CSCI E-103

Data Engineering for Analytics to Solve Business Challenges

Data Modeling

Lecture 02

Anindita Mahapatra & Eric Gieseke

Harvard Extension, Fall 2025

Agenda

- Review previous lecture
- Data Modeling Approaches
- Metadata - Data about Data!
- Data Formats and how to choose one
- Data Compression
- Data Profiling
- Best Practices

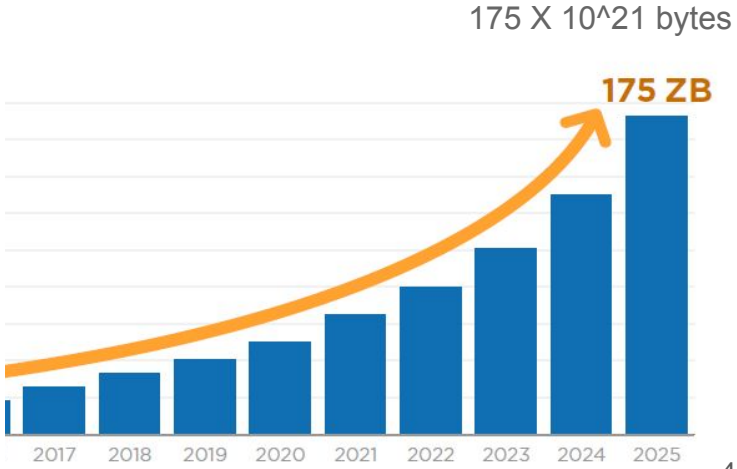
- Lab-01

ACID stands for	Atomicity Consistency Isolation Durability
BASE stands for	Basically Available, Soft State, Eventual Consistency
IaaS, PaaS & …	SaaS
Lakehouse combines the best characteristics of	Data Lake & Data Warehouse
DAG stands for	Directed Acyclic Graph
The biggest challenges with data is around	Data Quality & Data Staleness
The 3 main data types are	Structured, semi-structured & unstructured
ETL stands for	Extract Transform Load
Spark architecture consists of a driver &	1 or more worker nodes
Why is Spark considered a polyglot	Multi language support - Scala, Python, R, SQL, Java
Why is Spark faster than Hadoop	In mem Vs disk oper for every map/reduce operation
5 Vs of Big Data	Volume, Variety, Velocity, Veracity, Value

Example of Data Sources

According to Forbes, about 2.5 quintillion (10^18) bytes of data is generated every day.

[Big Data Statistics]



What is Data Modeling?

Organizing data so it fits the needs of business processes. (aka consumption) Some considerations are needed to support ingestion (especially to support SLAs)

It requires the design of logical relationships so the data can interrelate with each other and support the business.

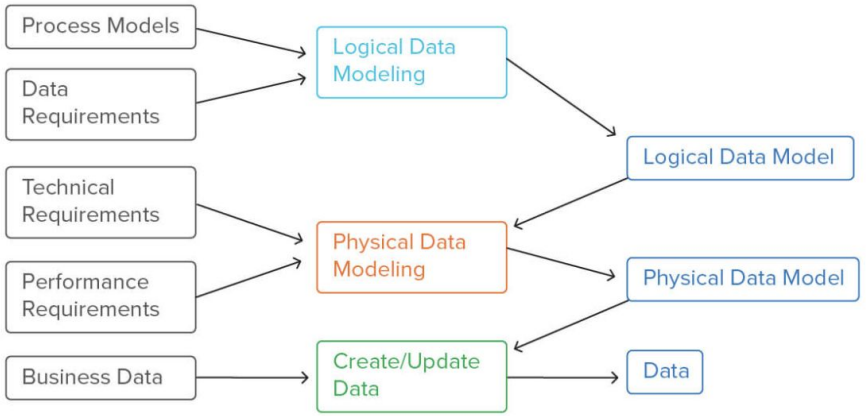
Document design as a conceptual model so that anyone can understand the rules, the schema, the ingestion and consumption.

Stages of Data Modeling		
Conceptual (Semantic) Data Model	**Logical Data Model**	**Physical Data Model**
Visual representation of entities and the relationships between them.These data models are created for key business stakeholders.	Further defines the structure of the data entities (attributes) and sets the relationships between them.	Used for database-specific modeling, goes into more detail with column types, constraints, partitions

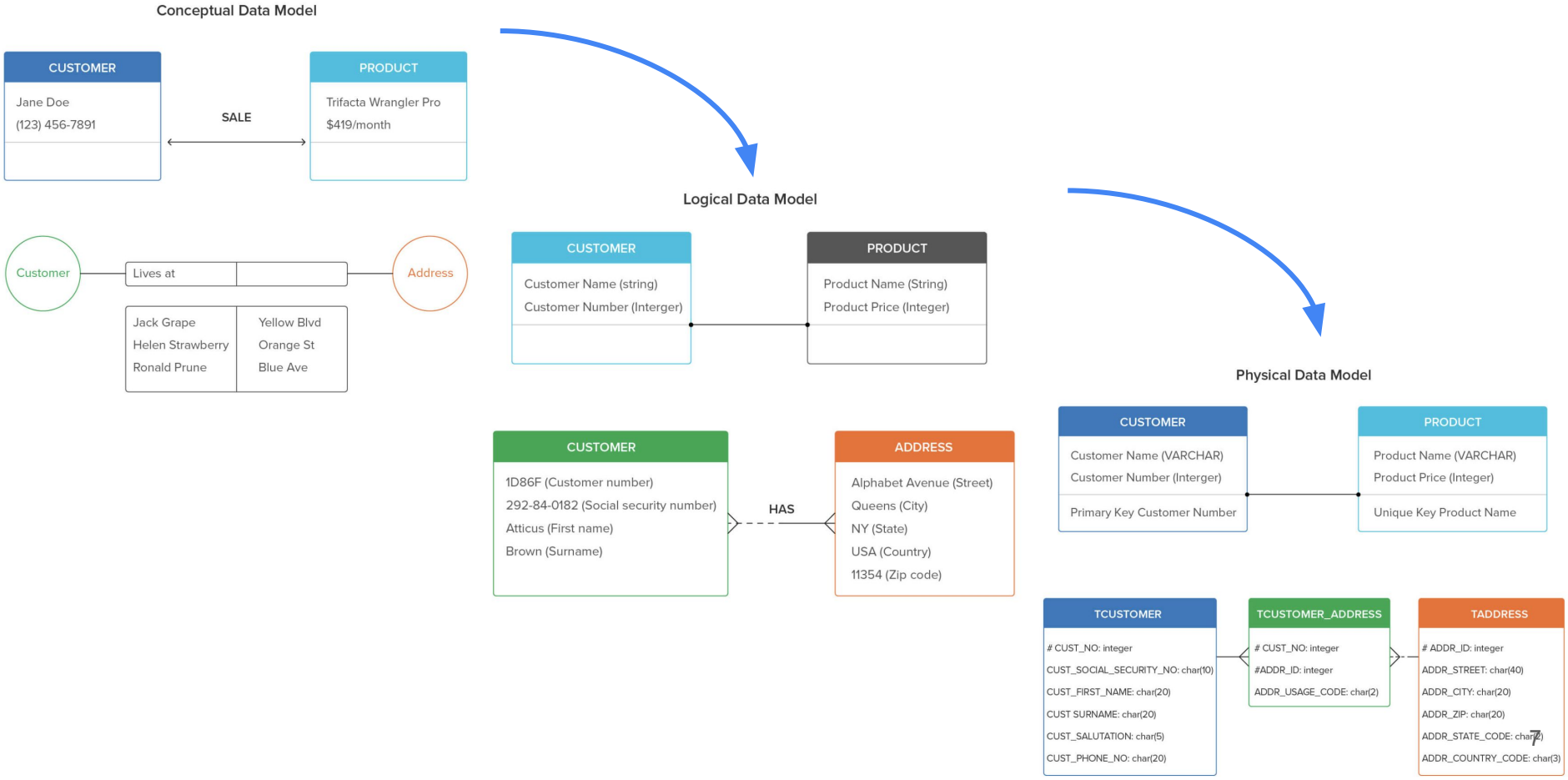
Why is Data Modeling important?

To get the most value out of data, one needs to understand its intrinsic properties and inherent relationships
Data Modeling is the process of visualizing and representing data
It is determined by both the business operational processes and the structure of the generated data

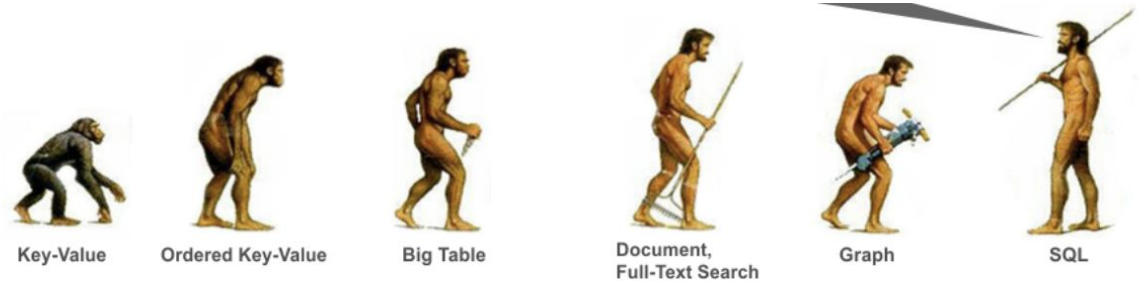
- Designed for persisting data and retrieving it in an optimal way
- The visual representation of relationships helps prevent incorrect representations
  - Data constraints enforce rules to improve data quality
  - Increases consistency in naming, rules, semantics, and security, while also improving data analytics.
- Provides common vocabulary for documentation
  - Business and all Tech counterparts
  - Metadata searches are faster
- Inconsistencies are discovered early on in the process which results in early and cheaper fixes



Modeling Example

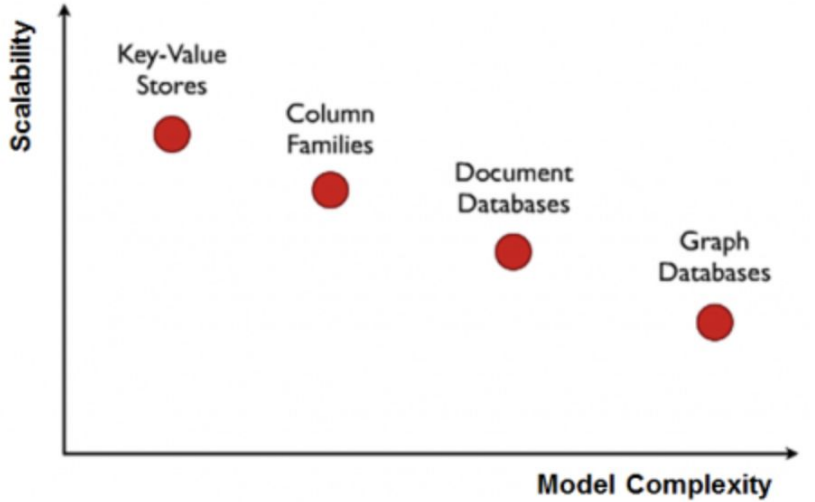


NoSQL Data modeling = Not Just SQL



Key-Value	BigTable	Document	Full Text Search	Graph
S3	Dynamo, HBase, Cassandra	Mongo, CouchDB	Lucene, Elastic, Solr	Neo4J

- Describe the workload
  - How will users query the data?
  - Identify patterns (bin)
  - How often is data added vs updated?



Handling nuances of Big Data/NoSQL modeling

- Since NoSQL is Schema free
  - Leads people to believe you don't need a model with NoSQL technologies
  - You do have to define the facets of how you plan to organize your data
- Contrast with relational modeling
  - Relational modeling is typically driven by the structure of available data.
    - The main design theme is "**What answers do I have?**"
  - NoSQL data modeling is typically driven by application-specific access patterns,
    - i.e. the types of queries to be supported.
    - The main design theme is **"What questions do I have?"**
- Data duplication and denormalization are first-class citizens.
  - Joins : Less prevalent
    - Joins are more expensive and sometimes not supported at all by the underlying data store
    - Wide Tables are a norm
    - Unlike RDBMS, max #columns is not a hard number
  - How is data linked? Embedded Vs Referenced
    - keep data that is frequently used together
    - Identify and model the relationships

Embedded

```
{
  "order_ID":"0001",

  "product":{
     "product_name":"Pencil",
     "product_category":"supplies",
     "list_price":"10.00"
  }
}
```

Referenced

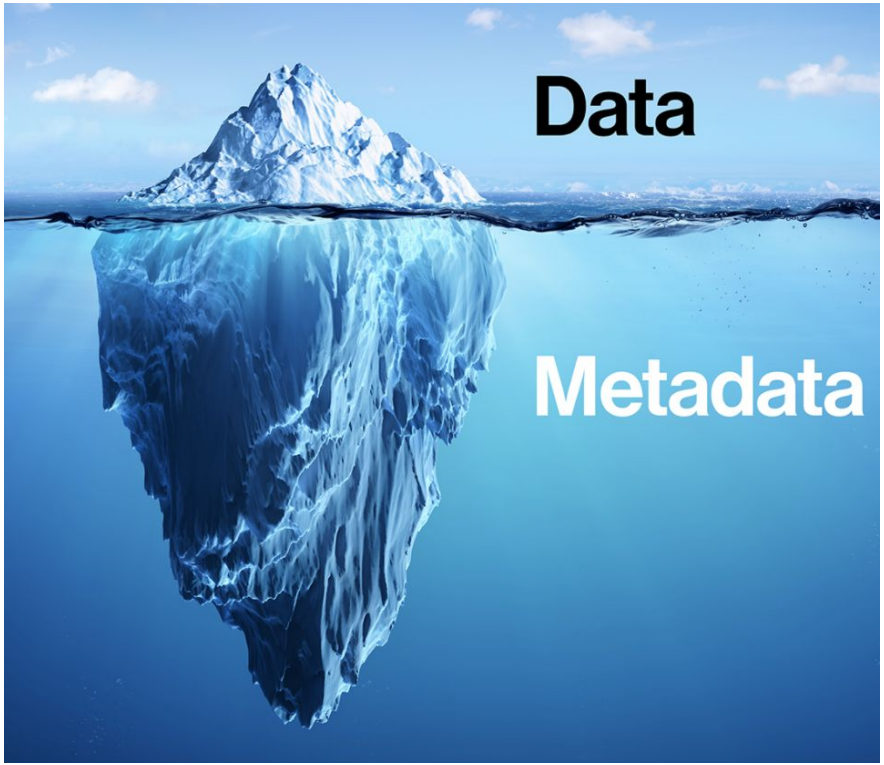
```
{
 "order_ID":"0001",
"Product":"0002"
}

{ "product_ID":"0002",
   "product_name":"Pencil",
  "product_category":"supplies",
  "list_price":"10.00"
}
```

General rules of thumb

Store Type	Primary Modeling Technique
SQL Data Stores	Normalizations to reduce data redundancy and improve data integrity
NoSQL Data Stores	Denormalization - to improve the read performance by adding redundant copies of data - motivated by performance or scalability Either no joins or minimal joins
Data Marts	Utilize Snowflake Schema & Star Schema
Graph Data Store	Edges & Vertices each with properties
In Memory Data Stores	Key/Value pairs, quick access

Metadata Management

- Catalog



Metadata (data about data) Management

“a cross-organizational agreement on how to define informational assets for converting data into an enterprise asset.

As data volumes and diversity grow, metadata management is even more critical to derive business value from the gigantic amounts of data. “

Reference

Questions that metadata helps answer

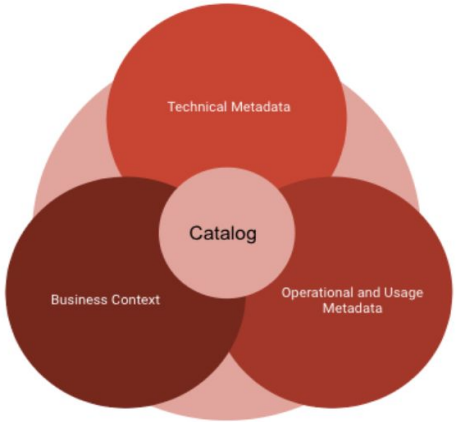
Who	What	Where	Why	When	How
Who created this data?	What is the business definition of this data element?	Where is this data stored?	Why are we storing this data?	When was this data created?	How is this data formatted? (character, numeric, etc.)
Who is the Steward of this data?	What are the business rules for this data?	Where did this data come from?	What is its usage & purpose?	When was this data last updated?	How many databases or data sources store this data?
Who is using this data?	What is the security level or privacy level of this data?	Where is this data used & shared?	What are the business drivers for using this data?	How long should it be stored?	
Who “owns” this data?	What is the abbreviation or acronym for this data element?	Where is the backup for this data?		When does it need to be purged/deleted?	
Who is regulating or adding this data?	What are the technical naming standards for database implementation?	Are there regional privacy or security policies that regulate this data?			

Metadata

Metadata management drives business value, improves innovation and collaboration, and helps mitigate risk. It also enables data citizens to access high-quality and trusted data, thus ensuring that they work with the right data to deliver accurate insights.

- Increasing trend of adoption on account of
  - Increasing need for **data governance**, regulatory and compliance requirements and data enablement
  - Increasing importance of higher **data quality and trusted analytics** driving business value from data
  - Growing complexity of data, with new sources augmenting the traditional sources
  - More business users actively interacting with data
  - Increasing need to accelerate transformation efforts, such as digitization, omnichannel enablement and enterprise-resource-planning modernization

- Types
  - **Business Metadata:** Defines everyday business terms
    - Eg. ontology, business rules, KPIs
  - **Technical Metadata:** Provides information on the format and structure of the data
    - Eg. schema, table relationship, data models, data lineage or access permissions.
  - **Operational Metadata**: Provides visibility into how data is being used and its quality
    - Eg. Data quality, SLAs, profiling, freshness, usage patterns



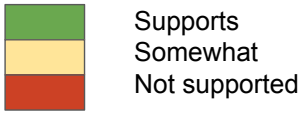
Catalogs

Manage and provide access to Metadata

Use Case	Description
Discovery	Probably the most valuable use case, the Catalog helps users - Data Engineers, Data Analysts, Data Scientists search, find and understand data
Data Governance	Document data lineage and data freshness for auditing and compliance purposes
Data Privacy & Risk	Discover and document sensitive data like PII and PHI and its flow through the data landscape
Data Value	Identify key data assets and assess their economic importance to the organization.

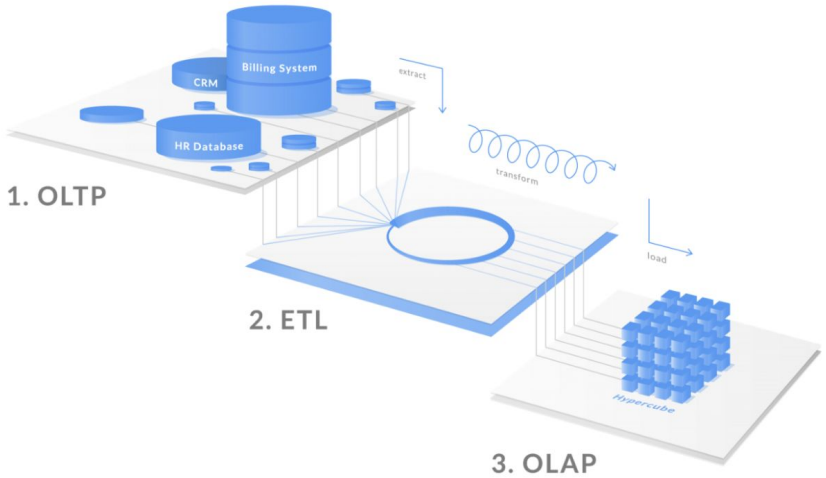
Catalog Providers & their Capabilities

	Collibra	Alation	ADC V2	Glue	GDC	Informatica	Data.world
Glossary							
Lineage							
Collab.							
Profiling							
DQ							
Workflows							
Prep							



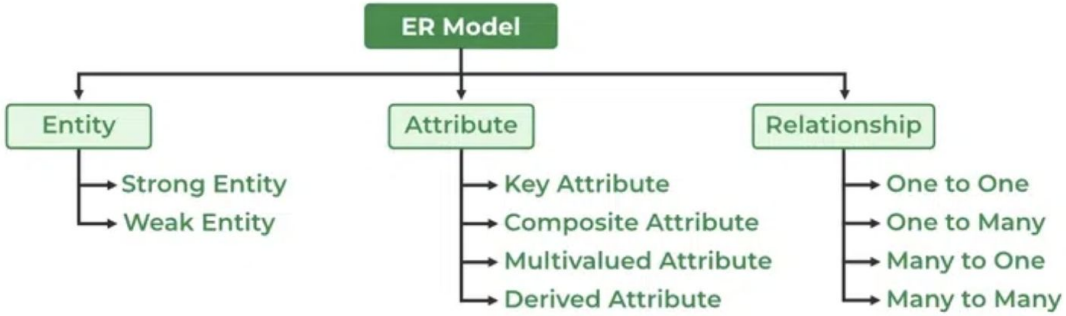
OLTP (ER) Vs OLAP(dim modeling)

- **Online Transaction Processing Systems aka OLTP Systems (ODS)**
  - The main data operation in the OLTP system is random read/write.
  - mainly employs ER aka entity-relationship models that meet 3NF to store data to solve data redundancy problems and inconsistency in transaction processing.
  - Ex. Relational stores
- **Online Analytic Processing Systems aka OLAP Systems (Analytics)**
  - focuses on data integration and performance of one-off, complex big data queries
  - Mainly employs dimensional modeling (eg. star, snowflake)
  - Ex. Data Warehouses



	OLTP	OLAP
Function	Day to day operation	Decision support
Database Design	Application oriented	Subject oriented
Data	Current, up-to-date detailed, flat relational, isolated	Historical, summarized multi-dimensional, consolidated
Usage	Repetitive	Ad-hoc
Access	Read/Write	Lots of scans
Unit of Work	Short, simple transaction	Complex query
Database Size	Gigabytes	Terabytes
Metric	Transaction throughput	Query throughput, response

Entity Relationship Diagram (ERD) [Link]

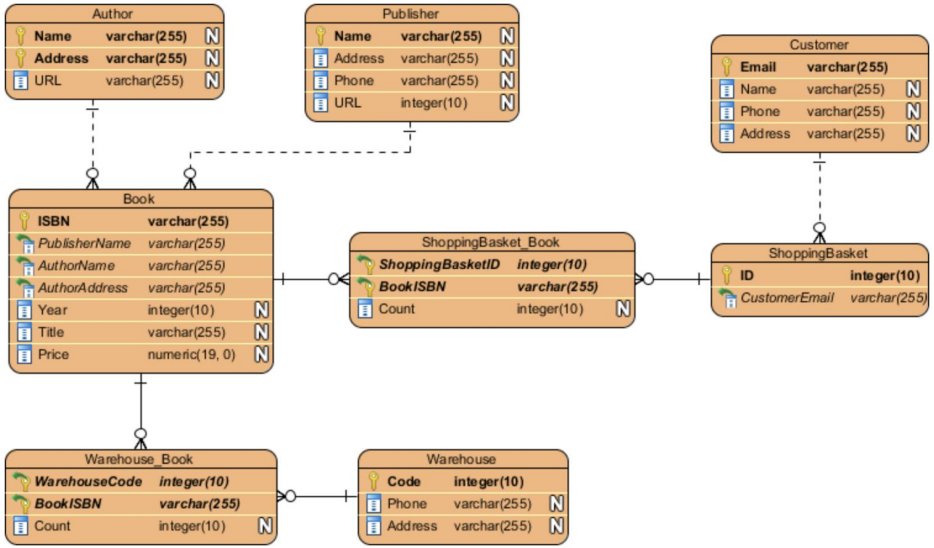


Entity

Attributes(PK/FK)

Relationship

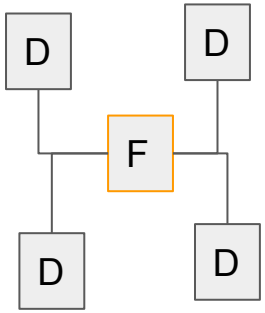
Cardinality



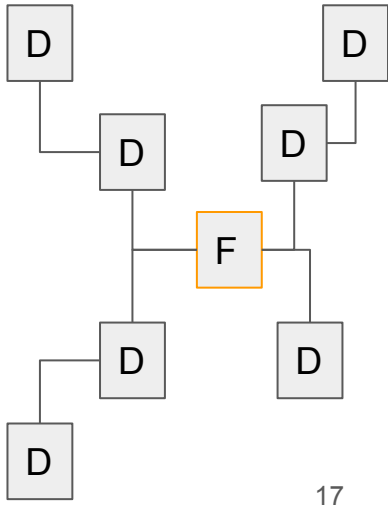
Dimensional Models: Star Vs Snowflake

Star Schema	Snowflake Schema
Hierarchies for the dimensions are stored in the dimensional table.	Hierarchies are divided into separate tables.
It contains a fact table surrounded by dimension tables.	One fact table surrounded by dimension table which are in turn surrounded by dimension table
In a star schema, only single join creates the relationship between the fact table and any dimension tables.	A snowflake schema requires many joins to fetch the data.
Simple DB Design.	Very Complex DB Design.
Denormalized Data structure and query also run faster.	Normalized Data Structure.
High level of Data redundancy	Very low-level data redundancy
Single Dimension table contains aggregated data.	Data Split into different Dimension Tables.
Cube processing is faster.	Cube processing might be slow because of the complex join.
Offers higher performing queries using Star Join Query Optimization. Tables may be connected with multiple dimensions.	The Snowflake schema is represented by centralized fact table which unlikely connected with multiple dimensions.

Star



Snowflake



ER Vs Dimensional Data Modeling

Data Processing

ER Modeling	Dimensional Modeling
Transaction oriented	Subject oriented
High Create/Read/Update/ Delete (CRUD) activity	High Read activity
Many users	Few users
Continuous updates	Batch updates
Real-time information	Historical information
Supports Business Operations	Supports Tactical and Strategic needs
Operational database	Informational database

Data Modeling

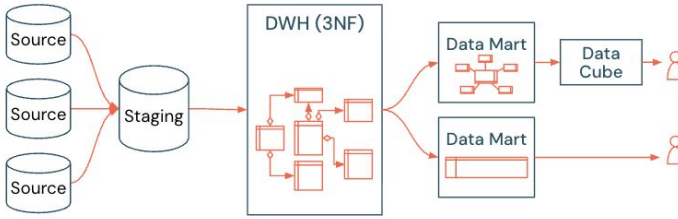
ER Modeling	Dimensional Modeling
Single purpose model - supports Operational System	Multiple models - support Informational Systems
Eliminate redundancy	Plan for redundancy
Natural keys	Surrogate keys
Validate Model against business Functional Analysis	Validate Model against reporting requirements
This moment in time is important	Many moments in time are essential elements
Normalization is suggested	De-Normalization is suggested

Data Application

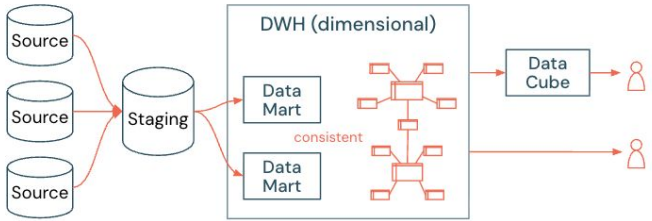
ER Modeling	Dimensional Modeling
High transaction volumes using few records at a time	Low transaction volumes using many records at a time
Balancing needs of online vs scheduled batch processing	Design for Reports for analytical processing
Highly volatile data	Non-volatile data
Data redundancy - *BAD*	Data redundancy - *GOOD*
Few levels of granularity	Multiple levels of granularity
OLTP Applications	OLAP Applications
Ex: *Applications used for Buying products from e-commerce sites like Flipkart, amazon etc*	**Ex:** Application to analyse buying patterns of customers of various cities for the past 10 years

Data Warehouse Modeling Techniques

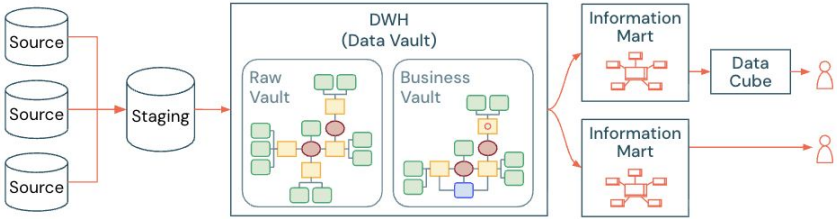
- **Top-down approach** as defined by Bill Inmon (~1992)
  - 3NF Relational Modeling with normalized data as the core of the DWH
  - Data marts (often dimensional or denormalized models)



- **Bottom-up approach** as defined by Ralph Kimball (~1996)
  - Dimensional Modeling
  - Denormalized data model, built as a star or snowflake schema:
  - fact tables surrounded by dimension tables

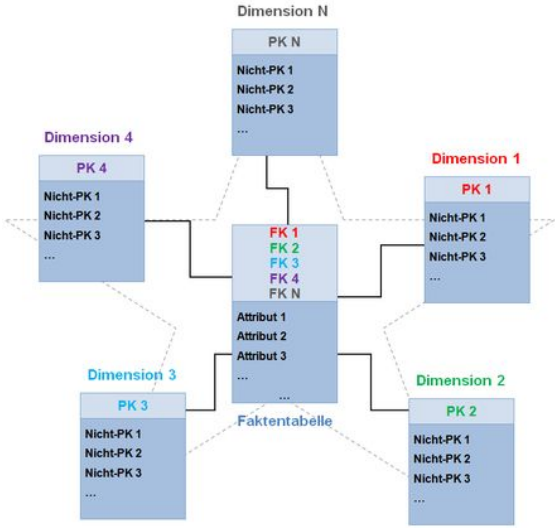


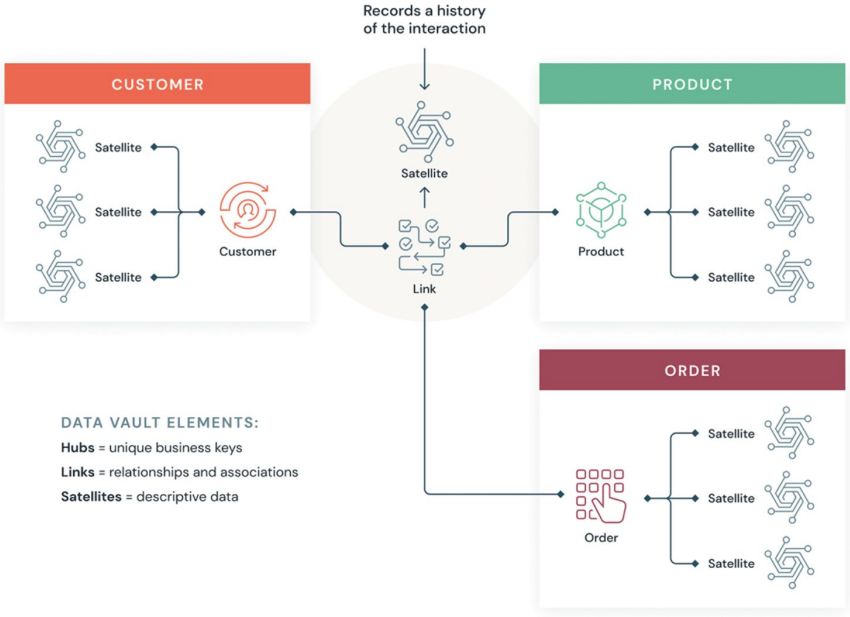
- **Data Vault** (v 2.0) as defined by Dan Linstedt (~2010)
  - **Hubs** separate core business concepts
  - **Links** store relationships between business concepts
  - **Satellites** store attributes of a concept/relationships



Data Warehousing aka analytics friendly modeling styles

- Star Schema
  - Facts
  - Dimensions
- Data Vault
  - Hubs
  - Links
  - Satelites





Data Vault addresses:

- Auditing
- Loading speed
- Resilience to change
- Traceability of data
- Captures all data

WHAT IS CONFORMED DATA

Conformed Data
- Sorted
- Arranged
- Grouped
- Consistent

NDH Data Lake → DATA

Layer 1 Raw Vault → SORTED

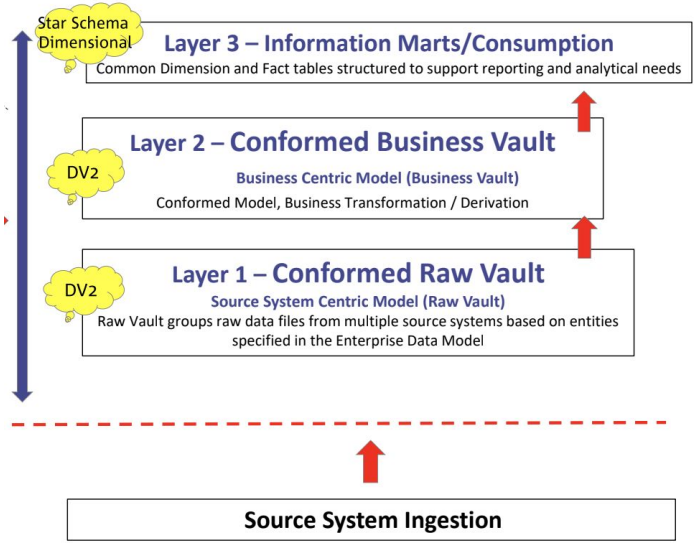
Layer 2 Business Vault → ARRANGED

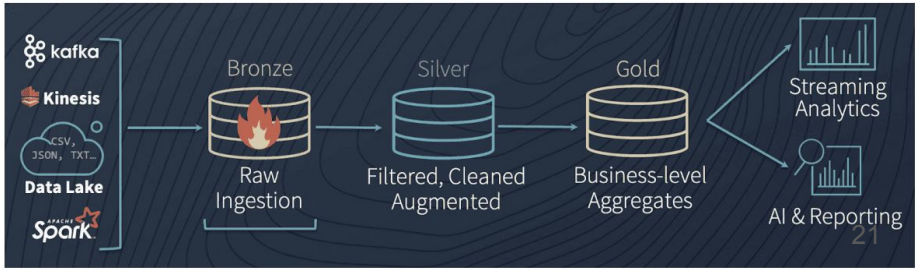
Layer 3 Conformed Info Marts → PRESENTED VISUALLY

Management Scorecards → EXPLAINED WITH A STORY

KPIs → ACTIONABLE (USEFUL)

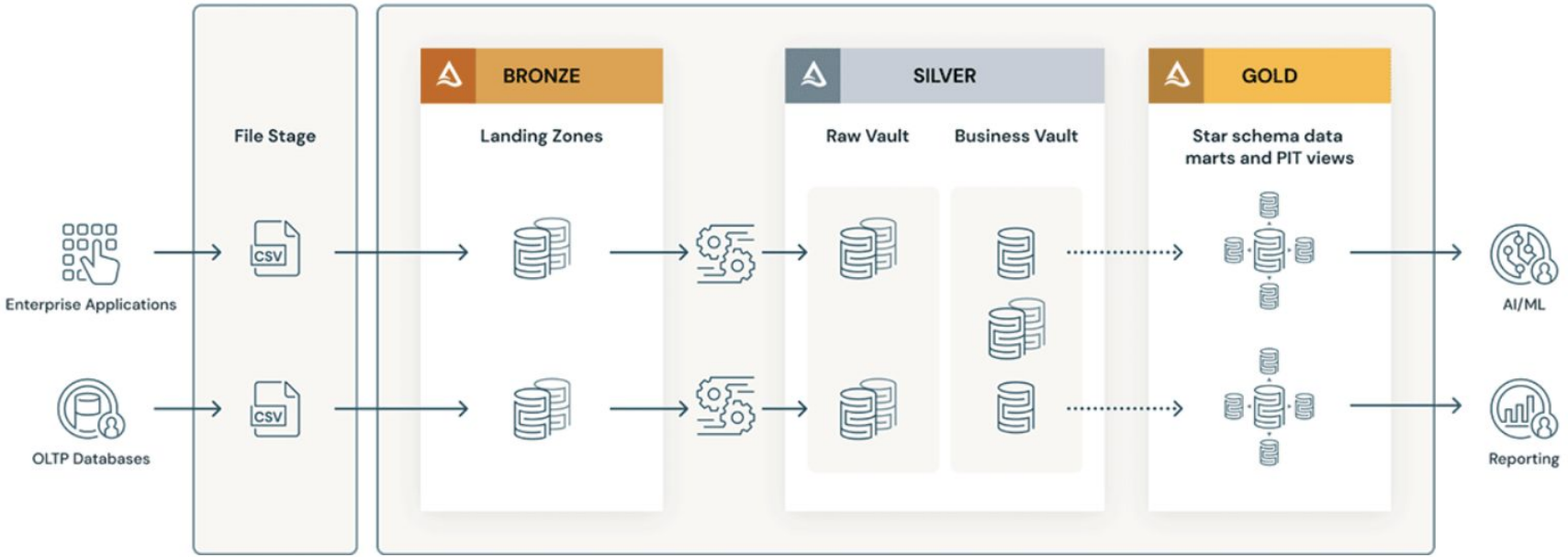
Youth Creativity, Innovation & Sustainable Leadership » Blog Archive » LinkedIn: The Lego Data Story (stanford.edu)





How a data vault fits in a Lakehouse

Medallion Architecture



Comparing the 3 modeling techniques

Inmon (NF)

Normalized Structure:
Inmon models follow a normalized approach, reducing data redundancy.

Single Source of Truth:
The data warehouse is designed as a single integrated repository.

Well-Suited for Large Enterprise Data Warehousing:
Recommended for large-scale data integration.

Cons: Complex to implement & maintain, slower query perf due to more joins, not as intuitive to business user

Kimball (dimensional)

Optimized for Reporting and Analytics:
Dimensional models are designed specifically for efficient querying and reporting. They provide a clear structure for business users to understand.

Easy to Understand:
Business users find dimensional models intuitive due to their star schema or snowflake schema representation.

Well-Suited for Smaller Projects: Quicker to implement for smaller-scale data marts.

Cons: No Single Source of Truth:
Data marts are organized around business areas, which can result in multiple versions of the same data.

Data Vault

Operational Flexibility:
Data Vault allows you to stay close to the source data, making it auditable and scalable.

Easier to Add New Sources:
Data Vault is flexible and accommodates new data sources seamlessly.

Historical Data Tracking:

Inherent support for historical data**.**

Cons: Data Vault models can become complex, especially when directly populating data marts from them. you might still need dimensional modeling for virtual data marts.Populating data marts from Data Vault models can lead to complex joins and recursions.

Source: bing copilot 2024-02-07
"Compare "Dimensional Modeling" (Kimball), Inmon modeling, Data Vault v2. What are the advantages and what are the challenges"

Points to note

- Physical Data modeling techniques depend on the underlying data store
  - Ex. Relational data modeling is different from nosql
- Design a system not a schema
  - As a schema evolves continuously
  - The system components include business information requirements, corporate governance and security, the physical storage used for the data, integration and open interfaces for all types of data, and the ability to handle a variety of different data types.
- Focus on the core data that is critical to your business and get that modeled correctly first
  - It is easy to get overwhelmed with all the datasets around you
- Improve quality of metadata
  - The more you know about each piece of data, the more you can place it properly into the data models that support your business.
- Identify key consumption patterns aka key entities
  - Ex. location, date, product, customer, etc.

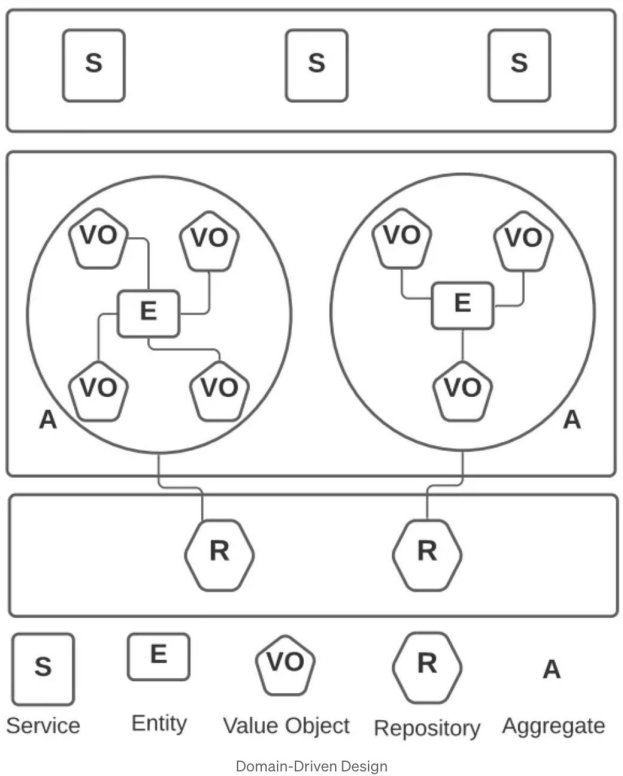
Domain specific modeling specifications

DDD - Domain Driven Design

OMG - Open Management Group

Eg.

P & C Data Model For Property And Casualty Insurance

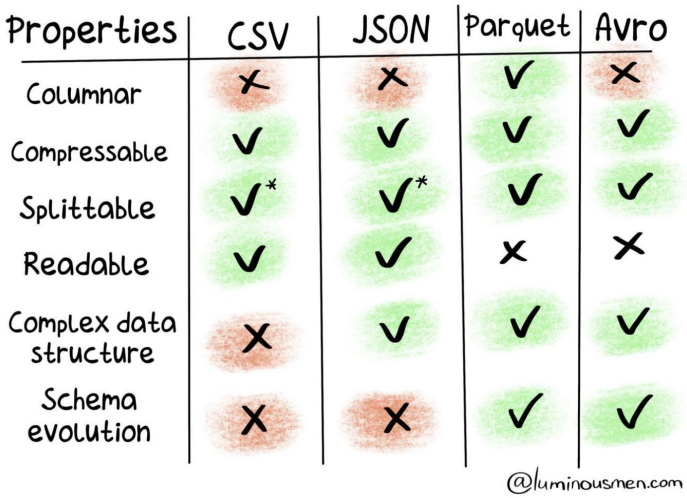


Domain-Driven Design

Choosing a data format (the ‘L’ in ETL) aka storage

- Not all **tools** support all of the data formats
- Are you constrained on **storage** or **memory**?
- How is your raw data structured?
- If your data is not splittable we lose the parallelism that allows fast queries.
- How many columns are being stored and how many columns are used for the analysis?
- Does your data **change** over time? If it does, how often does it happen and how does it change?

Data Formats



- Text format Vs Binary format
  - Human readable are text format
    - text format: Ex. csv, json
    - binary format: Ex. orc, parquet, avro
- Row Vs Column formats
  - csv, json, avro, etc are row based
  - orc, parquet etc are column based
- When to use what
  - Data Storage
    - Row stores have the ability to write data quickly and hence are good for **transaction** processing,
    - Column stores are good at aggregating large volumes of data for a subset of columns. One of the benefits of a columnar database is its **fast query speeds**. In Big Data, column based data formats are preferred for analytic querying because of better compression and performance
  - Normalization Vs Denormalization
    - columnar databases prefer a denormalized data structure
    - Normalization of data makes updates to some information much more efficient in a row store

Data is generally read off disk by blocks of KB (or MB/GB), so a single read for 1 record brings in a lot more than what you may want. Different systems have different default block sizes.

Row oriented data stores	**Column oriented data stores**
Data is stored and retrieved one row at a time and hence could read unnecessary data if some of the data in a row are required.	In this type of data stores, data are stored and retrieve in columns and hence it can only read the relevant data if required.
Records in Row Oriented Data stores are easy to read and write.	In this type of data stores, read and write operations are slower as compared to row-oriented.
Row-oriented data stores are best suited for online transaction system.	Column-oriented stores are best suited for online analytical processing.
These are not efficient in performing operations applicable to the entire datasets and hence aggregation in row-oriented is an expensive job or operations.	These are efficient in performing operations applicable to the entire dataset and hence enables aggregation over many rows and columns.
Typical compression mechanisms which provide less efficient result than what we achieve from column-oriented data stores.	These type of data stores basically permits high compression rates due to little distinct or unique values in columns.

Rules of thumb

- Why is it a good idea NOT to use a text format even though it is human readable?
  - Bad space utilization (numbers as strings waste space)
  - No type or structural checking (chars could wind up in numeric fields)
  - CSV — no metadata/header info, JSON — repeated meta/formatting
  - No native compression of repeating values
  - No native indexing /search ability
- Avro Vs Parquet?
  - Parquet(Columnar) for better query response and storage/capacity considerations
  - Avro(row based) for better schema evolution
- Delta
  - is Parquet + more including schema evolution, ACID transaction

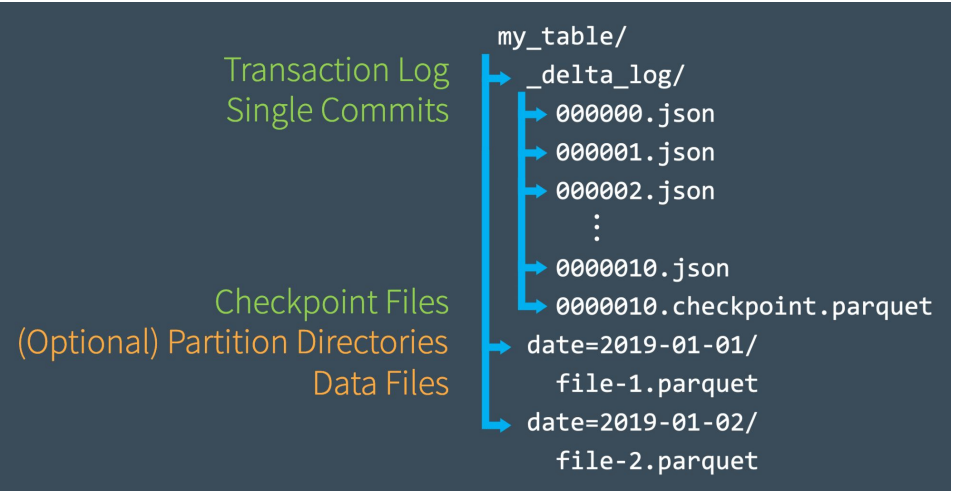
Delta

It is a open source data format

At its core it is parquet + transactional logs

A table built on top of Delta data format is called a Delta Table

A Data store of Delta Tables is called a Delta Lake



[Why Delta]

Data reliability with rich schema validation and transactional guarantees while improving performance.

Rather than performing an expensive LIST operation on the blob storage for each query, which is what the regular Parquet reader would do, the Delta transaction log serves as the manifest.

Simplified data pipeline with flexible UPSERT support and unified Structured Streaming + batch processing on a single data source.

Data Compression

- **The process of encoding information using less bits (data) than the original representation**.
  - Data compression is useful to save disk space or reduce the I/O or bandwidth used when sending data (e.g., over the internet, or from storage to RAM)

- Compression algorithms generally come in two varieties: lossy and lossless.
  - Lossy compression is highly advantageous for images, audio, and video because it removes data beyond a certain level of fine-grain detail. In fact, most people will not notice that information is missing.
  - The typical database user, however, does not want to lose any of the data inserted into the database. Even a tiny change in the data stored could make it unusable.

- *Codec = Compressor + Decompressor*
  - All compression algorithms must make a trade-off between the degree of compression and the speed of compression/decompression.

Codec	File Extension	Splittable ?	Degree of Compression	Compression Speed
Gzip	.gz	No	Medium	Medium
Bzip2	.bz2	Yes	High	Slow
Snappy	.snappy	No	Medium	Fast
LZO	.lzo	No, unless indexed	Medium	Fast

Understanding your data

Data Profiling

Data quality problems cost U.S. businesses more than $3 trillion a year.

The process of reviewing source data, understanding structure, content and interrelationships, and identifying potential for data projects. The process of examining, analyzing, and creating useful summaries of data. The process yields a high-level overview which aids in the discovery of data quality issues, risks, and overall trends. Data profiling produces critical insights into data

df.describe() df.summary()

Advantages	● Better data quality and credibility ● Predictive decision making ● Proactive crisis management ● Organized sorting
Techniques	● Structure discovery ○ helps determine whether data is consistent and formatted correctly. ○ It uses basic statistics to provide information about the validity of data. ● Content discovery ○ focuses on data quality. Data needs to be formatted, standardized, and properly integrated with existing data in a timely and efficient manner. ● Relationship discovery ○ identifies connections between different data sets.

Data correlation

Statistical Analysis

Univariate involves the analysis of a single variable while **multivariate** analysis examines two or more variables. Most multivariate analysis involves a dependent variable and multiple independent variables.

Although most real-world research examines the impact of multiple independent variables on a dependent variable, many multivariate techniques, such as linear regression, can be used in a univariate manner, examining the effect of a single independent variable on a dependent variable.

Displaying results in an understandable format is especially important in multivariate analysis because of the greater complexity of these techniques.

- Factor Analysis, Cluster Analysis, Variance Analysis
- Discriminant Analysis, Principal Component Analysis, Redundancy Analysis

Data Visualization (analyze trends and outliers)

- **Univariate**
  - Bar Charts
    - Useful while making comparisons between categories of data or different groups of data. It helps to track changes over time.
  - Pie Charts
    - Gives an overview of the group of data broken into smaller pieces that reflects in each slice of the pie.
    - The whole pie represents 100 percent, and the slices represent the relative size of that group or category.
  - Histograms
    - Similar to bar-charts, however histograms focus on distribution of variables binned instead of just comparisons
    - The height of the bars signifies the number of components in that category. The bin indicates the number of data points in a range.
  - Frequency Distribution Tables
    - The frequency distribution reflects the frequency of an occurrence in the data.
- **Multivariate**
  - Scatter Plots
    - It shows the measure of the influence of one variable on the other.
  - Regression Analysis
    - It is used to analyze how the data is related to each other.
  - Correlation Coefficients
    - It analyzes if the variables are related.
    - “0” suggests that the variables are not related to each other, and “1” reveals a positive or a negative one.

Summary of best practices

- Understand Requirements and Model the data for consumption patterns
- Profile data to understand trends and characteristics
- Establish quality guard rails
- Data Classification to identify sensitive data and other unique characteristics
- Update Metadata to educate other users on the dataset particulars
- Follow established design patterns
- Choose file format carefully
- Compress data from the start
- If necessary use co-processors ex. FPGA or specialized processors such as GPUs
- Match compression type to Data
- Combine with Data deduplication
- Build use cases off curated zone data that is refined

Homework

- Simplifying Data Eng - Chap 3 & 4

Assignment-1 posted - Due Sep 20th

Appendix

Big Data Strategy for business transformation

Data As an Asset/Service/Product

1. Define Data Strategy
   - Understand Enterprise Objectives and Goals
2. Assess Current State
   - Identify Data Sources
     - Transactional & non-tx data
   - Identify Use Cases
3. Collaboration
   - Develop a Roadmap
     - Data Integration
     - Decision Tools
     - Analytic Models
4. Data Democratization
5. Data Governance
6. Embed through Change Management
   - Educate, Explore, Engage, Execute
   - Deposit, Discover, Design, Decide

Big Data System Design Considerations

Foundations	● Reliability ● Scalability ● Maintainability	Hardware/Software/Human related Load/Performance Operations/Maintenance/Evolution
Data Models	● Relational ● No-SQL ● Graph ● Document	ER Denormalizing Nodes/Edges JSON
Storage	● Encoding ● Encryption	ORC, Parquet, Avro
Trade-offs	● Cost/Performance ● Load/Speed ● Complexity	CAP ACID BASE
Distributed	● Compute ● Storage	Clocks Consistency/Consensus

Case Study - 1

- You'll be assigned teams, each team has ~5 members and is assigned a firm to study
- Research the firm's data engineering challenges and solutions
- What Business problem were they trying to solve?
- How did technology help them?
- How well did they execute? What could they have done better?
- Document your findings in a short presentation (criterias based on the rubric provided)
- Record your presentation (~15 min)
- Upload the slide deck and video recording to Canvas

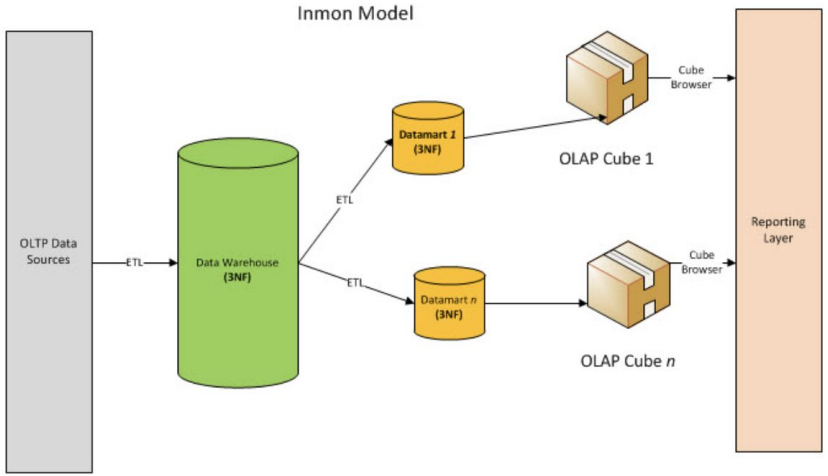
Data Modeling

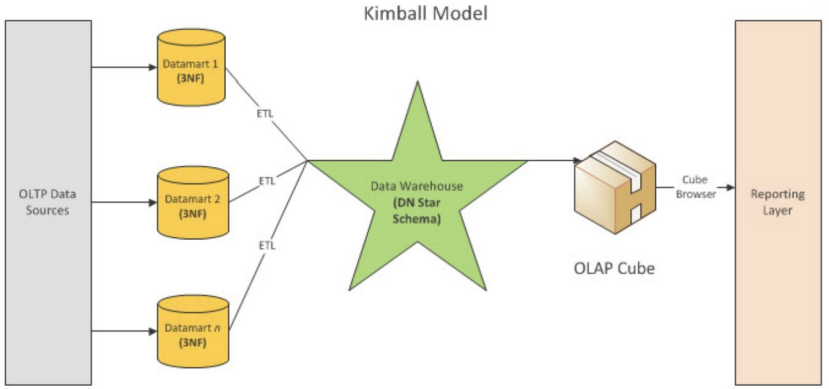
> "***Poor programmers care about code, and good programmers care about the data structure and the relationships between data***".
>
> -- Linus Torvalds, the founder of Linux, on the importance of data modeling

Undoubtedly, a big data system requires high-quality data modeling methods for organizing and storing data to achieve optimal balance of performance, cost, efficiency, and quality.

- **Performance:** Good data models can help us quickly query the required data and reduce I/O throughput.
- **Cost:** Good data models can significantly reduce unnecessary data redundancy, reuse computing results, and reduce the storage and computing costs for the big data system.
- **Efficiency:** Good data models can greatly improve user experience and increase the efficiency of data utilization.
- **Quality:** Good data models make data statistics more consistent and reduce the possibility of computing errors.

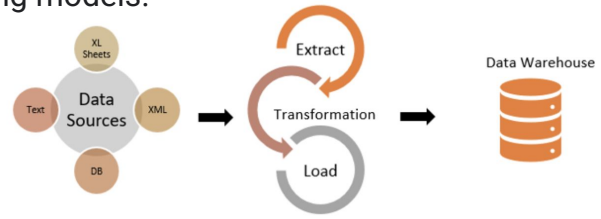
Relational Data Modeling Approaches





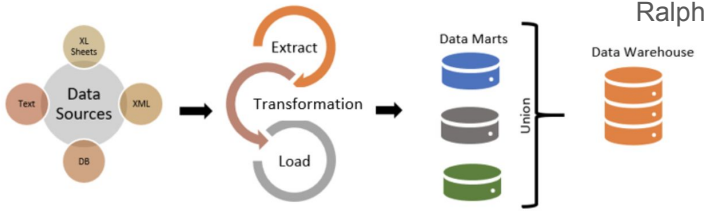
top-down or data-driven approach:
Start from the data warehouse and break it down into data marts, specialized as needed to meet the needs of different departments within the organization, such as finance, accounting, HR etc
It is more structured and easier to maintain while it is rigid and takes more time to build. DW is in 3NF; it is easier to build data mining models.

Bill Inmon



bottom-up
Here the dimensional data marts are first created to provide reporting and analytical capabilities for specific business areas such as “Sales” or “Production”.
More scalable.The ROI is usually faster. Reuse is low

Ralph Kimball

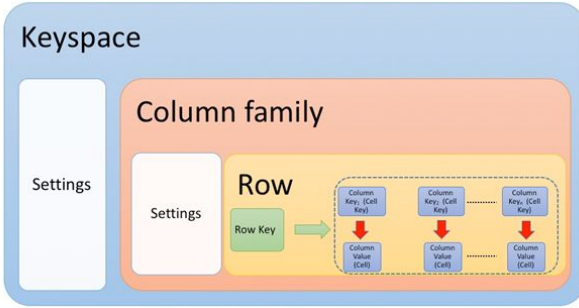


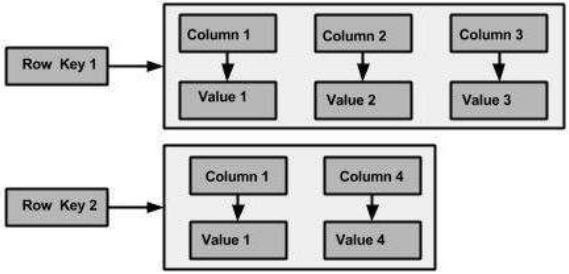
Steps (Blog1) (Blog2) (Blog3)

- Create catalog, schema, table
- Primary Key & Foreign Key definitions
- Identity keys for surrogate keys
- Column constraints for data quality
- Index, Optimize, Analyze
- Advanced techniques

Handling nuances of Big Data/NoSQL modeling

- Handling aggregates:
  - Composite key
    - Include multiple data elements in a “composite” row key, which can be useful for grouping rows together for finding by key range.
    - If you want the most recent first use a reverse timestamp
    - Another option is to add a hash prefix to the row key in order to get good distribution, and still have a secondary grouping
  - Column Families
    - Ex. in Cassandra, HBase, DynamoDB which are columnar databases, related fields can be grouped in column families
      - Unlike a table in a relational database, different rows in the same table (column family) do not have to share the same set of columns.





- Tree, Adjacency List:
  - Relational databases are not very convenient for hierarchical or graph-like data modeling and processing. NoSQL solutions do well

    { "_id": "USA", “type”:”state”, **"children"**: [...], **"parent"**: null }
    { "_id": "TN", “type”:”state”, "children": ["..."], "parent": “USA” }