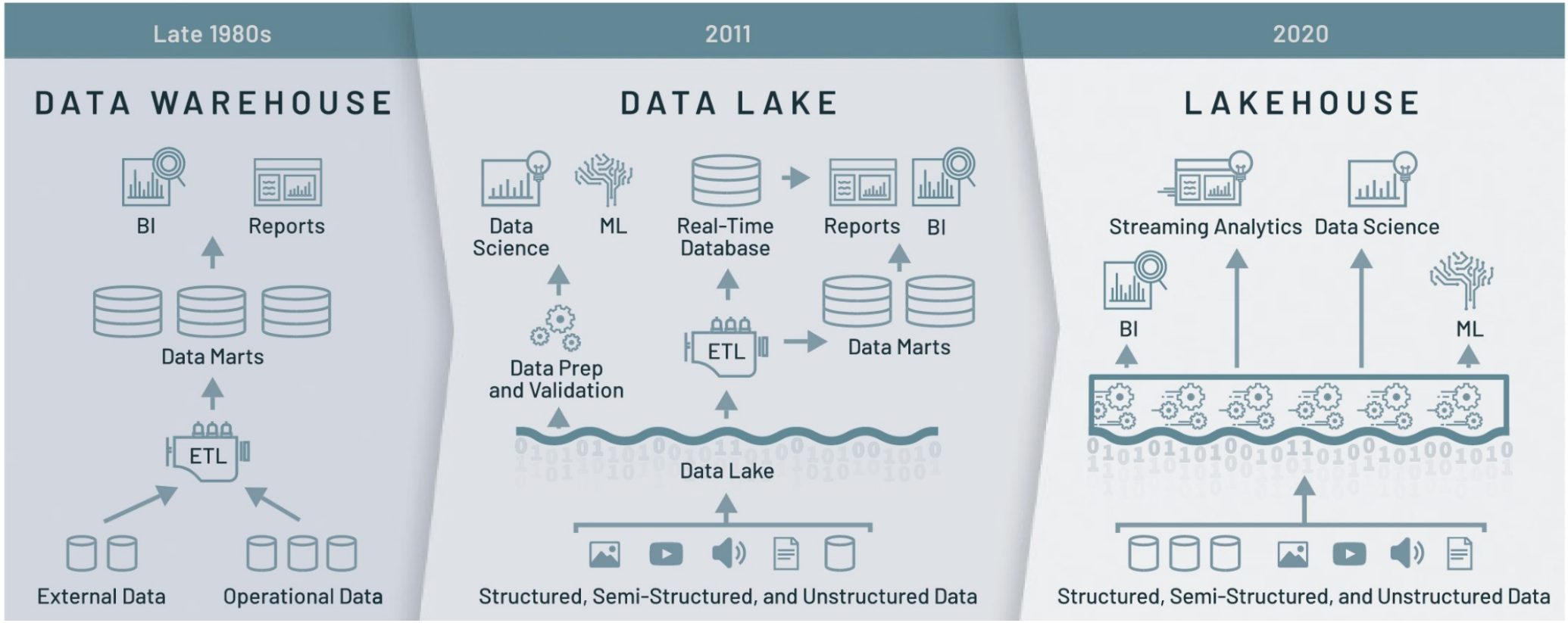
- **Infraestructure as code**

HELM

Build
Release
Deploy
build
release
deploy_produc...
deploy_staging

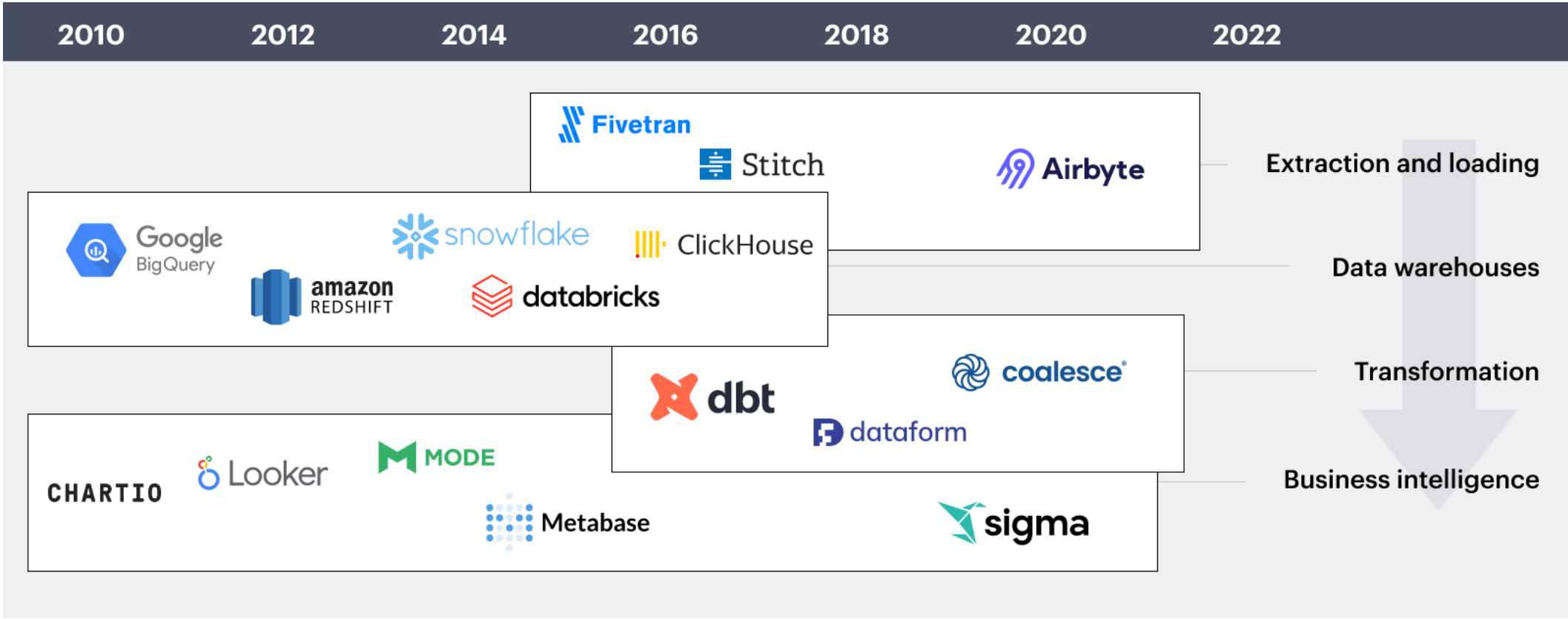
Modern Data Stack

- Infraestructure as code
- **From Datawarehouse to Lakehouse**


Late 1980s
2011
2020
DATA WAREHOUSE
BI
Reports
Data Marts
ETL
External Data
Operational Data
DATA LAKE
Data Science
ML
Real-Time Database
Reports
BI
Data Prep and Validation
ETL
Data Marts
Data Lake
Structured, Semi-Structured, and Unstructured Data
LAKEHOUSE
Streaming Analytics
Data Science
BI
ML
Structured, Semi-Structured, and Unstructured Data

Modern Data Stack

- Infraestructure as code
- From Datawarehouse to Lakehouse
- **From ETL to ELT**

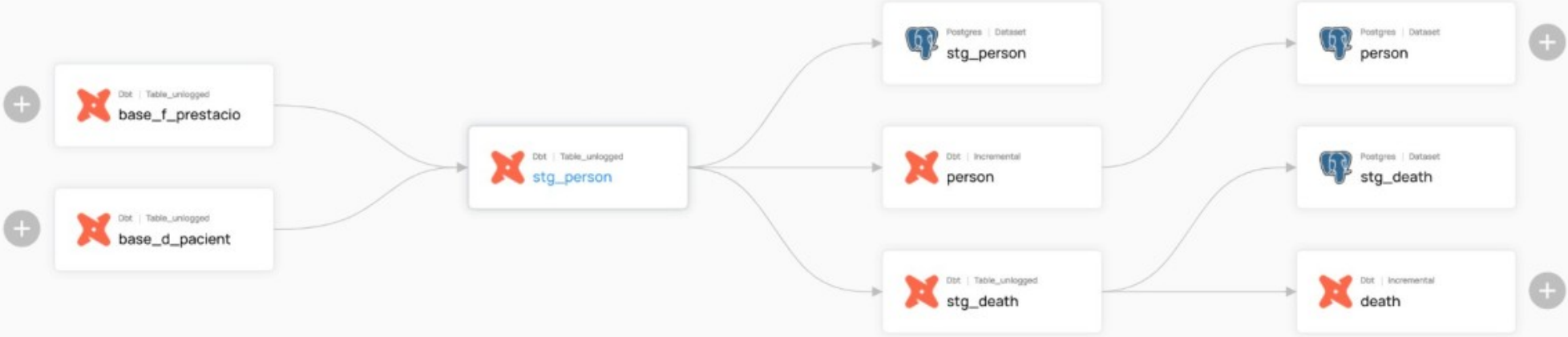


Adoption of the **Modern Data Stack**

Helps data teams pull data from sources (**extraction and loading**) into one place (**warehouses**), transform it into a usable form (**transformation**), and then give access with accessible tools (**business intelligence**).

Modern Data Stack

- Infraestructure as code
- From Datawarehouse to Lakehouse
- From ETL to ELT
- **Data governance and observability**



Accuracy
The degree to which the data correctly represents the entity or attribute being described.

Completeness
The percentage of missing data from a given data set.

Consistency
The absence of difference or contradiction in data irrespective of the data's source.

Validity
Invalid data affects the accuracy and completeness of a data set.

Integrity
The validity of relationships across various data entities.

Uniqueness
Ensures duplicate or overlapping data is identified and marked.

Modern Data Stack

- Infraestructure as code
- From Datawarehouse to Lakehouse
- From ETL to ELT
- Data governance and observability
- **Self-service analytics**

PRODUCTION APPMED

Search by Title...

Recent

PERSON

Person analysis

Domain Visit

DRUGS (QA)

EDA: DRUG_EXPOSURE (H12O)

Favorites ↓U@

No items in this section.

Lists

OMOP

All My Docs ↓U@

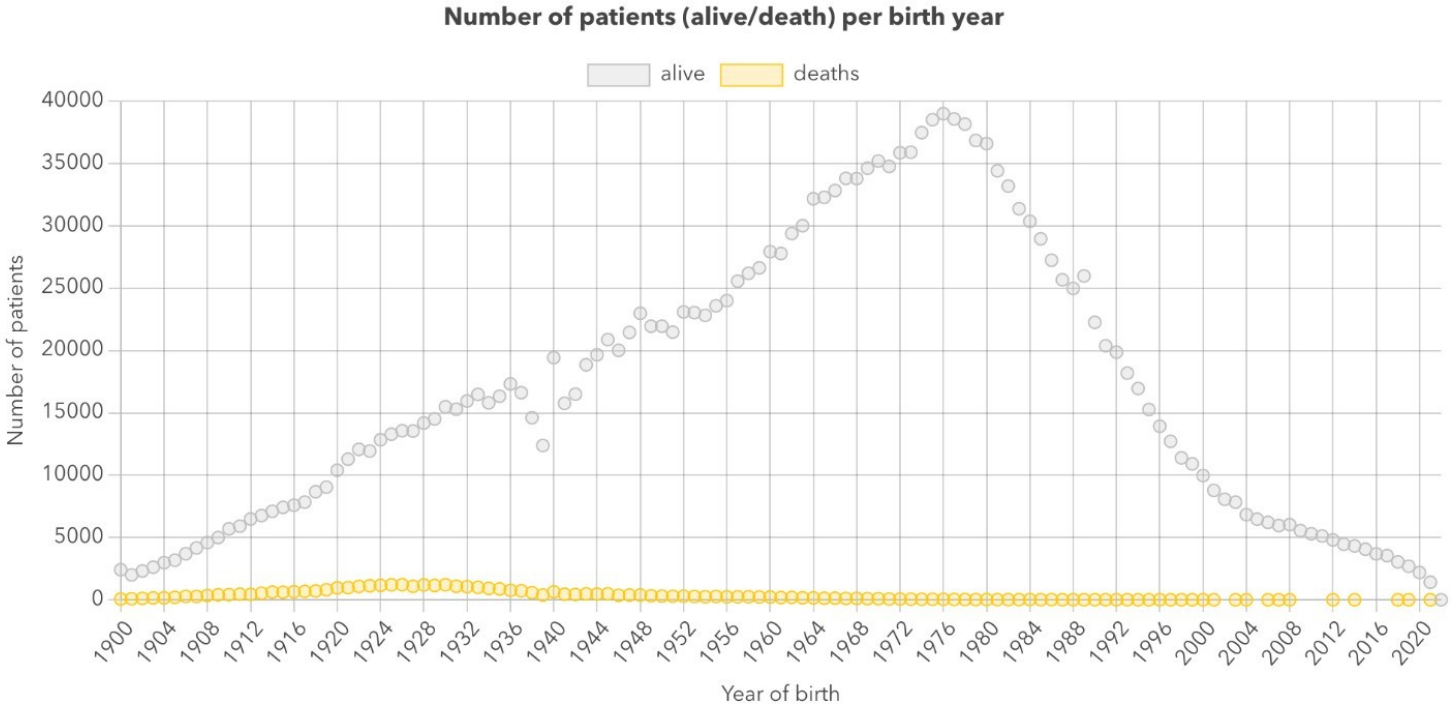
Untitled

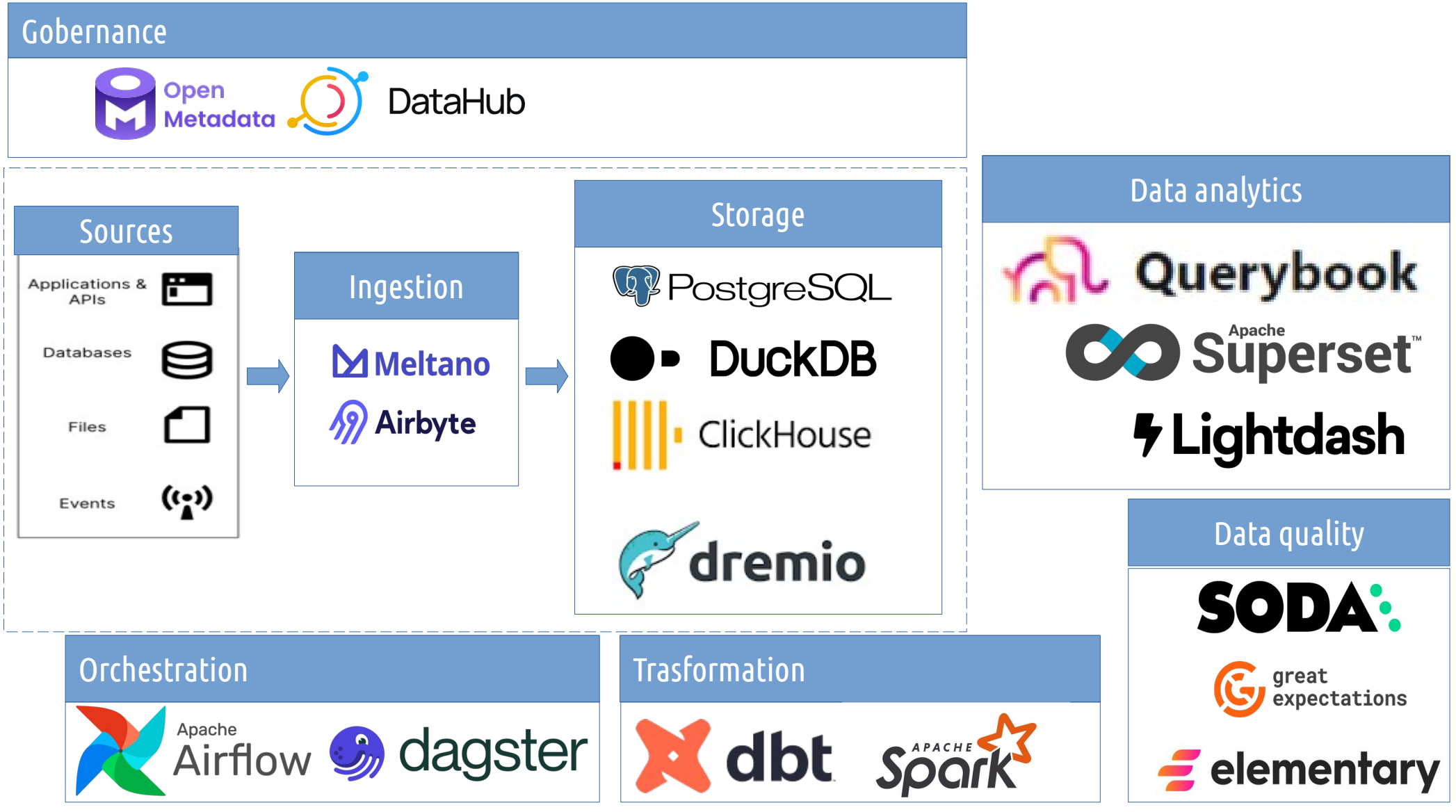
Adhoc Docs Tables Snippets Runs Search

EXECUTION 90

STATEMENT 1 OUT OF 1

CONFIG CHART



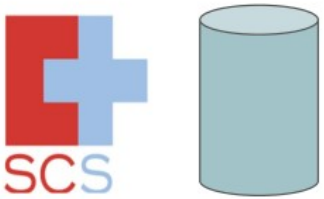

Gobernance
Open Metadata
DataHub
Sources
Applications & APIs
Databases
Files
Events
Ingestion
Meltano
Airbyte
Storage
PostgreSQL
DuckDB
ClickHouse
dremio
Data analytics
Querybook
Apache Superset
Lightdash
Data quality
SODA
great expectations
elementary
Orchestration
Apache Airflow
dagster
Trasformation
dbt
APACHE Spark

(Clinical) Data is complicated

Vall d'Hebron
Hospital

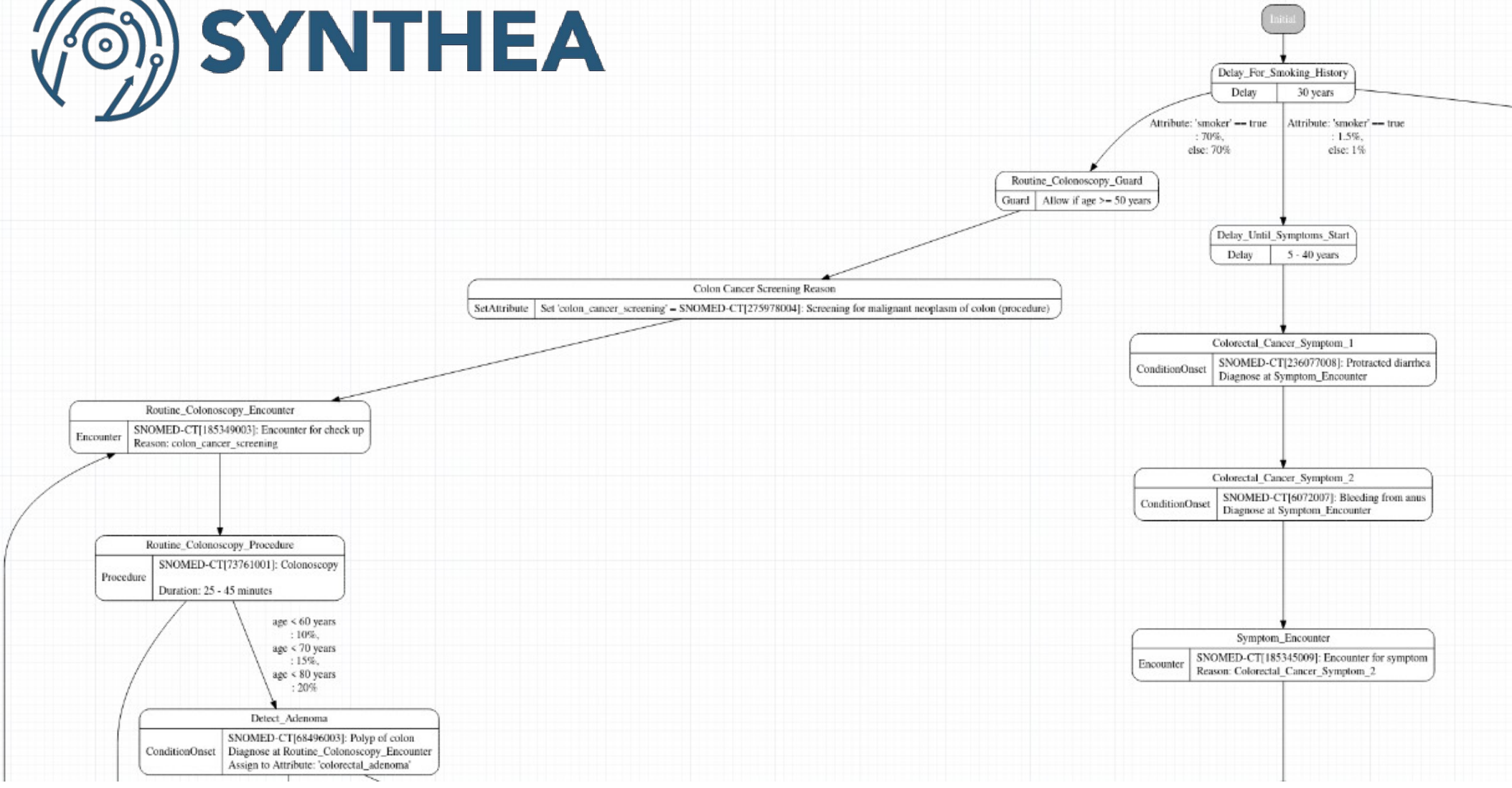
Hospital
del Mar

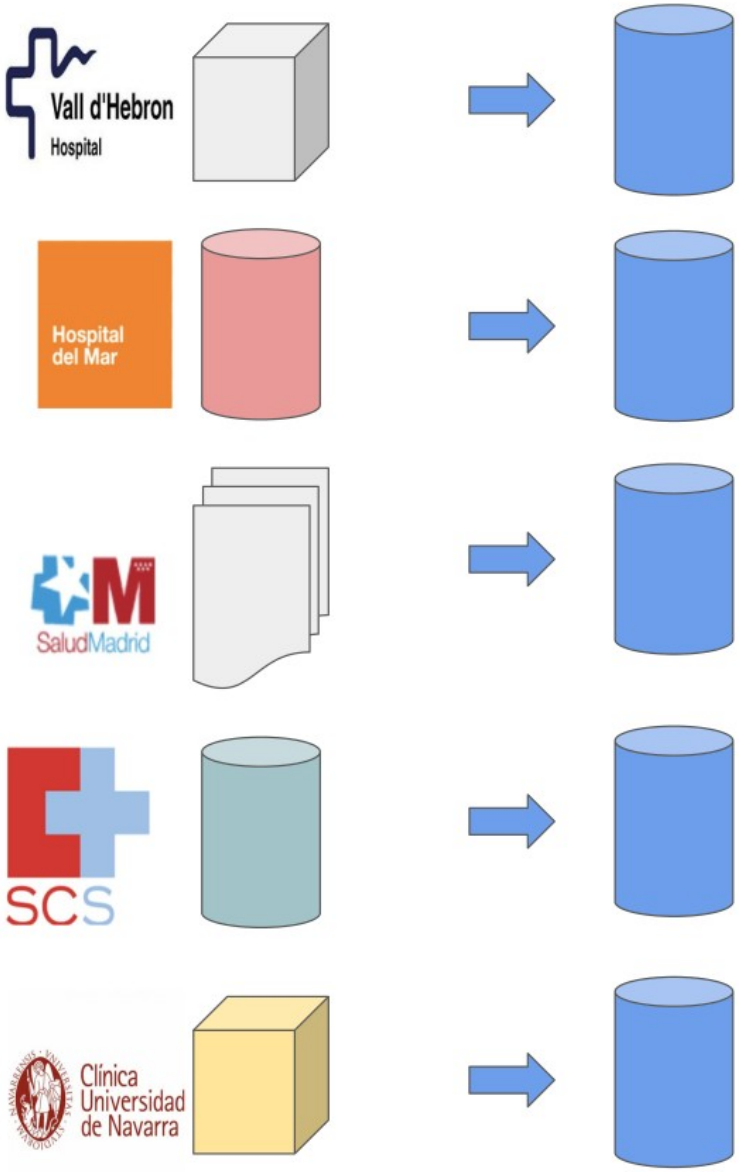
SaludMadrid


SCS

Clínica
Universidad
de Navarra

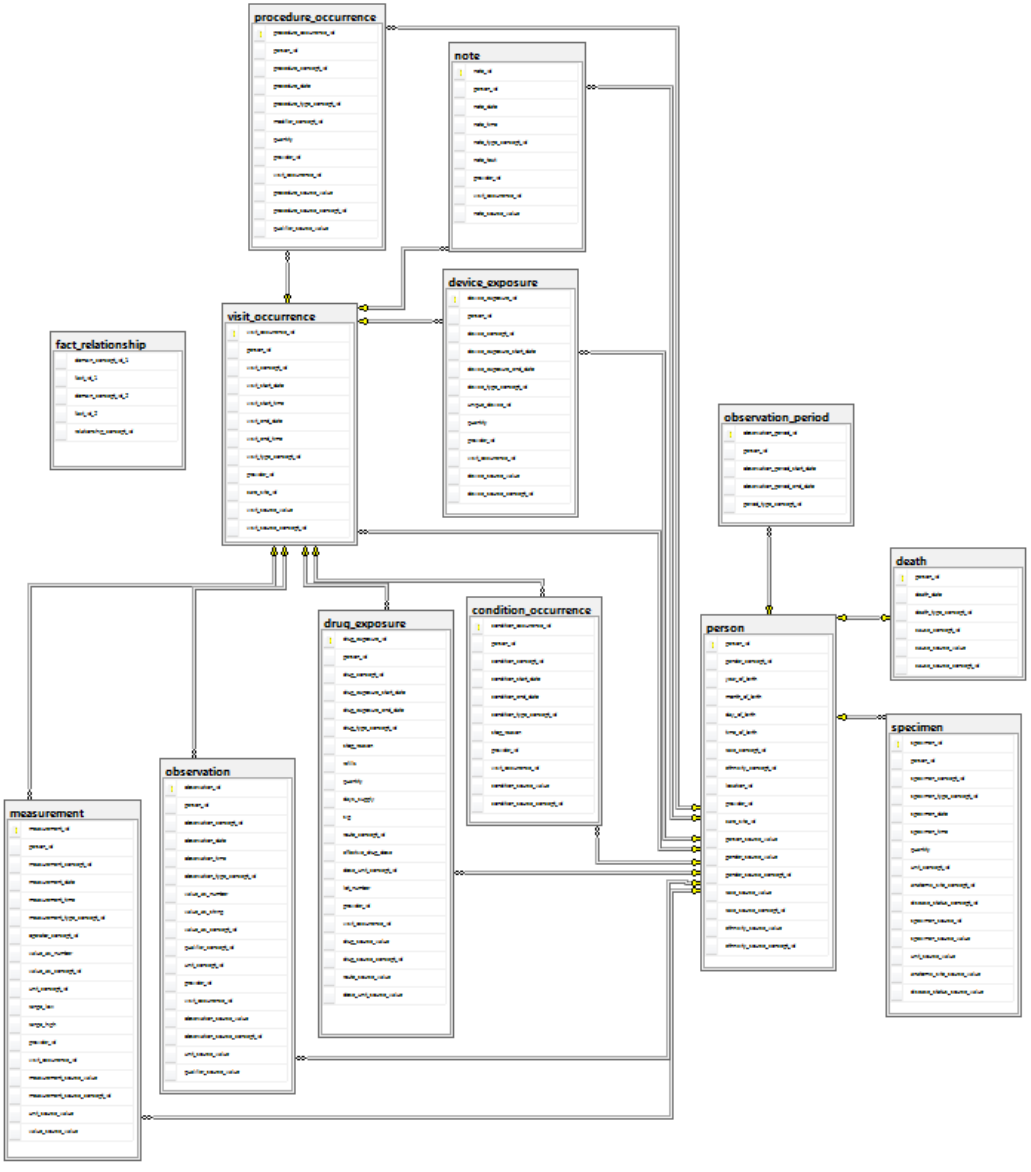
SYNTHEA


Initial
Delay_For_Smoking_History
Delay
30 years
Attribute: 'smoker' == true
: 70%,
else: 70%
Attribute: 'smoker' == true
: 1.5%,
else: 1%
Routine_Colonoscopy_Guard
Guard
Allow if age >= 50 years
Delay_Until_Symptoms_Start
Delay
5 - 40 years
Colon Cancer Screening Reason
SetAttribute
Set 'colon_cancer_screening' = SNOMED-CT[275978004]: Screening for malignant neoplasm of colon (procedure)
Colorectal_Cancer_Symptom_1
ConditionOnset
SNOMED-CT[236077008]: Protracted diarrhea
Diagnose at Symptom_Encounter
Routine_Colonoscopy_Encounter
Encounter
SNOMED-CT[185349003]: Encounter for check up
Reason: colon_cancer_screening
Colorectal_Cancer_Symptom_2
ConditionOnset
SNOMED-CT[6072007]: Bleeding from anus
Diagnose at Symptom_Encounter
Routine_Colonoscopy_Procedure
Procedure
SNOMED-CT[73761001]: Colonoscopy
Duration: 25 - 45 minutes
age < 60 years
: 10%,
age < 70 years
: 15%,
age < 80 years
: 20%
Symptom_Encounter
Encounter
SNOMED-CT[185345009]: Encounter for symptom
Reason: Colorectal_Cancer_Symptom_2
Detect_Adenoma
ConditionOnset
SNOMED-CT[68496003]: Polyp of colon
Diagnose at Routine_Colonoscopy_Encounter
Assign to Attribute: 'colorectal_adenoma'


Vall d'Hebron
Hospital
Hospital
del Mar
SaludMadrid
SCS
Clínica
Universidad
de Navarra

OHDSI

OBSERVATIONAL HEALTH DATA SCIENCES AND INFORMATICS



https://ohdsi.github.io/CommonDataModel/

ATHENA

SEARCH | DOWNLOAD | Alberto Labarga

SEARCH BY KEYWORD

cancer

cancer ×

Standard ×

DOWNLOAD RESULTS

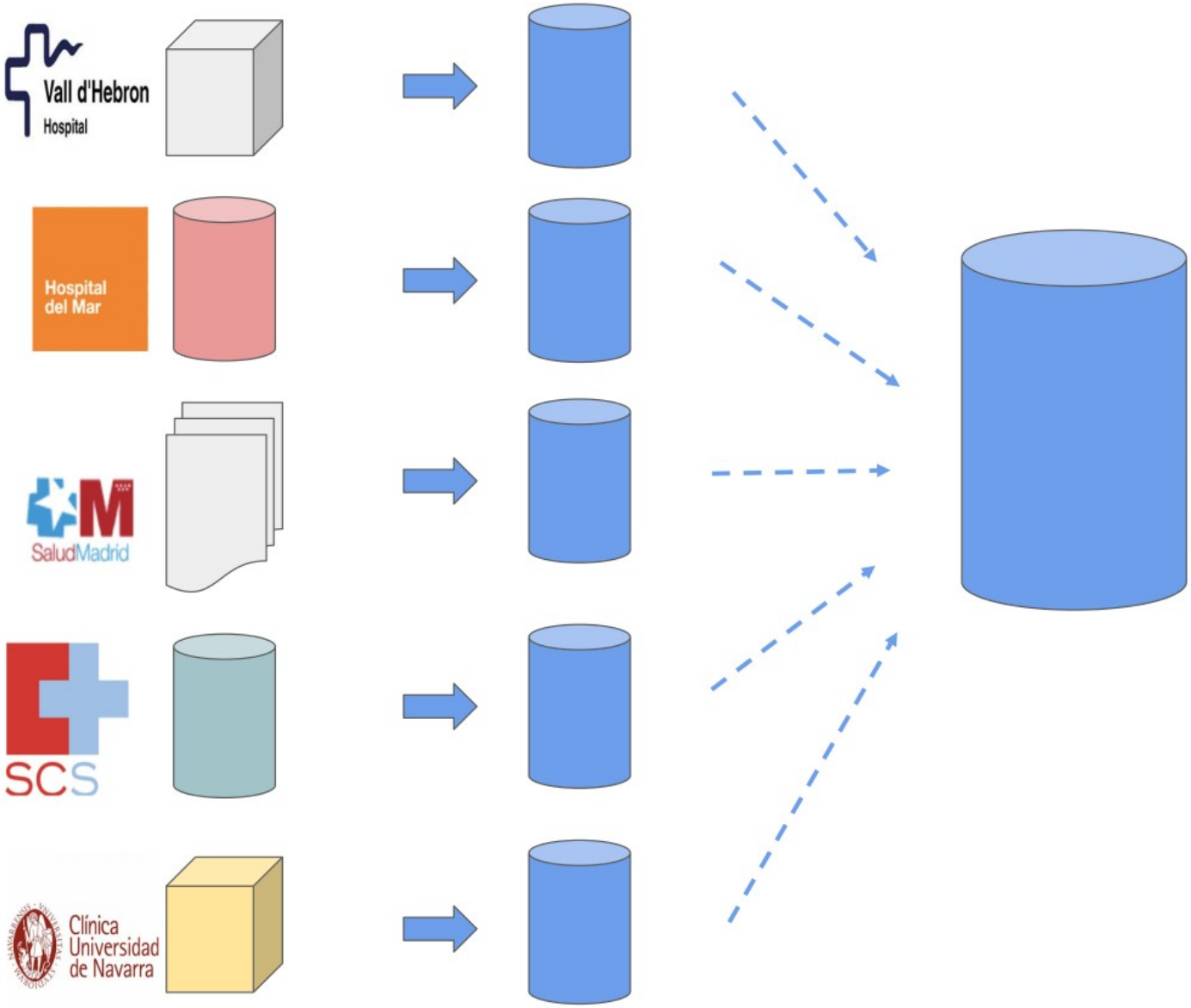
Show by 15 items Total 8,007 items

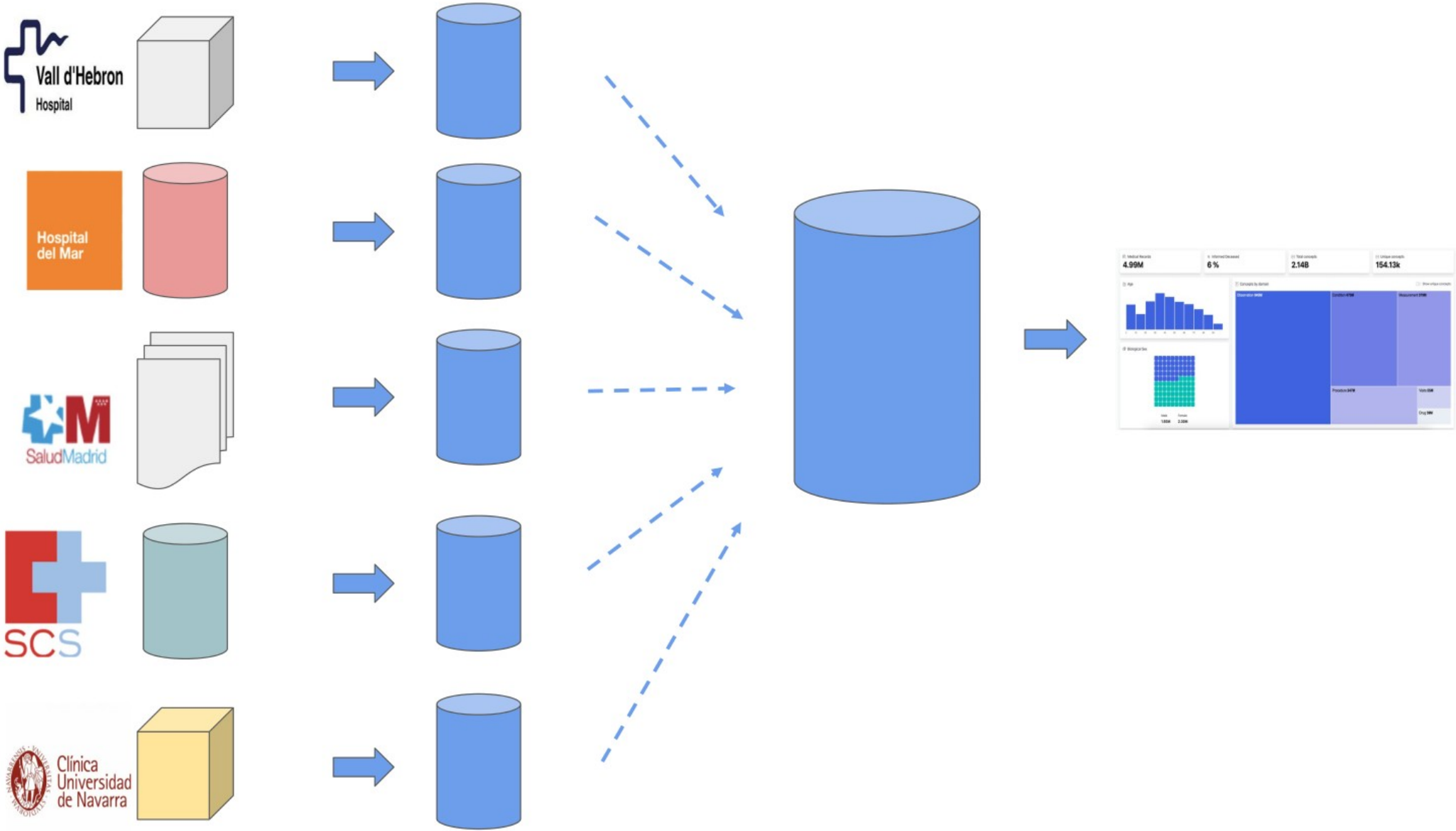
1 2 3 4 5 ... 534

- DOMAIN
- CONCEPT
  - Classification (346)
  - Non-standard (5079)
  - Standard (8007)
- CLASS
- VOCAB
- VALIDITY

ID	CODE	NAME	CLASS	CONCEPT	VALIDITY	DOMAIN	VOCAB
45877275	LA10524-9	Cancer	Answer	Standard	Valid	Meas Value	LOINC
4194405	395099008	Cancer confirmed	Context-dependent	Standard	Valid	Condition	SNOMED
4299598	386604001	Cancer registrar	Social Context	Standard	Valid	Observation	SNOMED
1384523	CancerConditions_NoCancer	Cancer Conditions: No Cancer	Answer	Standard	Valid	Observation	PPI
4073624	243072006	Cancer education	Procedure	Standard	Valid	Procedure	SNOMED
44803809	722731000000100	Colorectal cancer	Qualifier Value	Standard	Valid	Observation	SNOMED
40757663	54532-7	Cancer	Survey	Standard	Valid	Observation	LOINC

https://athena.ohdsi.org


Vall d'Hebron
Hospital
Hospital
del Mar
SaludMadrid
SCS
Clínica
Universidad
de Navarra


Vall d'Hebron
Hospital
Hospital
del Mar
SaludMadrid
SCS
Clínica
Universidad
de Navarra
4.99M
6 %
2.14B
154.13k

bsc-health-data / **pydatalondon23-modern-data-stack** Public

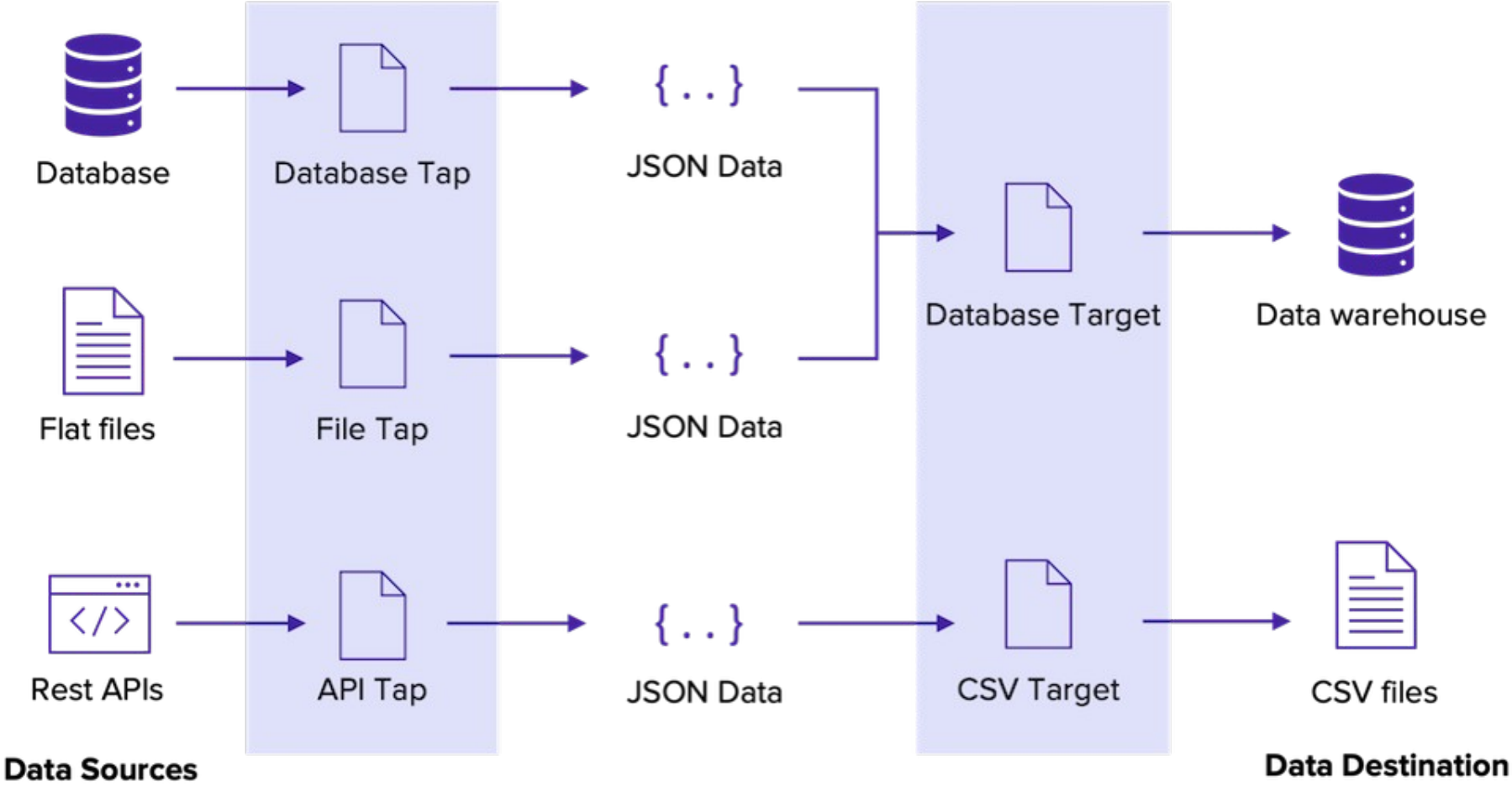
alabarga Update README.md `cd3d156` 5 minutes ago **50** commits

database	Add database DOckerfile	7 months ago
datahub	Add datahub	7 months ago
dremio	Add files	7 months ago
meltano	Create meltano.yml	2 days ago
obevo	Add obevo schemas	7 months ago
slides	Add files via upload	5 months ago
synthea	Update README.md	2 days ago
.gitattributes	Add obevo	7 months ago
.gitignore	Initial commit	7 months ago
LICENSE	Initial commit	7 months ago
README.md	Update README.md	5 minutes ago
TUTORIAL.md	Update TUTORIAL.md	2 days ago

https://github.com/bsc-health-data/pydatalondon23-modern-data-stack

What about Python?

SINGER


Database
Database Tap
JSON Data
Flat files
File Tap
JSON Data
Rest APIs
API Tap
JSON Data
Database Target
Data warehouse
CSV Target
CSV files
Data Sources
Data Destination

```python
class TapCSV(Tap):
    """CSV tap class."""

    name = "tap-csv"

    config_jsonschema = th.PropertiesList(
        th.Property(
            "csv_file",
            th.StringType,
            description="A path to the CSV file.",
        )).to_dict()

    def discover_streams(self) -> List[Stream]:
        """Return a list of discovered streams."""
        return [
            CSVStream(
                tap=self,
                name=file_config.get("entity"),
                file_config=file_config,
            )
            for file_config in self.config.get("csv_file")
        ]
```

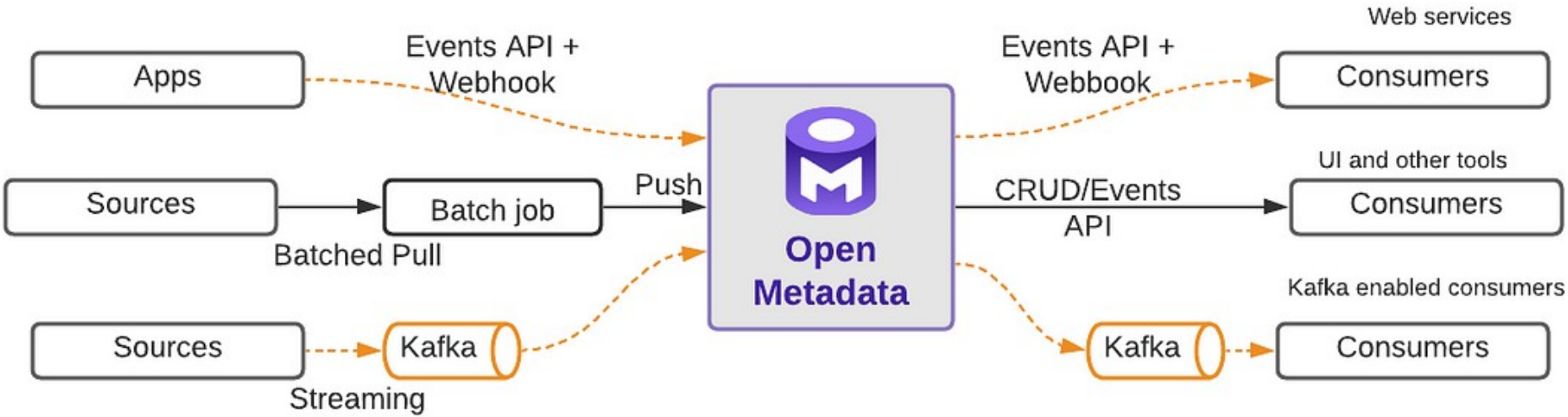
```python
class CSVStream(Stream):
    """Stream class for CSV streams."""
    def get_records(self, context: Optional[dict]) -> Iterable[dict]:
        """Return a generator of row-type dictionary objects.
        """
        for file_path in self.get_file_paths():

            for row in self.get_rows(file_path):

                yield dict(zip(self.header, row))
```

https://gitlab.com/meltano/sdk/-/tree/main/cookiecutter

Snowpark
Databricks cluster
Dataproc
Data Warehouse

```python
def model(dbt, session):

    # DataFrame representing an upstream model
    upstream_model = dbt.ref("upstream_model_name")

    # DataFrame representing an upstream source
    upstream_source = dbt.source("upstream_source_name", "table_name")

    final_df = ...  # stuff you can't write in SQL!

    return final_df

}
```

https://docs.getdbt.com/docs/building-a-dbt-project/building-models/python-models



Metadata Ingestion - Pull/Push/Streaming

Metadata Consumption - Pull/Push/Streaming

```python
metadata = OpenMetadata(server_config)
create_ml_model = CreateMlModelRequest(
    name=raw_meta["name"],
    description=raw_meta["description"],
    algorithm=raw_meta["algorithm"],
    target=raw_meta["target"],
    mlFeatures=[ ...],
    mlHyperParameters=[...],
    mlStore=MlStore(
        storage=raw_meta["mlStore"]["storage"],
        imageRepository=raw_meta["mlStore"]["imageRepository"]
    ),
    service=get_or_create_ml_model_service(metadata, raw_meta["serviceName"])
)
mlmodel: MlModel = metadata.create_or_update(create_ml_model)
```

https://github.com/open-metadata/openmetadata-demo/blob/main/mlmodel-cicd/mlmodel_cicd.py

```python
class MyAwesomeConnector(Source):
    """
    Custom connector to ingest Database metadata
    """

    def __init__(self, config: WorkflowSource, metadata_config: OpenMetadataConnection):
        self.config = config
        self.metadata_config = metadata_config

        self.metadata = OpenMetadata(self.metadata_config)
        self.status = MyAwesomeConnectorStatus()

    @classmethod
    def create(
        cls, config_dict: dict, metadata_config: OpenMetadataConnection
    ) -> "MyAwesomeConnector":
        config: WorkflowSource = WorkflowSource.parse_obj(config_dict)
        return cls(config, metadata_config)


    def next_record(self) -> Iterable[Entity]:
        pass
```

https://github.com/open-metadata/openmetadata-demo/tree/main/custom-connector/

PyData

FRIDAY

1:30 PM (1h30min)
Building an End-to-End Open-Source Modern Data Platform for Biomedical Data
Alberto Labarga

3:30 PM (1h30min)
Delta Lake 101: How many water metaphors does it take to describe data?
Holly Smith, Eoin O'Flanagan

SATURDAY

3:00 PM (40min)
Serverless Python Analytics at Petabyte scale using ArcticDB
William Dealtry

3:45 PM (40min)
Mastering Great Expectations: Ensuring Data Quality in Your Data Pipelines.
Carsten Frommhold

4:30 PM (40min)
Unleashing the Power of dbt and Python for Modern Data Stack
Meder Kamalov

Thanks!