ONLINE MASTERS IN DATA SCIENCE

DSC 208R - Data Management for Analytics

Data Collection and Governance

Arun Kumar

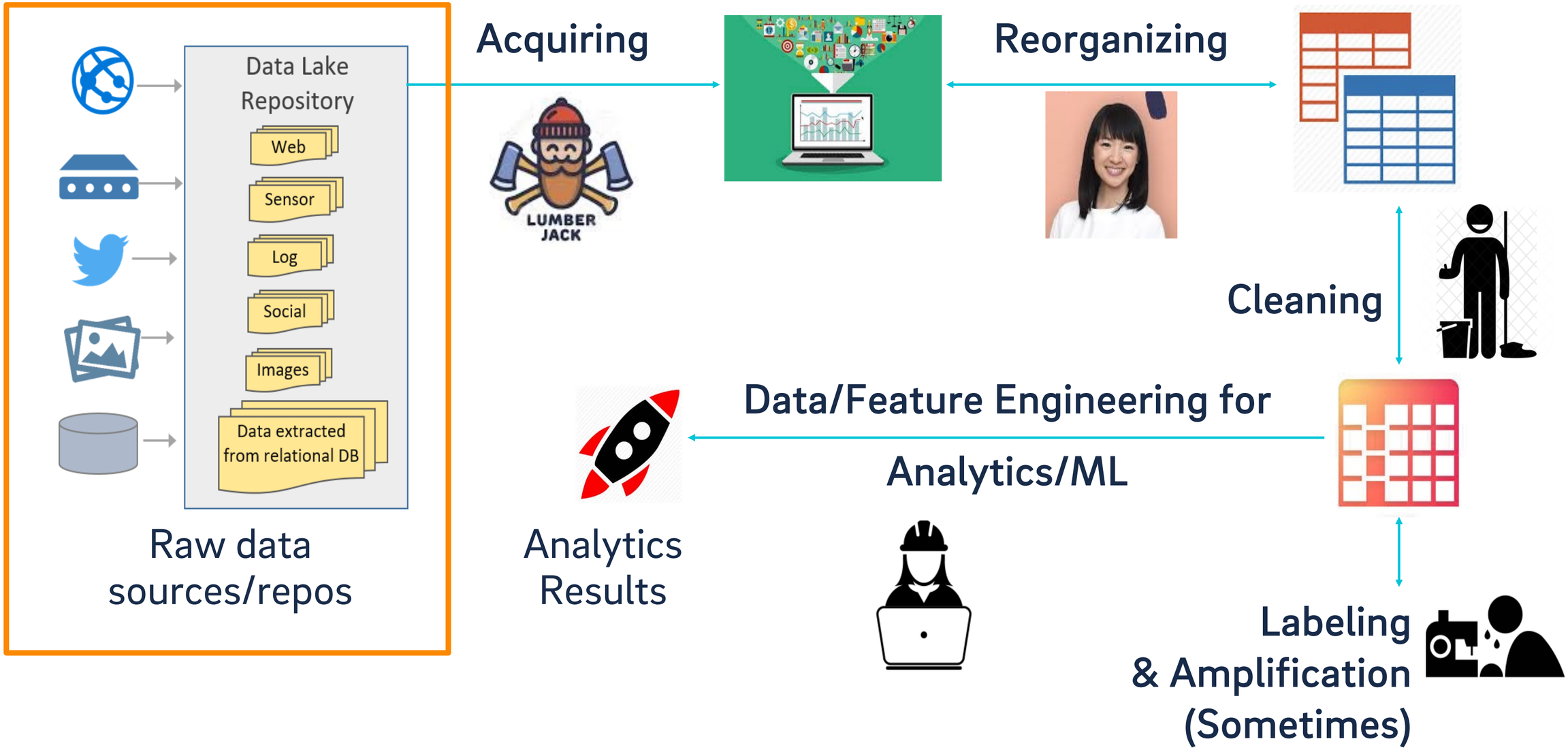
UC San Diego

COMPUTER SCIENCE & ENGINEERING
HALICIOĞLU DATA SCIENCE INSTITUTE

Outline

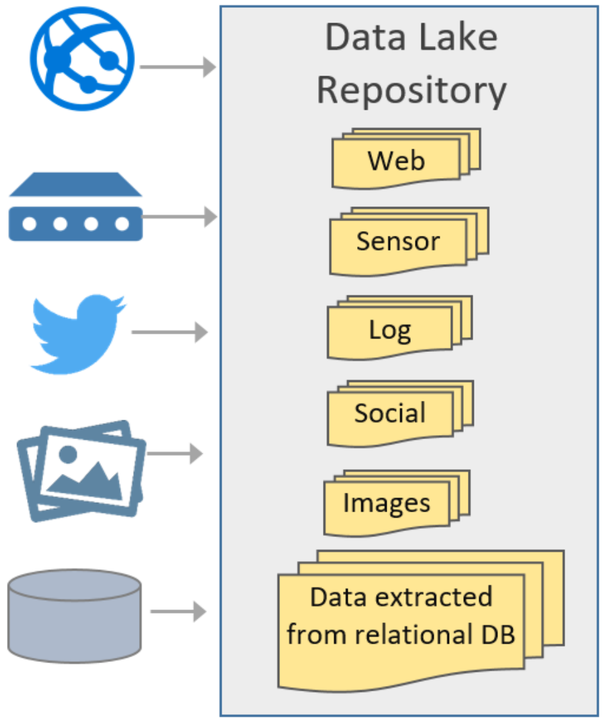
- Overview
- **Data Organization and File Formats**
- Data Acquisition
- Data Reorganization and Preparation
- Data Labeling and Amplification
- Data Governance and Privacy

Acquiring Data



Data Organization

- Modern data-intensive applications typically have multitudes of data modalities:



Raw data
sources/repos

- Structured data (e.g., relational data)
- Sequence data (e.q., IoT time series)
- Semistructured data (e.g., JSON logs)
- Graph-structured data (e.g., social media)
- Text files, documents (e.g., reviews)
- Multimedia data (images, audio, video, etc.,)
- Multimodal files (e.g., PDFs, notebooks)

Abstraction: File

- **File:** A persistent sequence of bytes that stores a logically coherent digital object for an application
    - **File Format:** An application-specific standard that dictates how to interpret and process a file's bytes
    - 100s of file formats exist (e.g., TXT, DOC, GIF, MPEG); varying data models/types, domain-specific, etc.
    - **Metadata:** Summary or organizing info about file content (aka *payload)* stored with file itself; format dependent
- **Directory**: A cataloging structure with a list of references to files and/or (recursively) other directories
    - Treated as a special kind of file

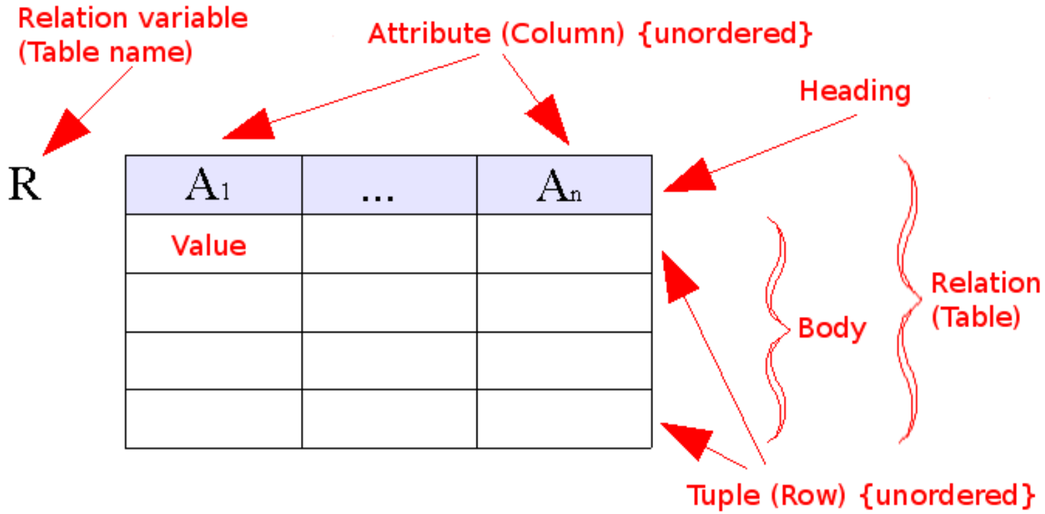
Files vs. Databases: Data Model

- **Database:** An *organized* collection of interrelated data
  - **Data Model**: An abstract model to define organization of data in a formal (mathematically precise) way
  - E.g., Relations, XMLS, Matrices, DataFrames
- Every database is just an *abstraction* on top of data files!
  - **Logical level**: Data model for higher-level reasoning
  - **Physical level**: How bytes are layered on top of files
  - All data systems (RDBMSs, Spark, TensorFlow, etc.) are software systems that manipulate data files under the hood

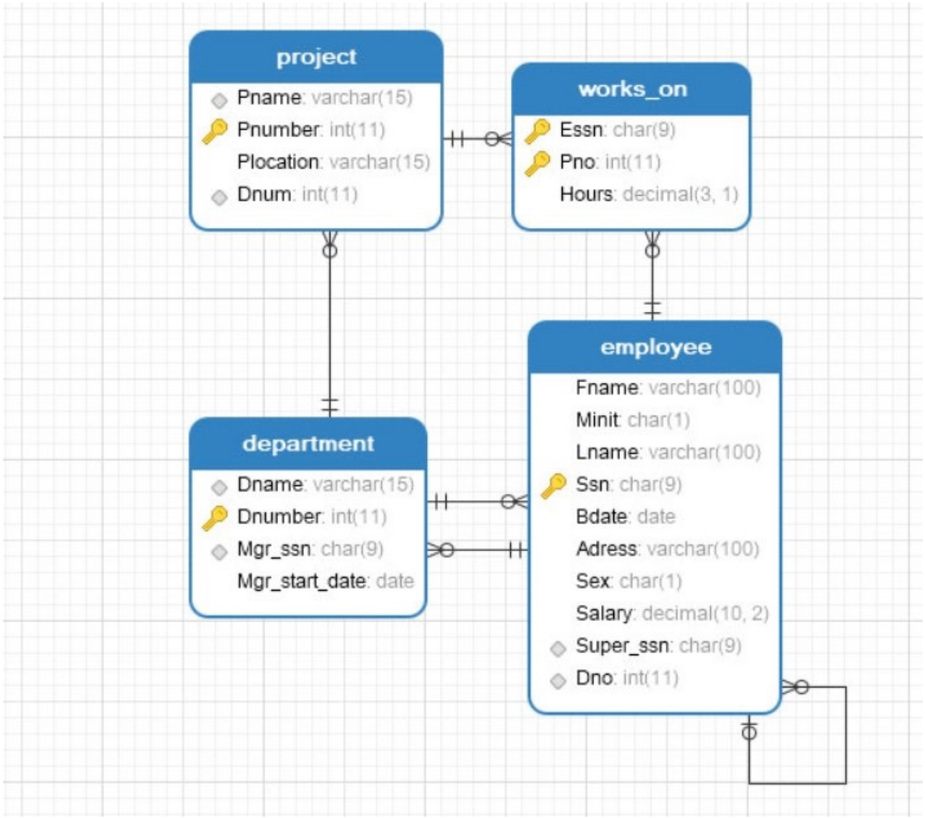
Data as File: Structured

- **Structured Data**: A form of data with regular structure

Relation



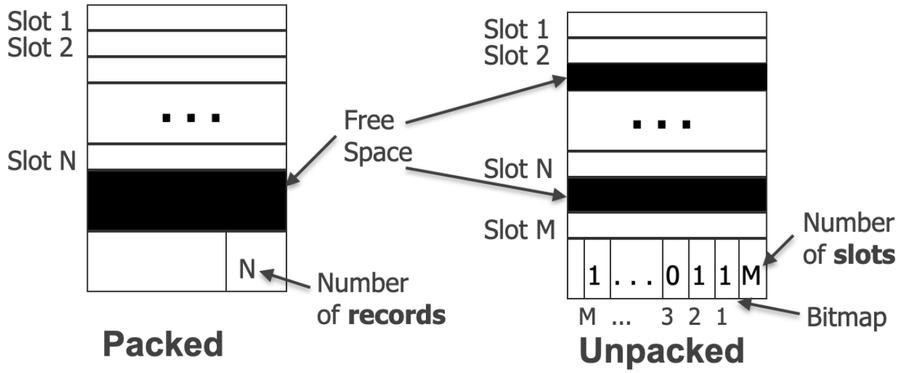
Relational Database



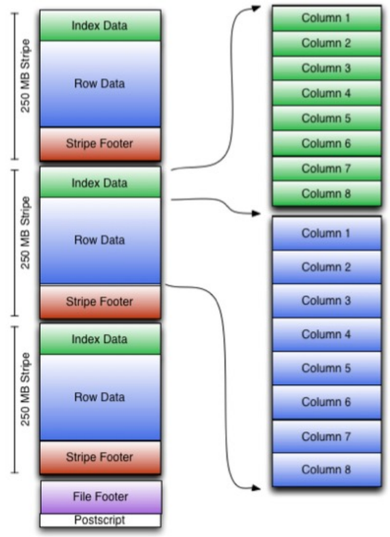
- Most RDBMSs and Spark serialize a relation as *binary* file(s), often compressed

Aside: Relational File Formats

- Different RDBMSs and Spark/HDFS-based tools serialize relation/tabular data in different binary formats, often compressed
  - One file per relation; *data layout* can be row vs. columnar (e.g., ORC, Parquet) vs. hybrid formats
  - RDBMS vendor-specific vs open Apache
  - Parquet becoming especially popular



Row-oriented



Columnar

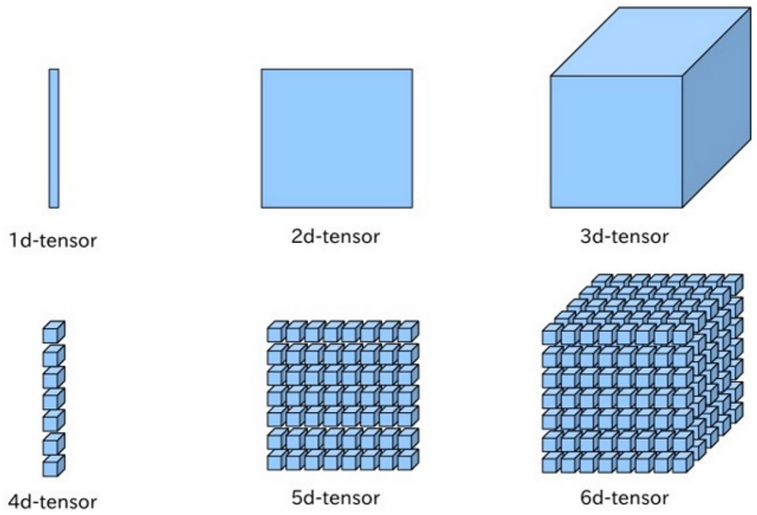
Data as File: Structured

- **Structured Data**: A form of data with regular substructure

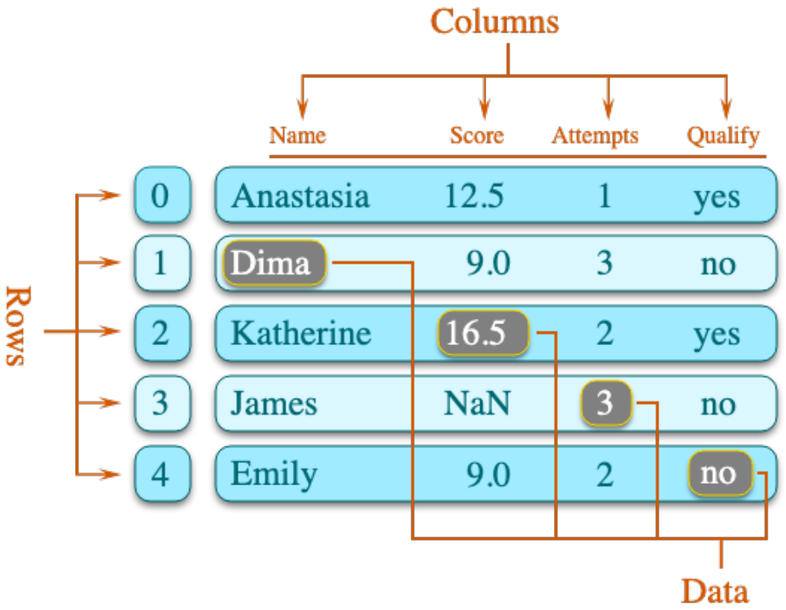
Matrix

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ \vdots \\ m \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & \dots & n \end{array} \\ \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ a_{31} & a_{32} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} \end{array}$$

Tensor



DataFrame



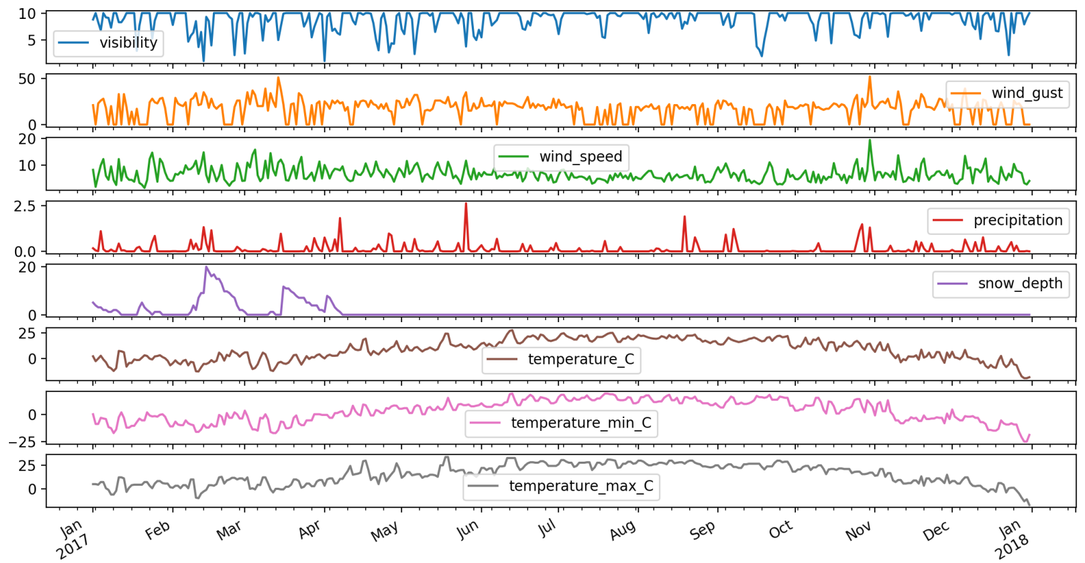
- Typically serialized as restricted ASCII text file (TSV, CSV, etc.)
- Matrix/tensor as binary too
- Can layer on Relations too!

Data as File: Structured

- **Structured Data**: A form of data with regular substructure

Sequence
(Includes Time-series)

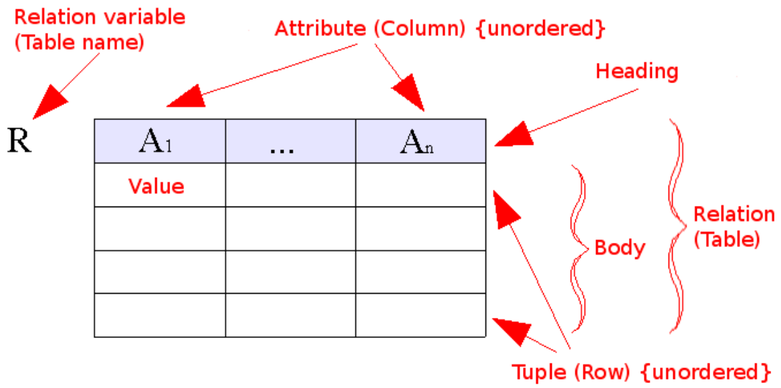
GAT AAAT CT GG TCT T ATT TCC

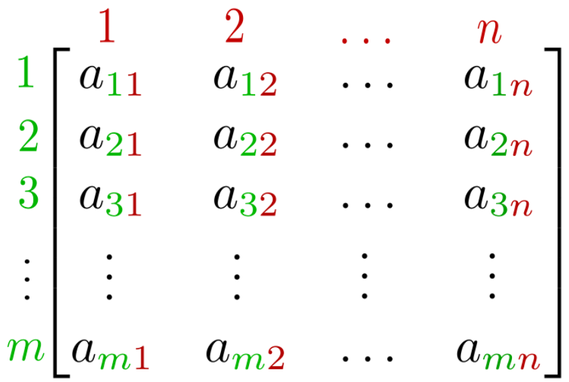


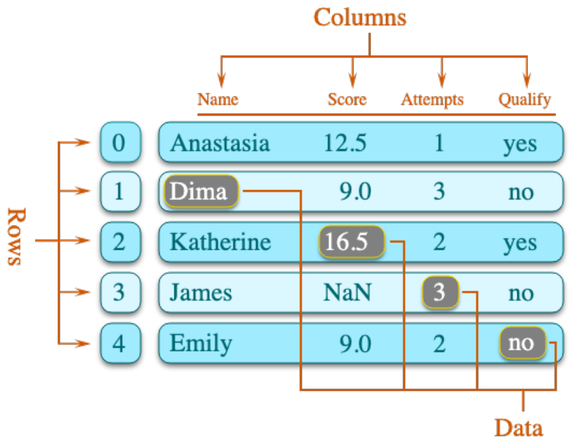
- Can layer on Relations, Matrices, or DataFrames, or be treated as first-class data model
- Inherits flexibility in file formats (text, binary, etc.)

Comparing Struct. Data Models

Q: What is the difference between Relation, Matric, and DataFrame?







- **Ordering:** Matrix and DataFrame have row/col numbers; Relation is orderless on both axes
- **Schema Flexibility:** Matrix cells are numbers. Relation tuples conform to pre-defined schema. DataFrame has no pre-defined schema but all rows/cols can have names; col cells can be mixed types
- **Transpose:** Supoorted by Matrix & DataFrame, not Relation

If interested in reading more:
https://towardsdatascience.com/preventing-the-death-of-the-dataframe-8bca1c0f83c8

Data as File: Semistructured

- **Semistructured Data**: A form of data with less regular/more flexible substructure than structured data

Tree-Structured

```xml
<?xml version="1.0" encoding="UTF-8"?>
<customers>
    <customer>
        <customer_id>1</customer_id>
        <first_name>John</first_name>
        <last_name>Doe</last_name>
        <email>john.doe@example.com</email>
    </customer>
    <customer>
        <customer_id>2</customer_id>
        <first_name>Sam</first_name>
        <last_name>Smith</last_name>
        <email>sam.smith@example.com</email>
    </customer>
    <customer>
        <customer_id>3</customer_id>
        <first_name>Jane</first_name>
        <last_name>Doe</last_name>
        <email>jane.doe@example.com</email>
    </customer>
</customers>
```

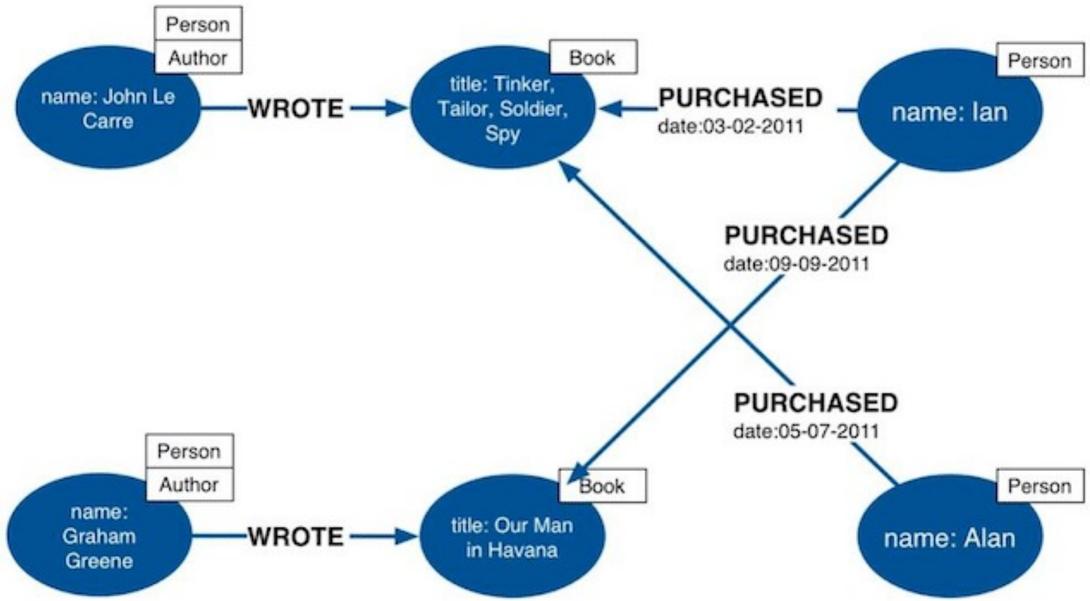
```
[
    {
        orderId: 1,
        date: '1/1/2014',
        orderItems: [
                {itemId: 1, qty: 3, price: 23.4},
                {itemId: 23, qty: 2, price: 3.3},
                {itemId: 7, qty: 5, price: 5.3}
            ]
    },
    {
        orderId: 2,
        date: '1/2/2014',
        orderItems: [
            {itemId: 31, qty: 7, price: 3.8},
            {itemId: 17, qty: 4, price: 9.2}
        ]
    },
    {
        orderId: 3,
        date: '1/5/2014',
        orderItems: [
            {itemId: 11, qty: 9, price: 13.3},
            {itemId: 27, qty: 2, price: 19.2},
            {itemId: 6, qty: 19, price: 3.6},
            {itemId: 7, qty: 22, price: 9.1}
        ]
    }
]
```

- Typically serialized as restricted ASCII text file with formatting as JSON, YML, XML, etc.
- Some data systems also offer binary file formats
- Can layer on Relations too

Data as File: Semistructured

- **Semistructured Data:** A form of data with less regular / more flexible substructure than structured data

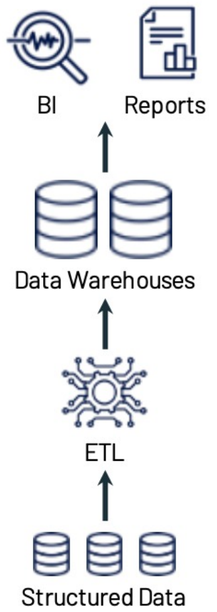
Graph-Structured



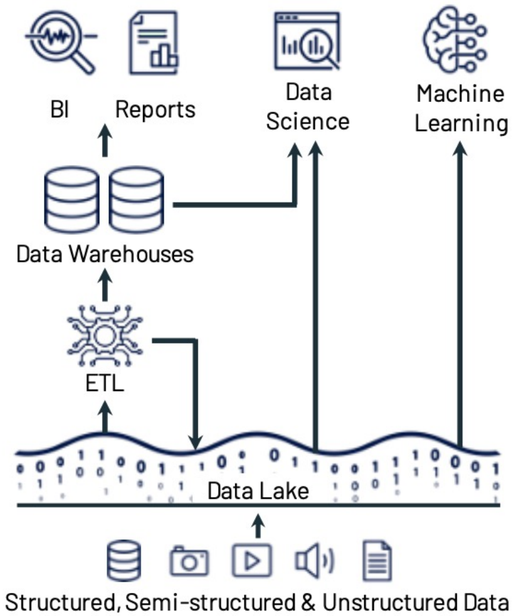
- Typically serialized with JSON or similar textual formats
- Some data systems also offer binary file formats
- Again, can layer on Relations too

Data File on Data "Lakes"

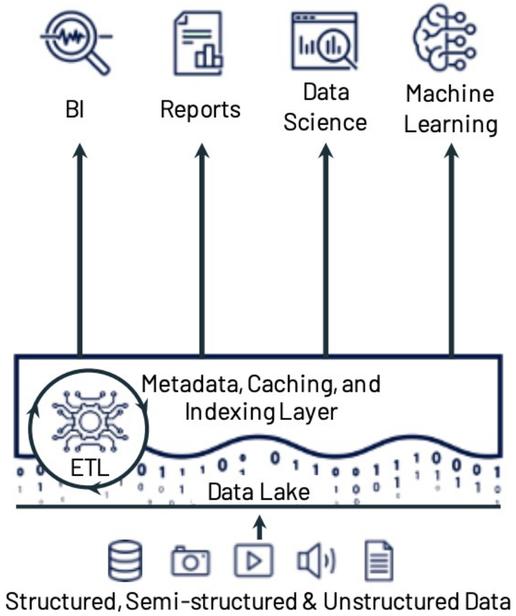
- **Data "Lake":** *Loose coupling* of data file format for storage and data/query processing stack (vs. RDBM's tight coupling)
  - JSON for raw data; Parquet processed is common



(a) First-generation platforms.



(b) Current two-tier architectures.



(c) Lakehouse platforms.

If interested, check out this vision paper on the future of data lakes:

http://cidrdb.org/cidr2021/papers/cidr2021_paper17.pdf

Data Lake File Format Tradeoffs

Pros and Cons of Parquet vs. Text-Based files (CSV, JSON, etc.):

- **Less storage:** Parquet stores in **compressed** form; can be much smaller (even 10x); lowers read latency
- **Column pruning:** Enables app to read only columns needed to DRAM; even lower query latency
- **Schema on file:** Rich metadata, stats inside format itself
- **Complex types:** Can store them in a column
- **Human-readability:** Cannot open with text apps directly
- **Mutability:** Parquet is immutable/read-only; no in-place edits
- **Decompression/Deserialization overhead:** Depends on application tool; can go either way
- **Adoption in practice:** CSV/JSON support more pervasive but Parquet is catching up

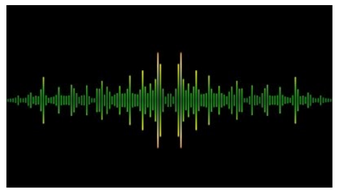
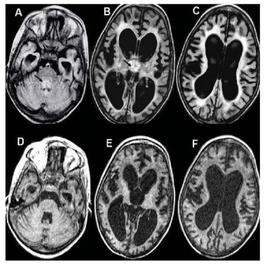
Data Lake File Format Tradeoffs

Dataset	Size on Amazon S3	Query Run Time	Data Scanned	Cost
Data stored as CSV files	1 TB	236 seconds	1.15 TB	$5.75
Data stored in Apache Parquet Format	130 GB	6.78 seconds	2.51 GB	$0.01
Savings	87% less when using Parquet	34x faster	99% less data scanned	99.7% savings

https://databricks.com/glossary/what-is-parquet

Data as File: Other Common Formats

- **Text File** (aka plaintext): Human-readable ASCII characters
- **Multimedia** files encode tensors and/or time-series of signals
  - Myriad binary formats, typically with (lossy) compression e.g., WAV for audio, MP4 for video, etc.


- **Docs/Multimodal File:** Myriad app-specific rich binary formats