Data Integration and Large Scale Analysis

02 Data Warehousing and ETL

Dr. Lucas Iacono - 2025
Know Center Research GmbH & Graz University of Technology

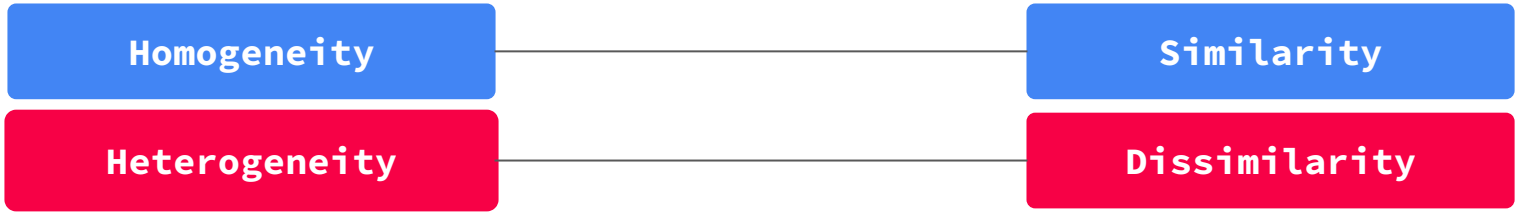
Agenda

- Recap
- Data Warehousing (DWH)
- Extraction, Transformation, Loading (ETL)
- SQL/OLAP Extensions

Recap

Recap: Data Sources and Heterogeneity

Some important concepts and keywords

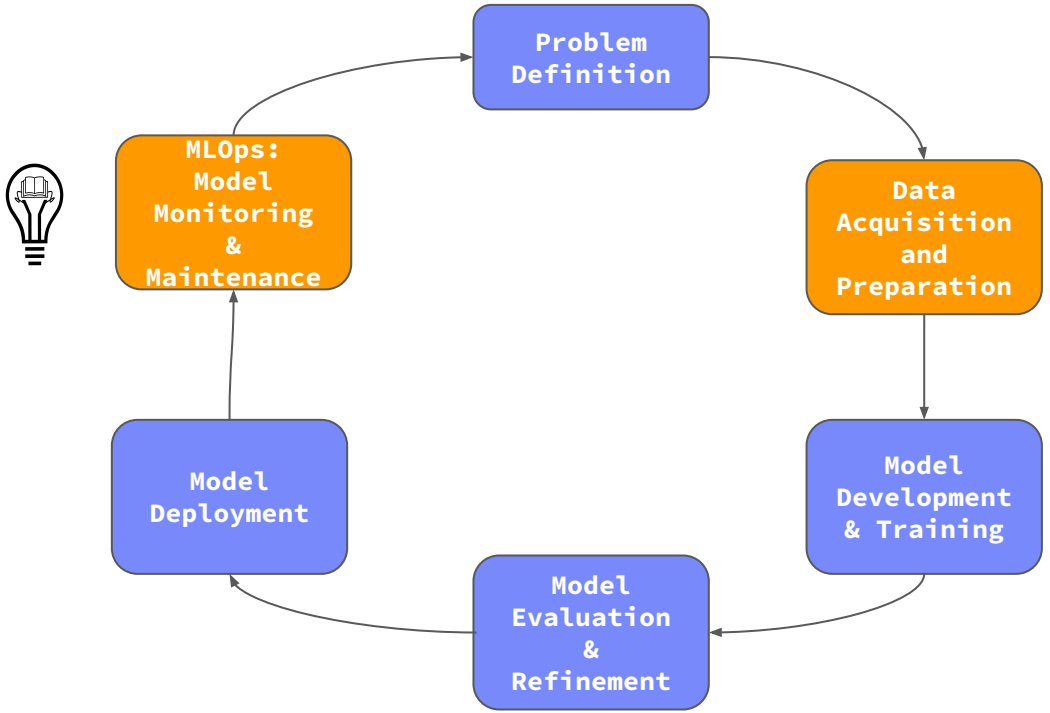


Register	Name	Class	Engine	Format
A	Corvette C7	GT3	Petrol	CSV
B	Porsche 911	GT3	Petrol	HDF5

- **Semantic** ---> Meaning (GT3 petrol race cars)
- **Ontology** ---> Conceptual Structure (Name/Class/Engine)
- **Format** ---> Physical Representation (Binary vs Text)

Recap: from data to models

AI Lifecycle

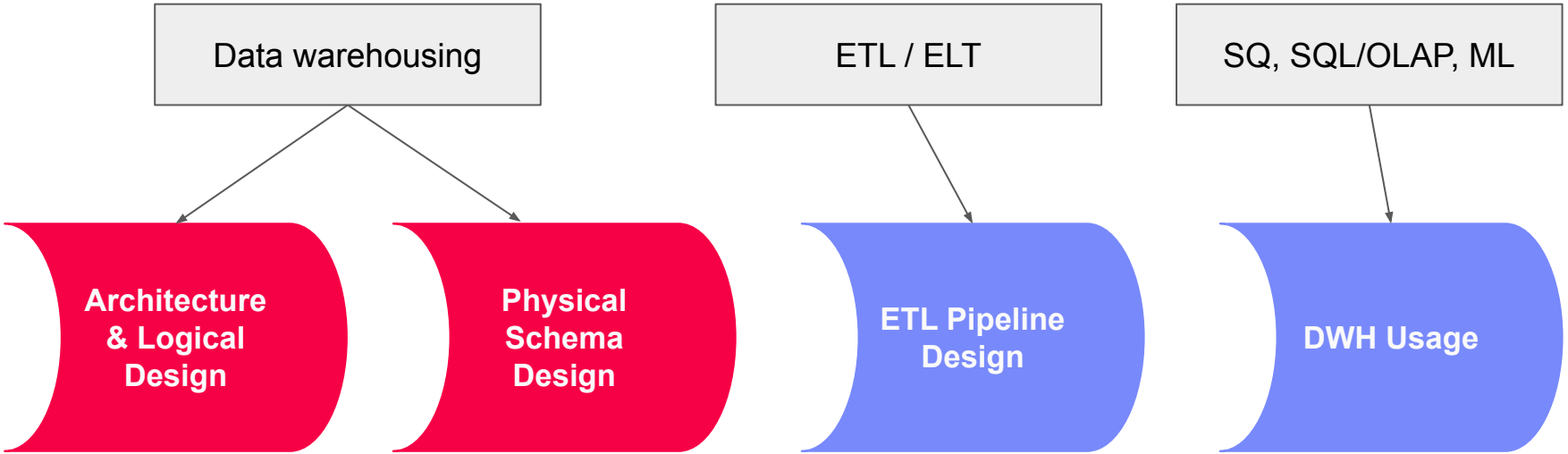


Ref: Jeffrey Saltz. Data Science Process Alliance.

Recap: course goals

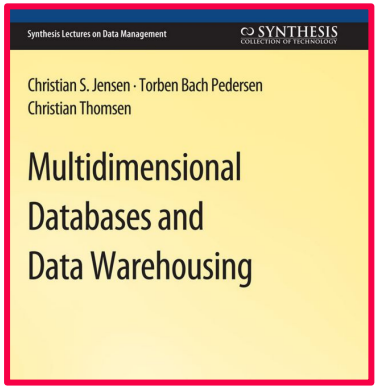
- Understand **major data integration architectures** and their role in modern data ecosystems.
- Apply **key techniques for data integration and cleaning** to ensure consistency, quality, and usability of data.
- Evaluate methods for **large-scale data storage and analysis**, with a focus on scalability and efficiency.

Today: DWH, ETL, SQL/OLAP Extensions

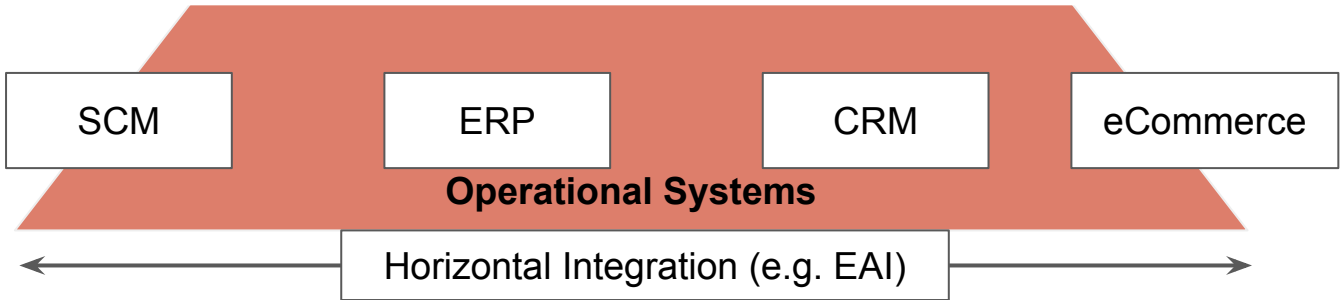

Data warehousing
ETL / ELT
SQ, SQL/OLAP, ML
Architecture & Logical Design
Physical Schema Design
ETL Pipeline Design
DWH Usage

Data Warehousing

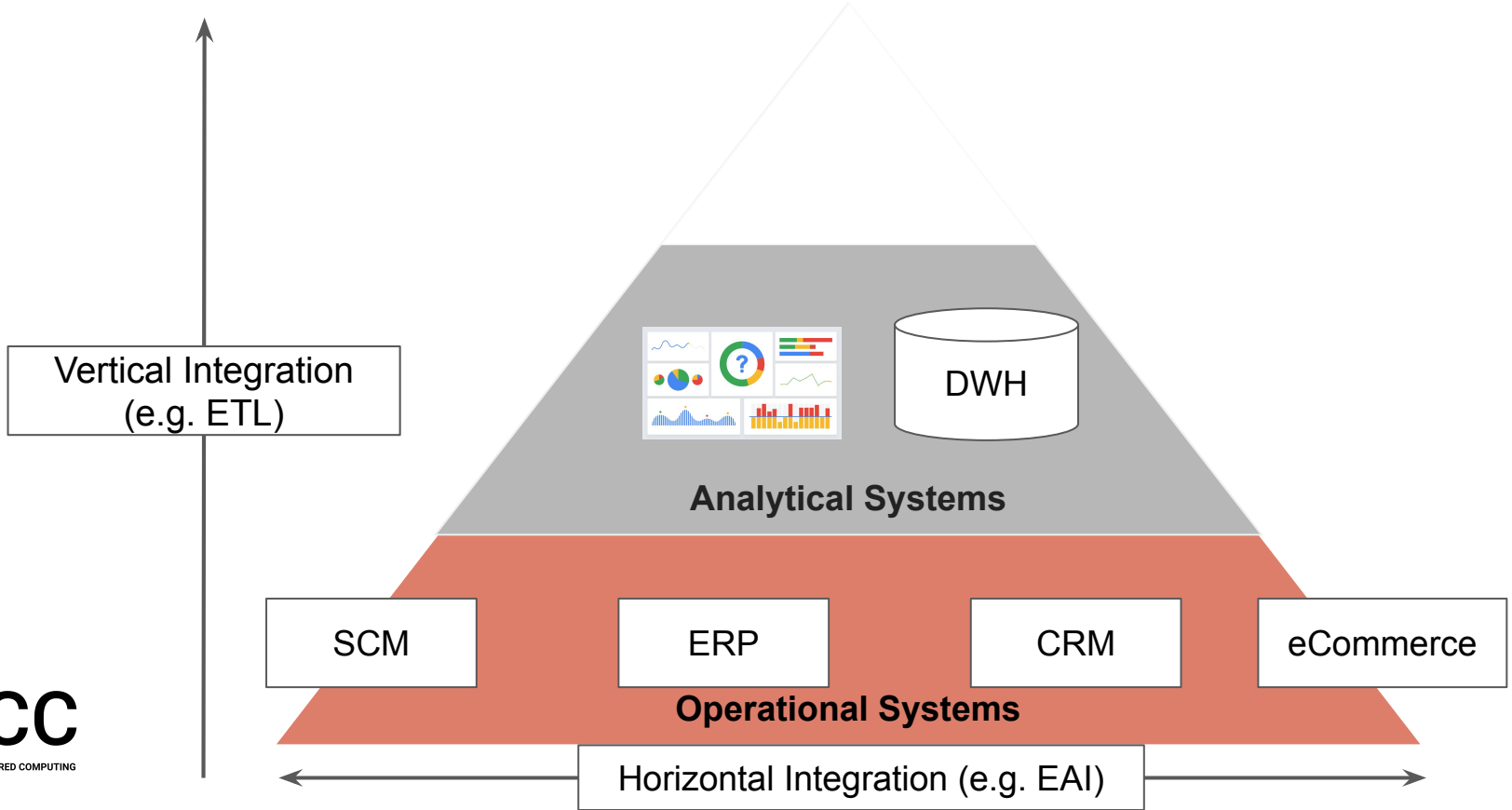
Complementary readings:



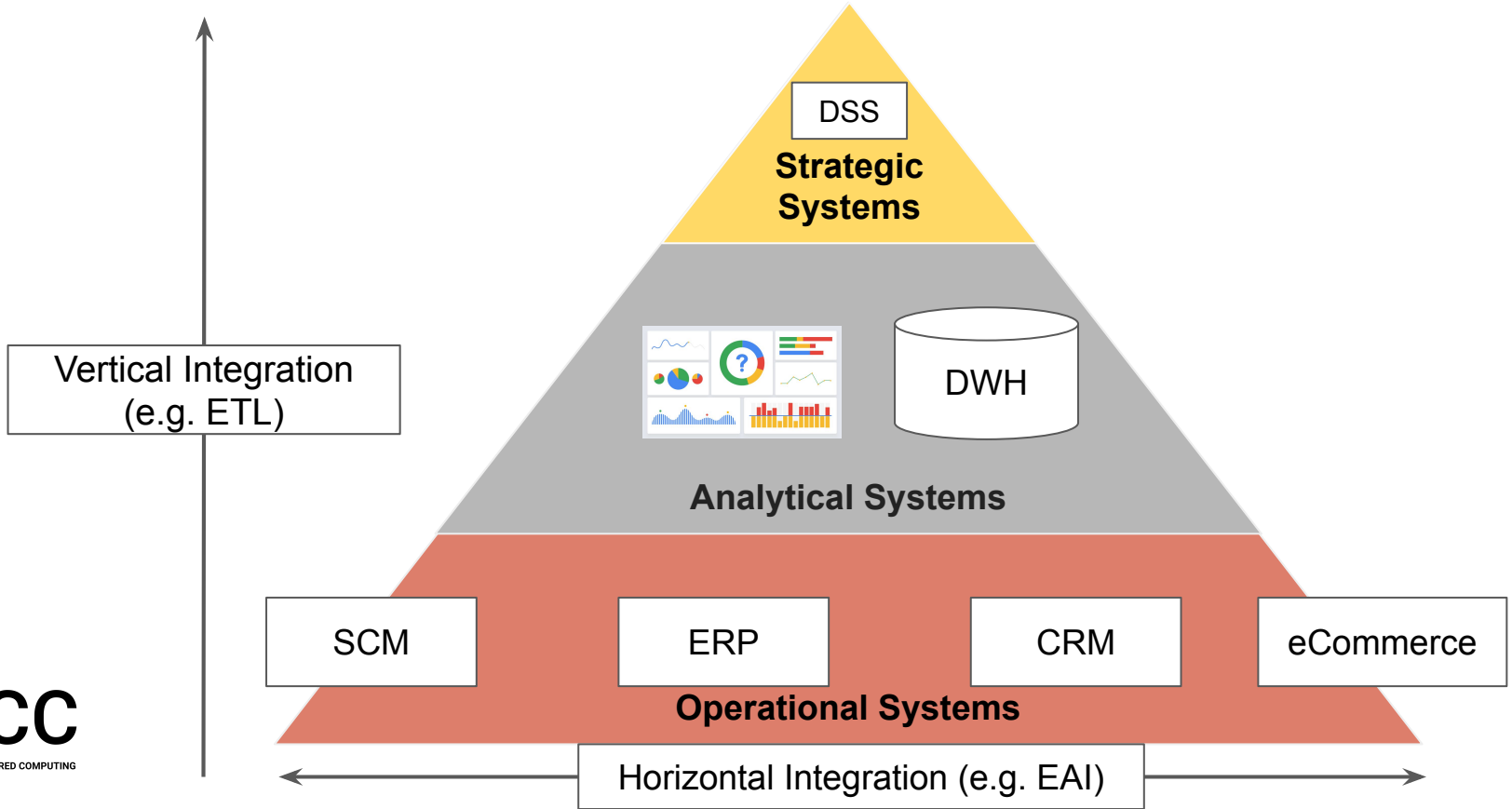
DWH: Integration Architecture



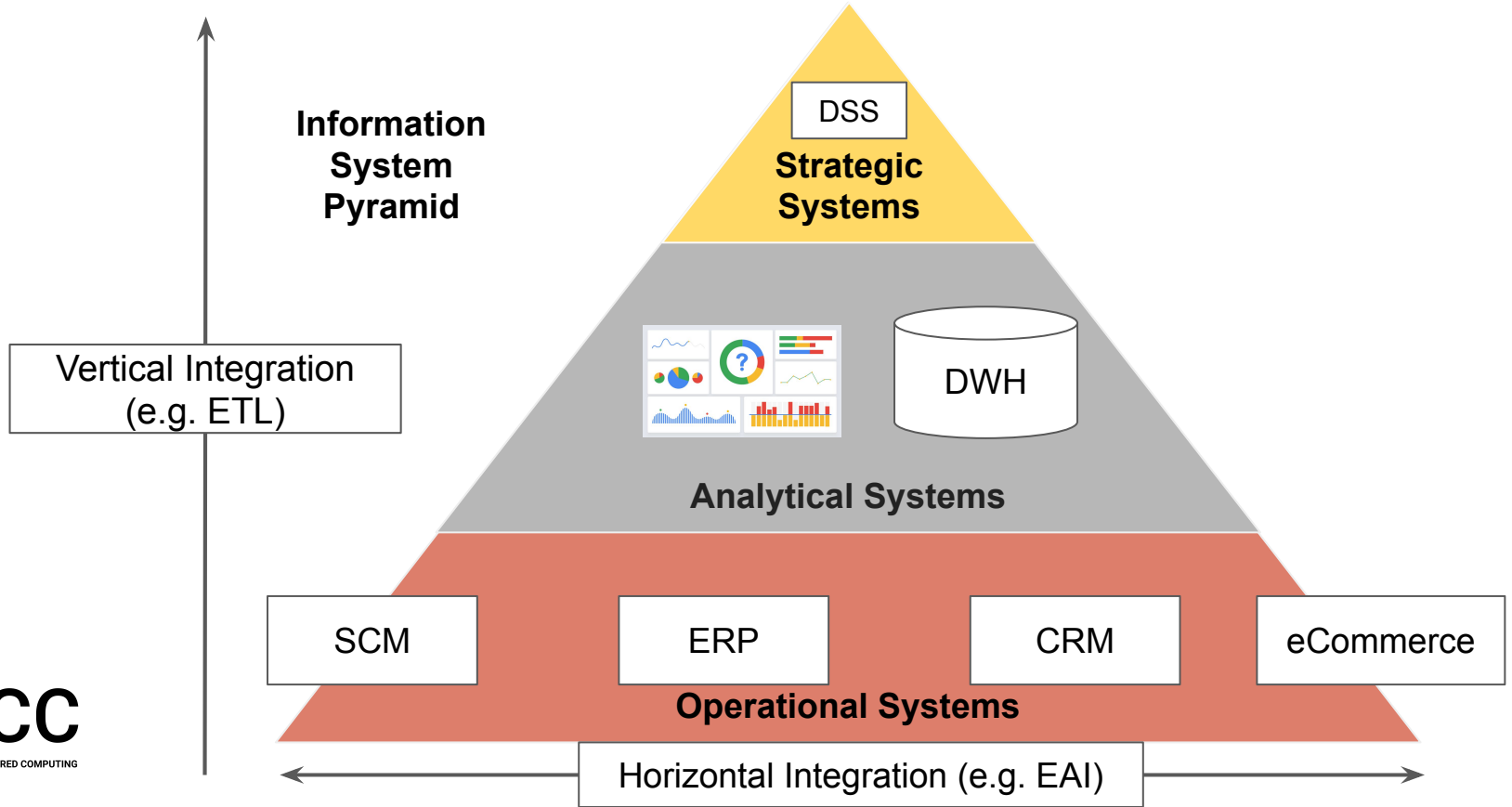
DWH: Integration Architecture



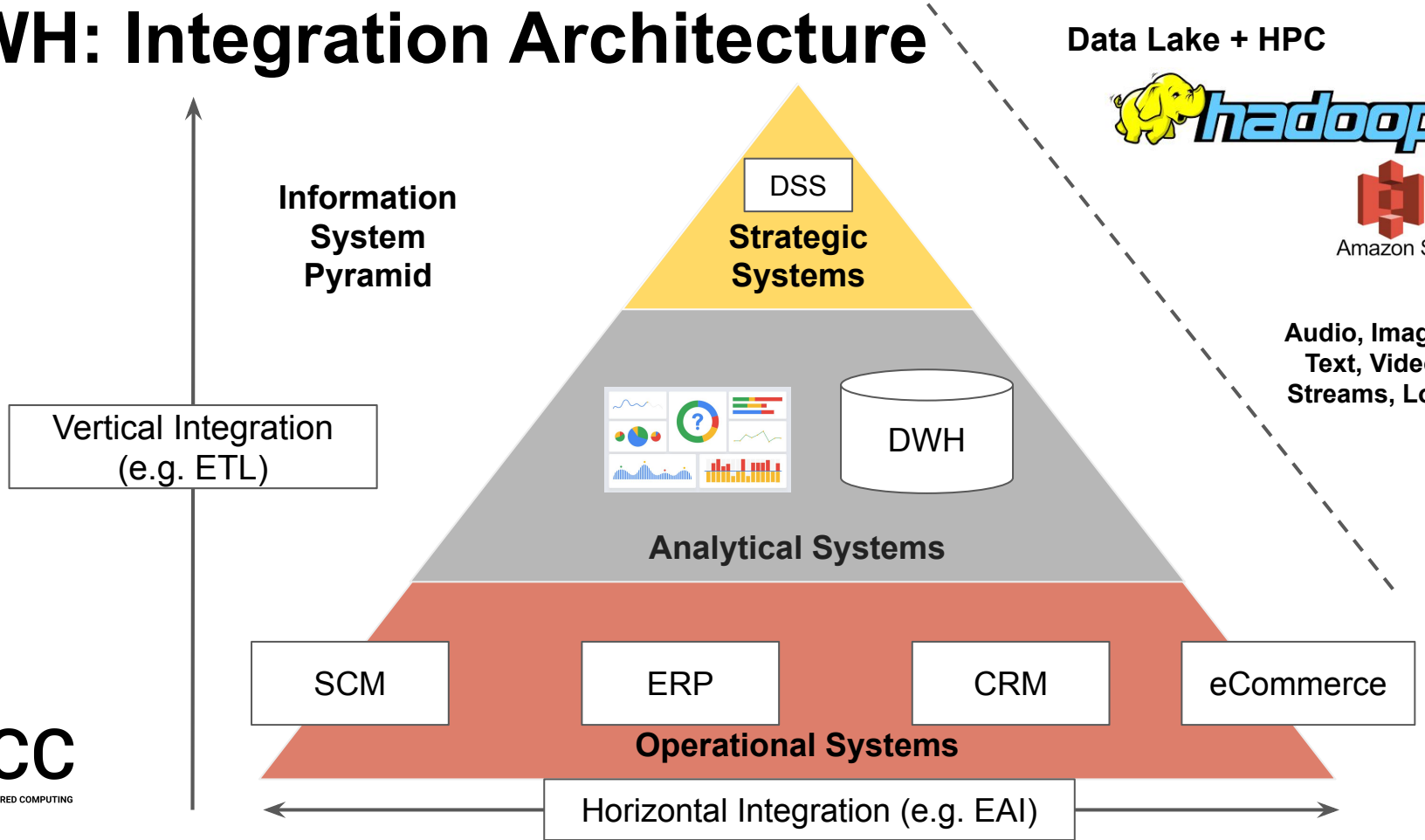
DWH: Integration Architecture



DWH: Integration Architecture



DWH: Integration Architecture

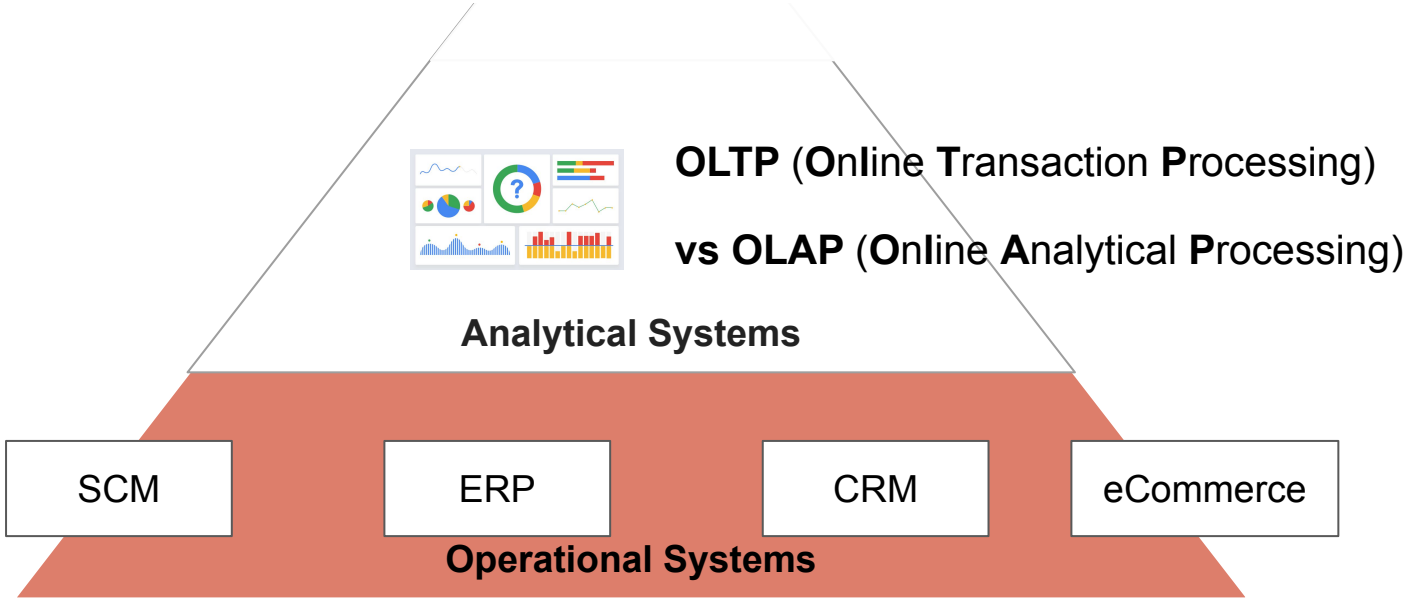


DWH: Motivations and Tradeoffs

- **Goal:** Queries over consolidated and cleaned data of several, potentially heterogeneous, data sources

DWH: Motivations and Tradeoffs

- **Goal:** Queries over consolidated and cleaned data of several, potentially heterogeneous, data sources



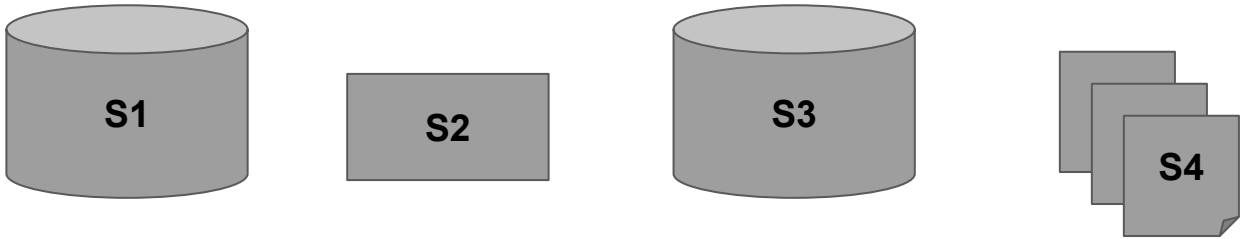
DWH: Motivations and Tradeoffs

- **Goal:** Queries over consolidated and cleaned data of several, potentially heterogeneous, data sources

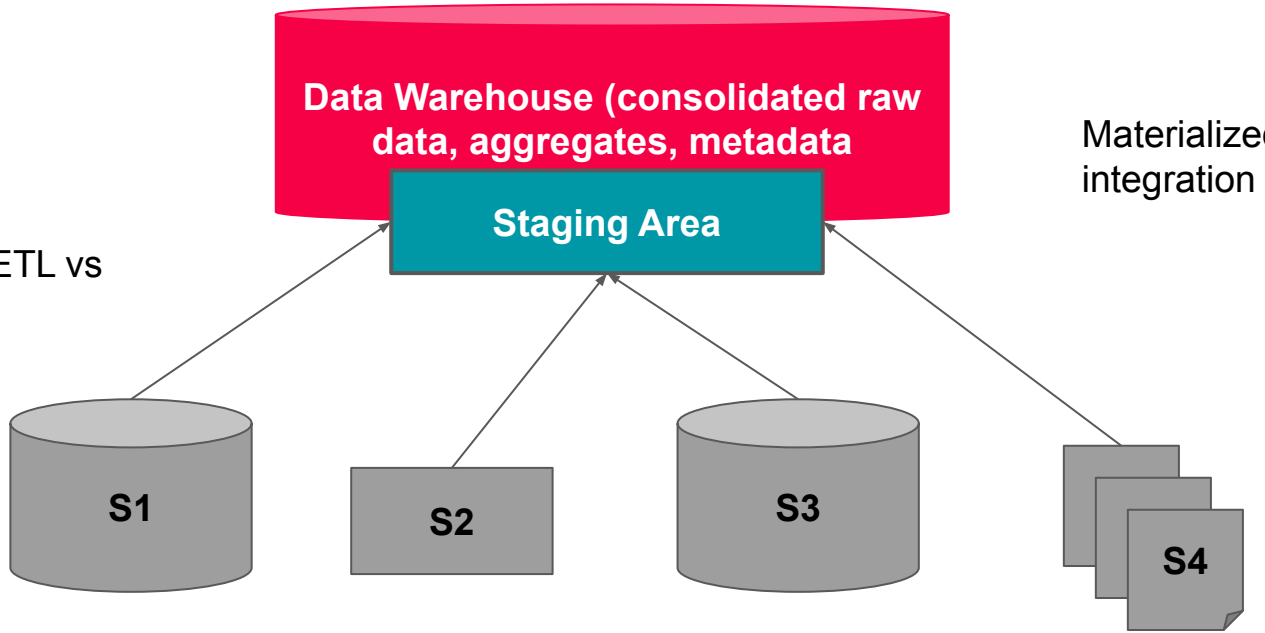
- **Tradeoffs** (OLTP vs OLAP)
  - **Analytical query performance:** write vs read optimized data stores
  - **Virtualization:** overhead of remote access, source systems affected
  - **Consistency:** sync vs async changes, time regime **->** up-to-date?
  - **Others:** flexibility, redundancy

DWH: Architecture

Operational source systems



DWH: Architecture

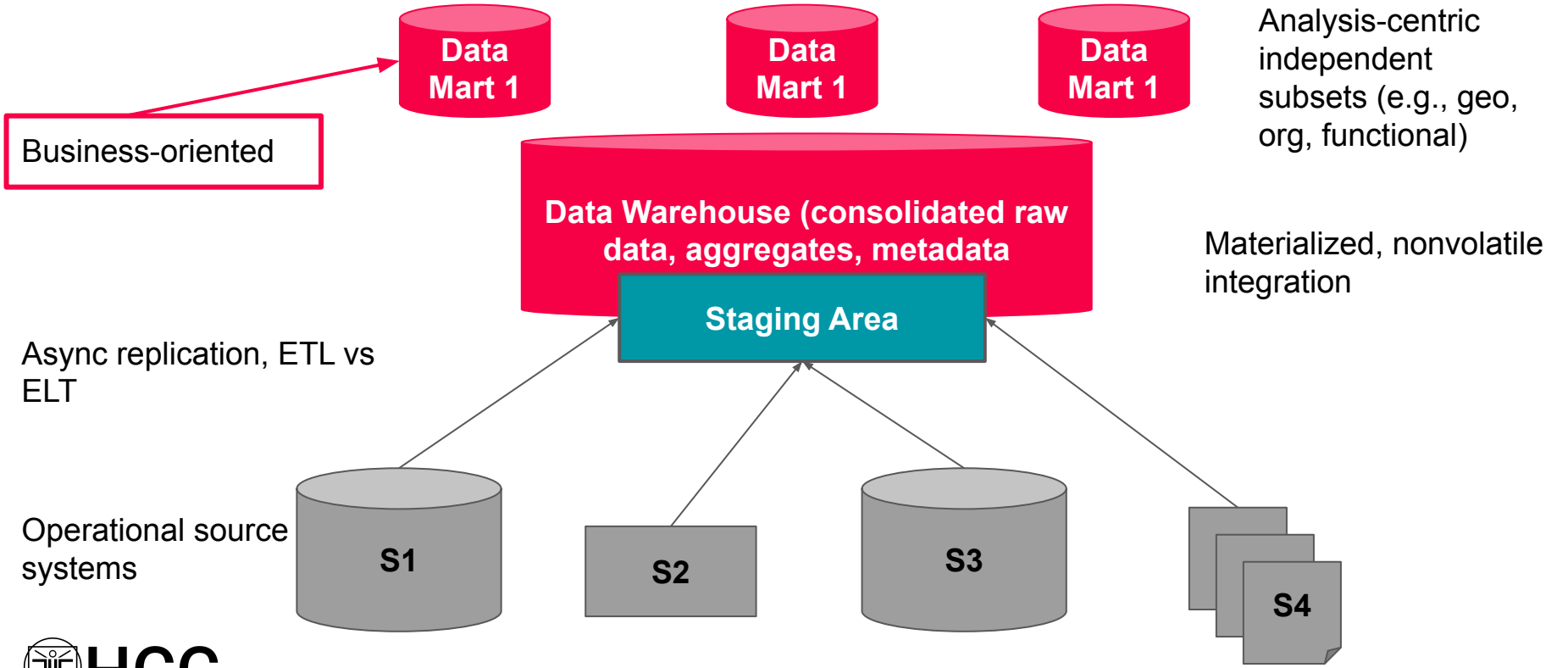


Materialized, nonvolatile integration

Async replication, ETL vs ELT

Operational source systems

DWH: Architecture



DWH: Architecture

- **Data Warehouse (DWH):** *“A data warehouse is a **subject-oriented**, **integrated**, **time-varying, non-volatile collection** of data in **support** of the management's **decision-making** process.” (Bill Inmon)*
  - **Subject-oriented:** analysis-centric organization (e.g., sales) -> Data Mart
  - **Integrated:** consistent data from different data sources
  - **Time-varying:** History (snapshots of sources), and temporal modelling
  - **Non-volatile:** Read-only access, limited to periodic data loading by admin

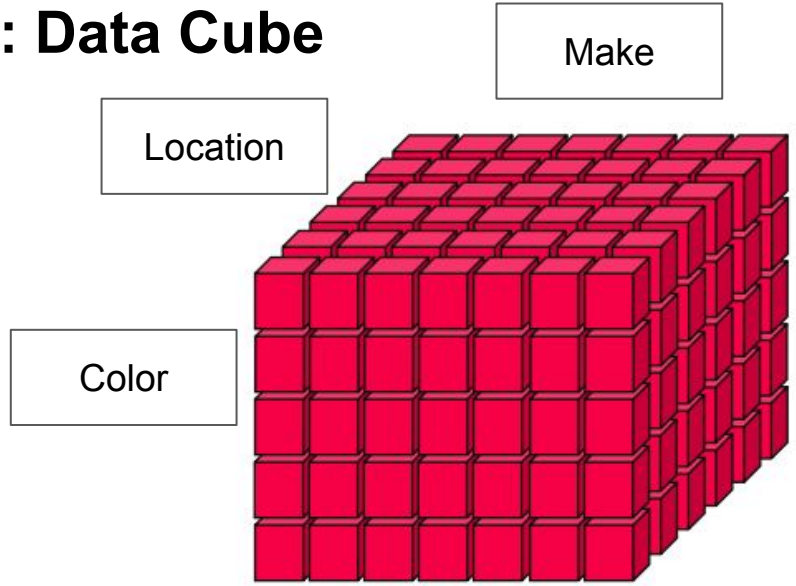
DWH: Architecture

- **Data Warehouse (DWH):** *"A data warehouse is a **subject-oriented**, **integrated**, **time-varying, non-volatile collection** of data in **support** of the management's **decision-making** process." (Bill Inmon)*
  - **Subject-oriented:** analysis-centric organization (e.g., sales) -> Data Mart
  - **Integrated:** consistent data from different data sources
  - **Time-varying:** History (snapshots of sources), and temporal modelling
  - **Non-volatile:** Read-only access, limited to periodic data loading by admin
- **DHW Instantiations:**
  - Data sources → staging areas → DWH → data marts **(top-down)**
  - Data marts → individual ETL → Data Sources →DWH **(bottom-up)**

DWH: Multi-dimensional Modeling: Data Cube

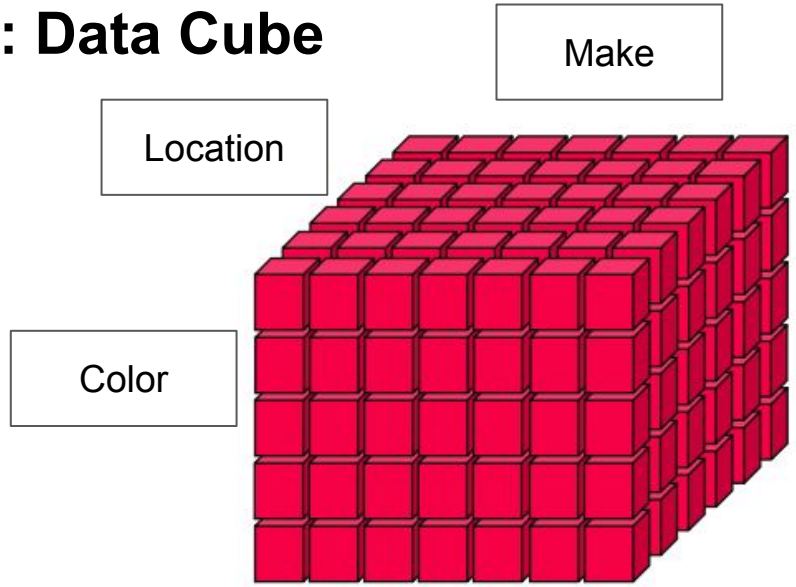
DWH: Multi-dimensional Modeling: Data Cube

- **Central Metaphor: Data Cube**
  - **Qualifying data** (categories, dimensions)
  - **Quantifying data** (cells)
  - **Often sparse** (0 for empty cells)



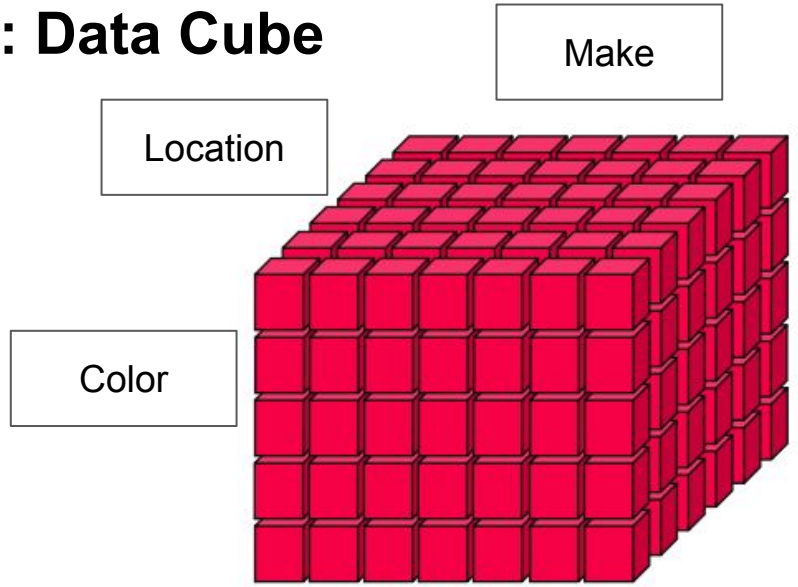
DWH: Multi-dimensional Modeling: Data Cube

- **Central Metaphor: Data Cube**
  - **Qualifying data** (categories, dimensions)
  - **Quantifying data** (cells)
  - **Often sparse** (0 for empty cells)
- **Multi-dimensional Schema**
  - Set of dimension hierarchies (D^1,…, D^n)
  - Set of measures (M^1,...,M^m)



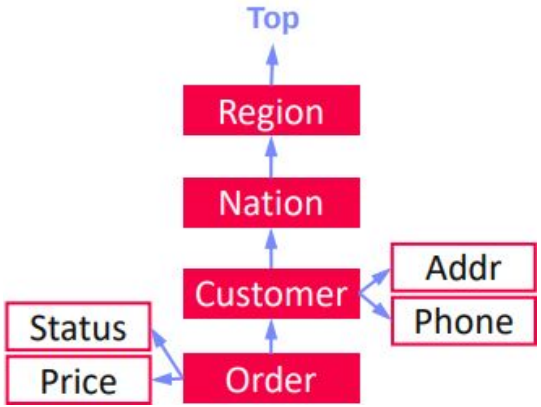
DWH: Multi-dimensional Modeling: Data Cube

- **Central Metaphor: Data Cube**
  - **Qualifying data** (categories, dimensions)
  - **Quantifying data** (cells)
  - **Often sparse** (0 for empty cells)
- **Multi-dimensional Schema**
  - Set of dimension hierarchies (D^1,…, D^n)
  - Set of measures (M^1,...,M^m)
- **Dimension Hierarchy**
  - Partially-ordered set D of categorical attributes
  - Generic **maximum element (TopD)**
  - Existing minimum element **(Day)**
  - **Functional dependency**



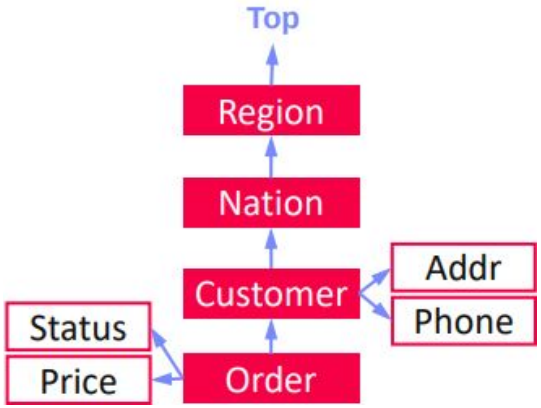
DWH: Multi-dimensional Modeling: Data Cube

- **Dimension Hierarchy, cont.**
  - Classifying (categorical)
    - Define the hierarchy used for aggregations
  - Descriptive attributes
    - Describe data used for filtering
  - **Orthogonal dimensions:** there are **no functional dependencies** between **attributes of different dimensions**



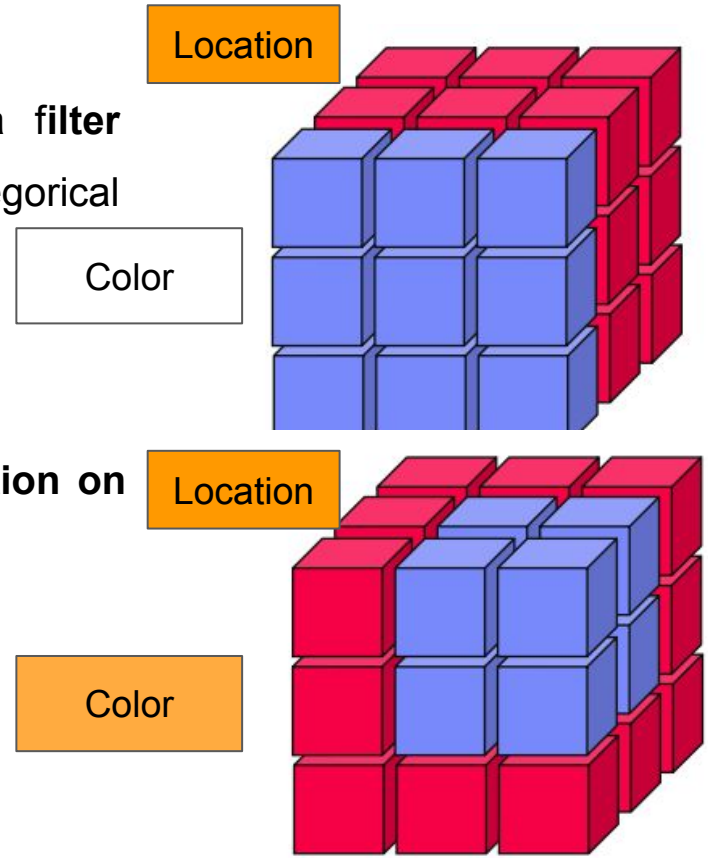
DWH: Multi-dimensional Modeling: Data Cube

- **Dimension Hierarchy, cont.**
  - Classifying (categorical) vs descriptive attributes
  - **Orthogonal dimensions:** there are **no functional dependencies** between **attributes of different dimensions**
- **Fact F**
  - Base tuples w/ measures of summation type (**e.g units sold**)
  - Granularity
- **Measure M**
  - Metrics computed over non-empty subset of facts in schema (**e.g. units sold per year**)
  - Scalar function (**temp offset)** vs aggregation function (**avg temp per day**)
  - Granularity



DWH: Multi-dimensional Modeling: Operations

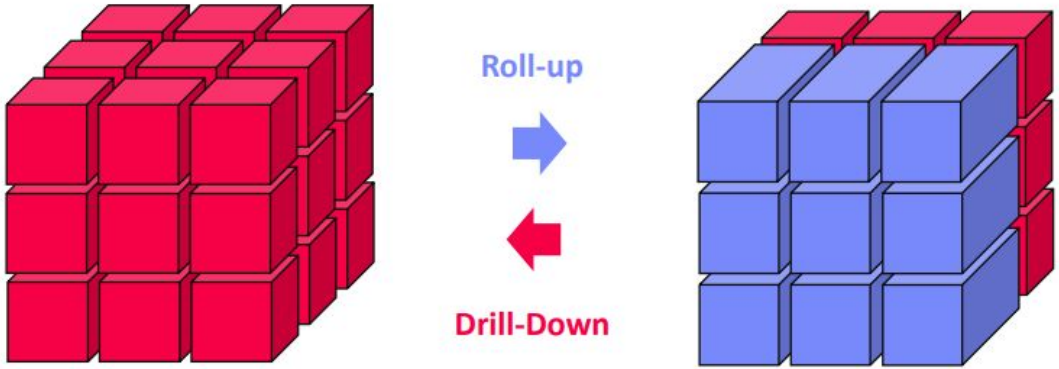
- **Slicing (cutting-out a single slice)**
  - Select a “slice” of the cube by specifying a **filter condition on one of the dimensions** (categorical attributes)
  - Same **data granularity** but subset of dimensions
- **Dicing (take a “chunk” of the cube)**
  - Select a “sub-cube” by specifying a **filter condition on multiple dimensions**
  - Complex Boolean expressions possible
  - Sometimes slicing used synonym



Example: Location=Graz AND Color = Orange AND Make= Hyundai

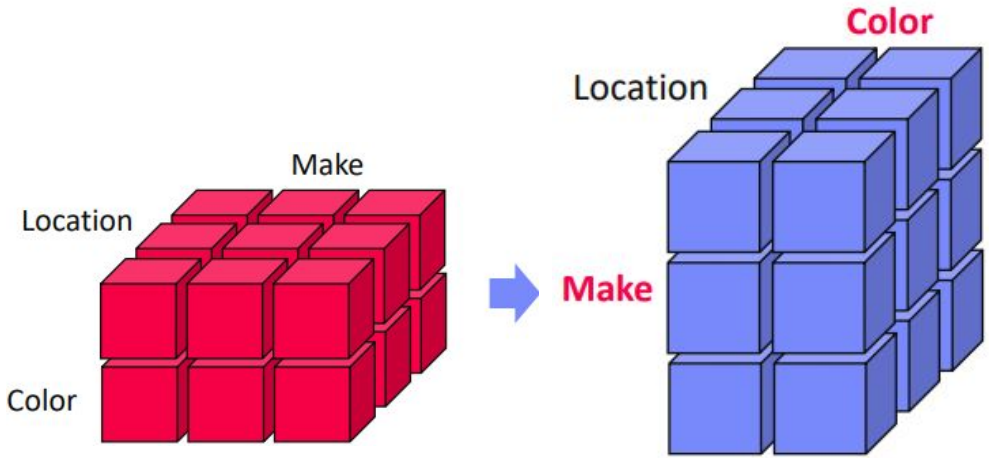
DWH: Multi-dimensional Modeling: Operations

- **Roll-up (similar Merge) → Zooming-out**
  - Aggregation of facts or measures into **coarser-grained aggregates** (measures)
  - Same dimensions but different granularity
- **Drill-Down (similar Split) → Zooming-in**
  - Disaggregation of measures into **finer-grained** measures



DWH: Multi-dimensional Modeling: Operations

- **Pivot (How the data looks)**
  - Rotate cube by exchanging dimensions
  - Swap rows per columns



DWH: Aggregation Types

- **Recap: Classification of Aggregates**
  - **Additive aggregation** functions (SUM, COUNT), can be summed across all cube dimensions [e.g. **Sales**]
  - **Semi-additive aggregation** functions (MIN, MAX), can be summed across some cube dimensions but not in others like time [e.g. **Amount of Employees**]
  - **Non-additive** functions, can't be summed across any dimension (AVG, STDDEV, VAR)

- **Summation Types of Measures**
  - **FLOW:** arbitrary aggregation possible (**e.g. sales**)
  - **STOCK:** aggregation possible, except over temporal dim (**e.g. bank account balance**)
  - **VPU:** value-per-unit typically (**e.g., price per product**)

Prog	16/17	17/18	18/19	19/20	20/21	Total
CS	1153	1283	1321	1343	1368	?
SEM	928	970	939	944	985	?
ICE	804	868	846	842	849	?
Total	2855	3121	3106	3129	3202	?

DWH: Multi-dimensionality

- **MOLAP (Multi-Dim. OLAP)**
  - **OLAP server** with **native multi-dimensional data storage**
  - Dedicated query language: Multidimensional Expressions (MDX)
  - IBM Cognos Powerplay, Essbase

DWH: Multi-dimensionality

- **MOLAP (Multi-Dim. OLAP)**
  - **OLAP server** with **native multi-dimensional data storage**
  - Dedicated query language: Multidimensional Expressions (MDX)
  - IBM Cognos Powerplay, Essbase
- **ROLAP (Relation OLAP)**
  - **OLAP server** w/ storage in **RDBMS**
  - All **commercial RDBMS vendors**

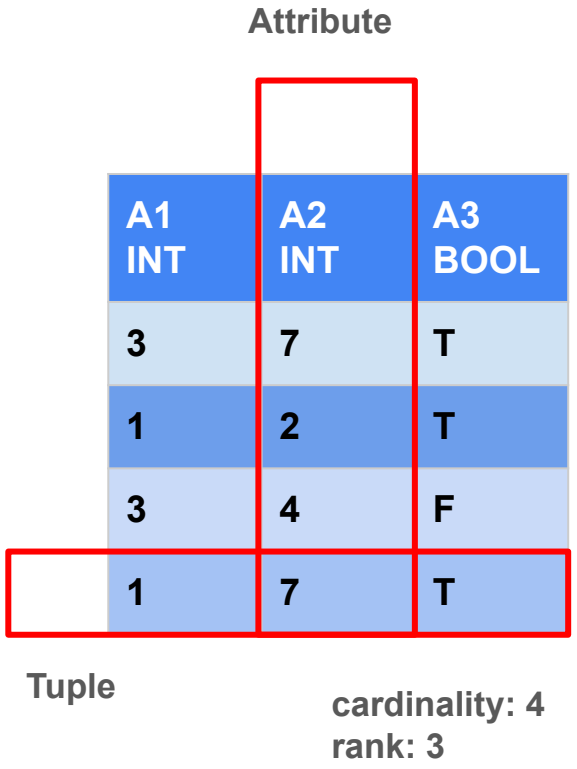
DWH: Multi-dimensionality

- **MOLAP (Multi-Dim. OLAP)**
  - **OLAP server** with **native multi-dimensional data storage**
  - Dedicated query language: Multidimensional Expressions (MDX)
  - IBM Cognos Powerplay, Essbase
- **ROLAP (Relation OLAP)**
  - **OLAP server** w/ storage in **RDBMS**
  - All **commercial RDBMS vendors**
- **HOLAP (Hybrid OLAP)**
  - **OLAP server** w/ storage in RDBMS and **multi-dimensional in-memory caches and data structures**

Requires mapping to relational model

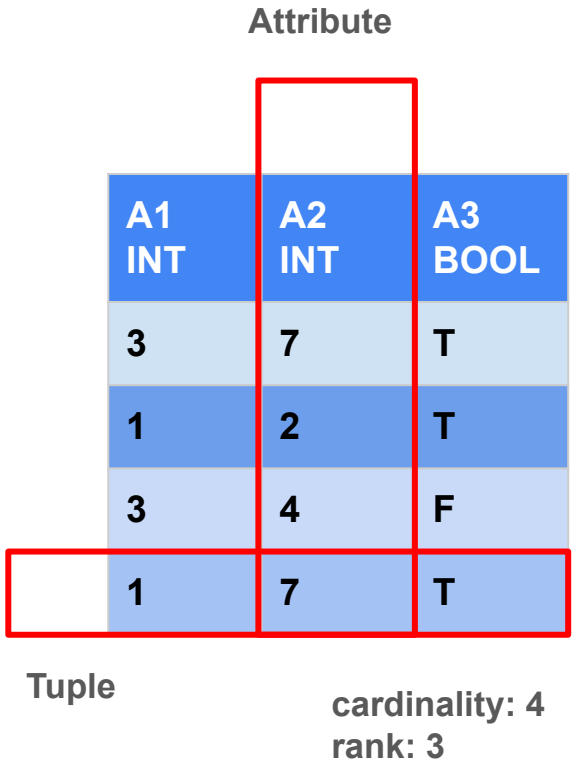
DWH: Relational Data Model (RECAP)

- **Domain D**: set of possible values (INT, CHAR[20], BOOL).
- **Relation R (table):**
  - **Relation Schema** (RS): set of attributes {A₁,…,A□} with **domains**.
- **Additional Terminology**
  - **Tuple:** row of k elements in a relation. **[1 with 3]**
  - **Cardinality** (relational): number of tuples in the relation. **[4]**
  - **Rank** (degree): number of attributes. **[3]**
  - **Semantics:**
    - Set semantics → no duplicates
    - Bag semantics → duplicates allowed (practical use)
  - **Order irrelevant:** neither tuples nor attributes have a defined order.



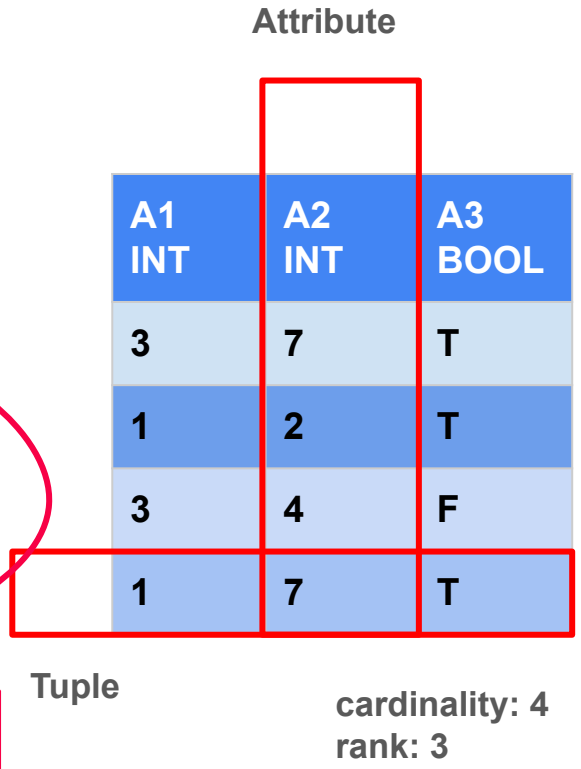
DWH: Relational Data Model (RECAP)

- **Domain (D)**: set of possible values (INT, CHAR[20], BOOL).
- **Relation (R):** subset of the Cartesian product of domains.
  - Relation Schema (RS): set of attributes $\{A_1,\ldots,A_n\}$ with domains.
- **Additional Terminology**
  - **Tuple:** row of k elements in a relation.
  - **Cardinality (relational): number of tuples in the relation.**
  - **Rank** (degree): number of attributes.
  - **Semantics:**
    - Set semantics → no duplicates
    - Bag semantics → duplicates allowed (practical use)
  - **Order irrelevant:** neither tuples nor attributes have a defined order.



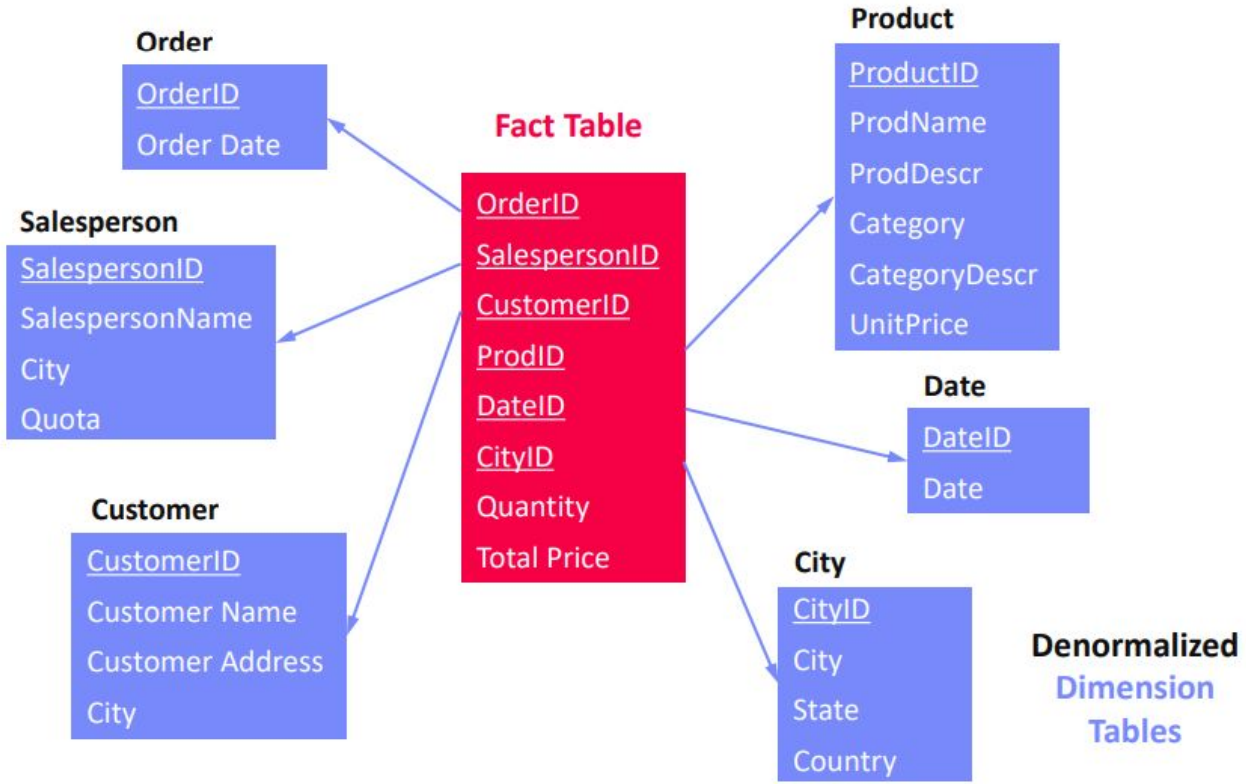
DWH: Relational Data Model (RECAP)

- **Domain (D)**: set of possible values (INT, CHAR[20], BOOL).
- **Relation (R):** subset of the Cartesian product of domains.
  - Relation Schema (RS): set of attributes $\{A_1,\ldots,A_n\}$ with domains.
- **Additional Terminology**
  - **Tuple:** row of k elements in a relation.
  - **Cardinality (relational): number of tuples in the relation.**
  - **Rank** (degree): number of attributes.
  - **Semantics:**
    - Set semantics → no duplicates
    - Bag semantics → duplicates allowed (practical use)
  - **Order irrelevant:** neither tuples nor attributes have a defined order.

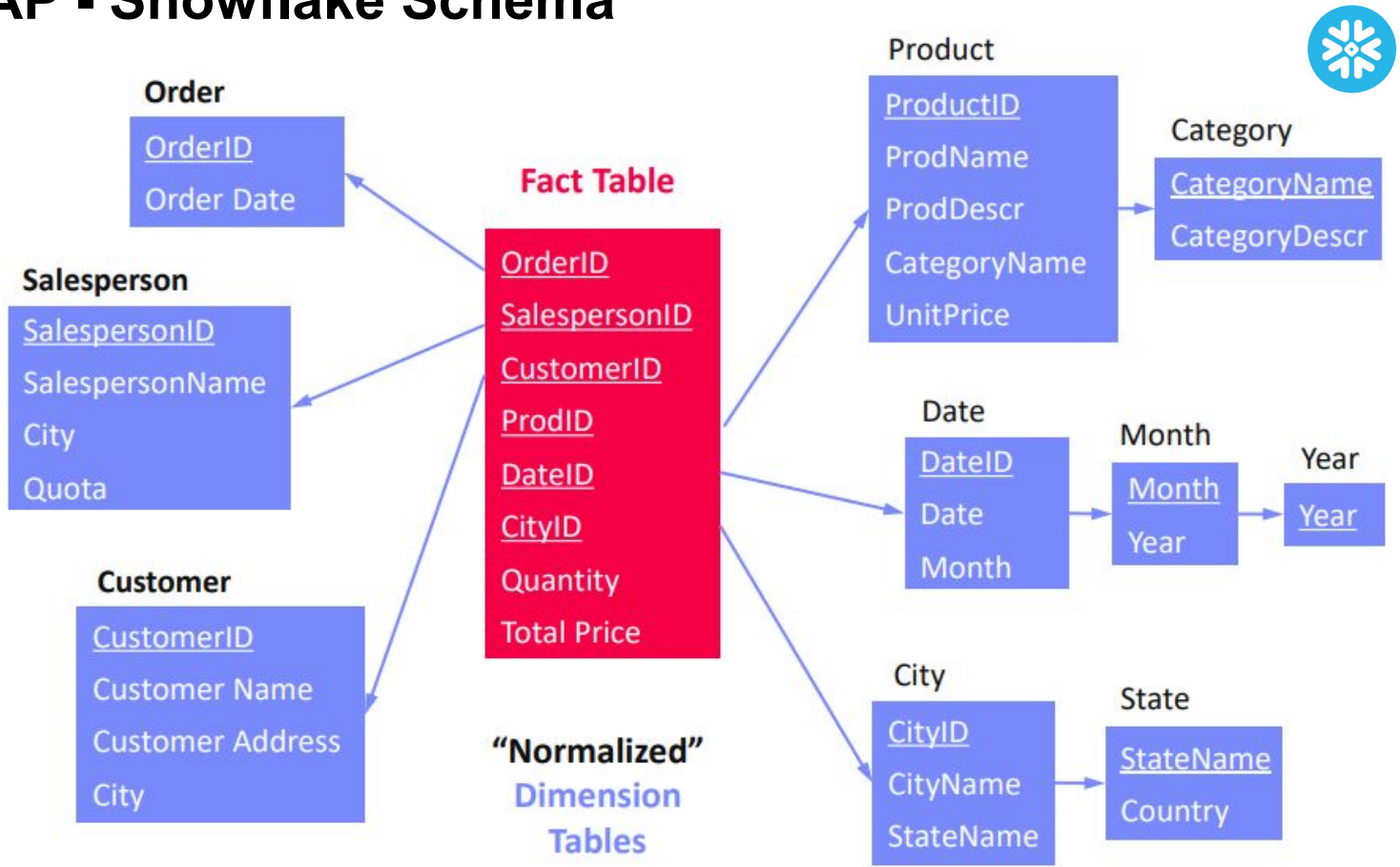


In ER Model its a relationship between entities
e.g. 1:N (one department -> many employees)

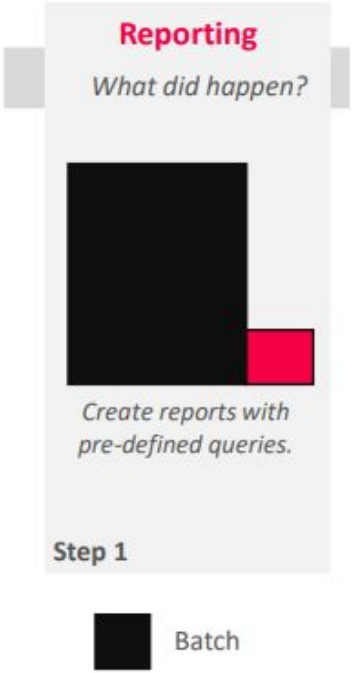
DWH: ROLAP - Star Schema



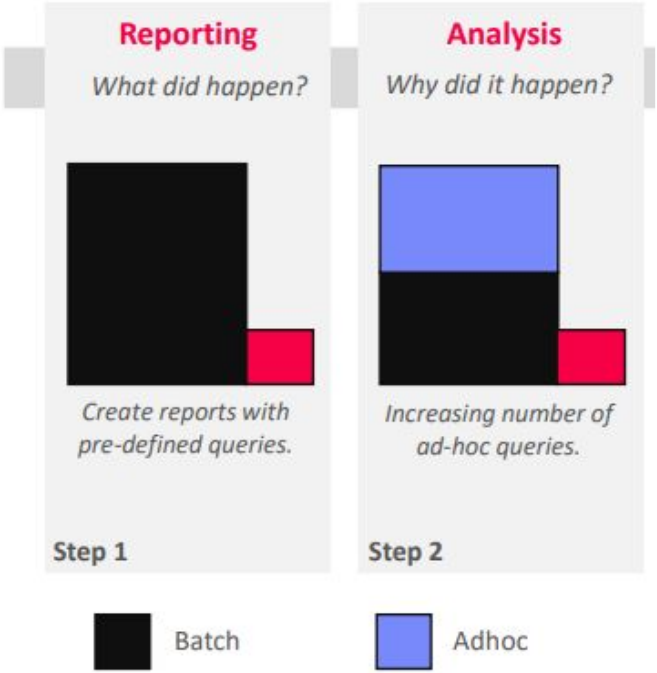
DWH: ROLAP - Snowflake Schema



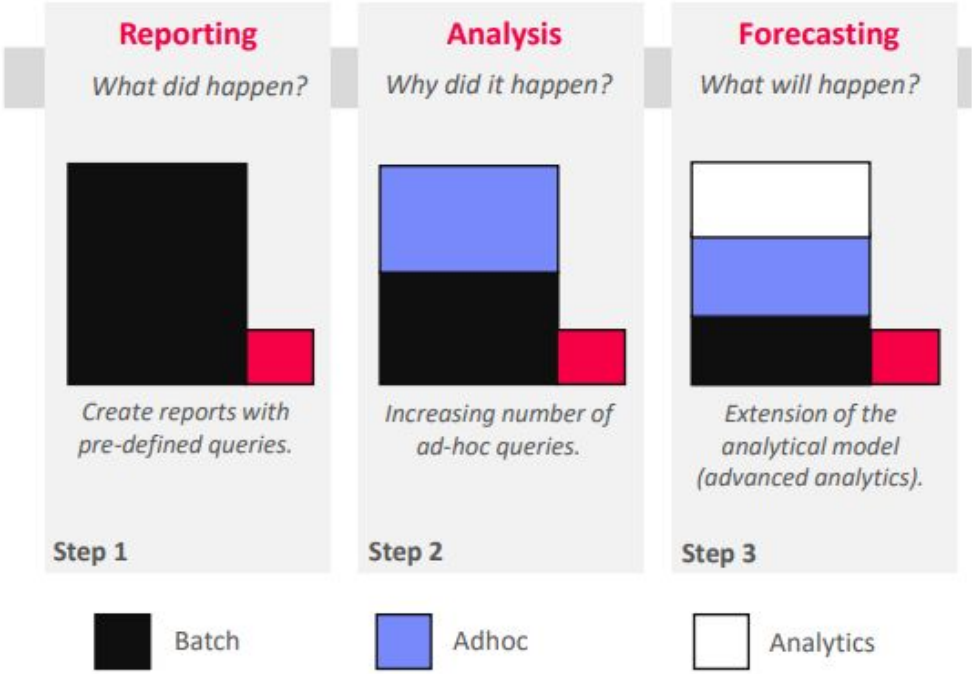
DWH: Evolution of DWH/OLAP Workloads



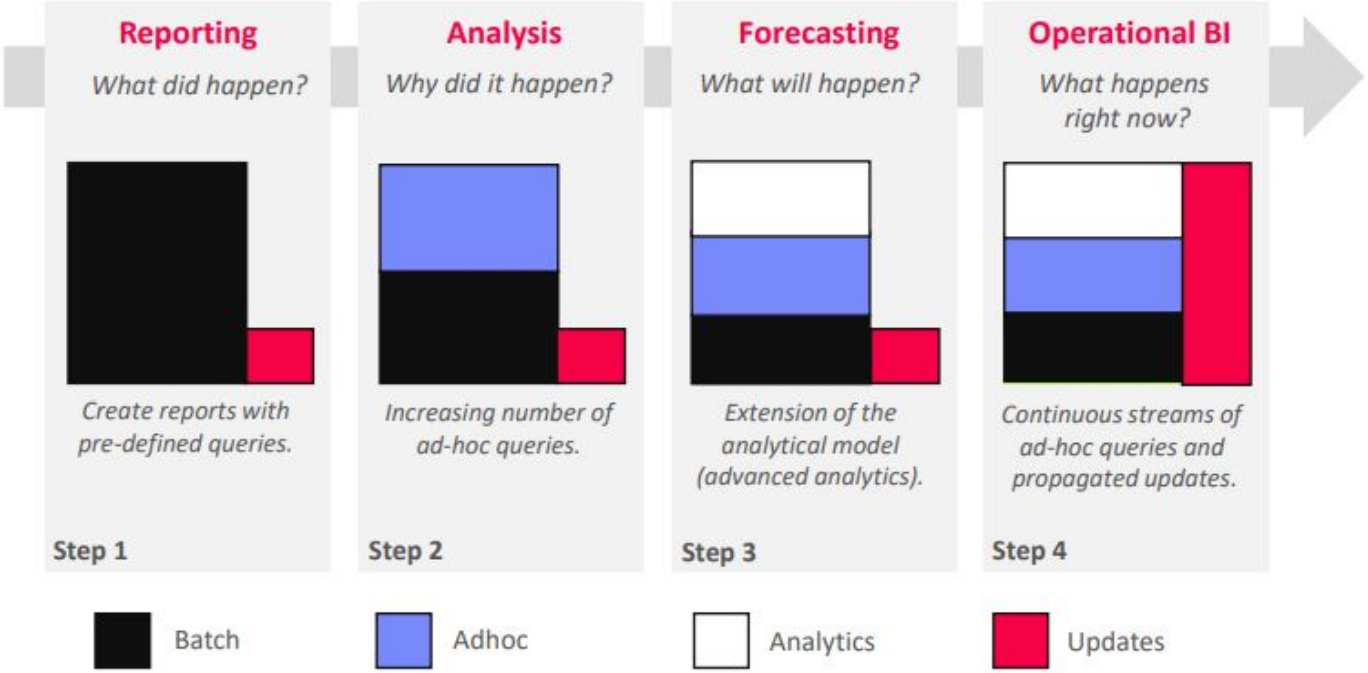
DWH: Evolution of DWH/OLAP Workloads



DWH: Evolution of DWH/OLAP Workloads

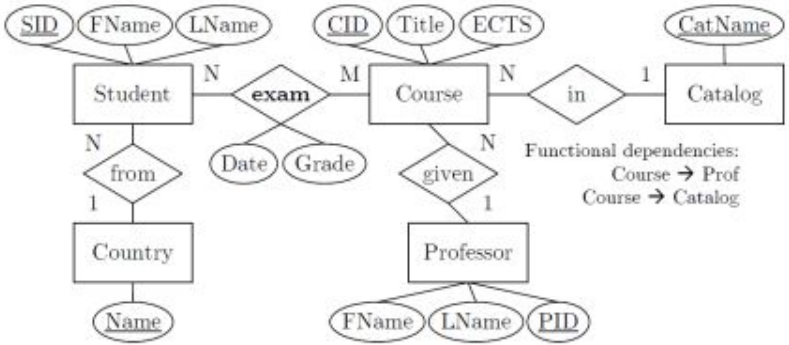


DWH: Evolution of DWH/OLAP Workloads



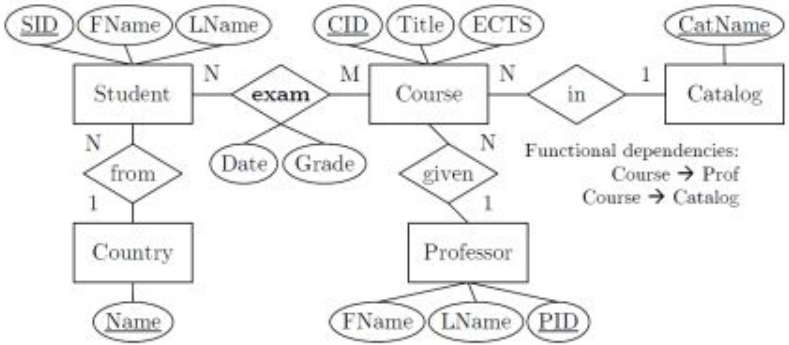
DWH: Star schema example

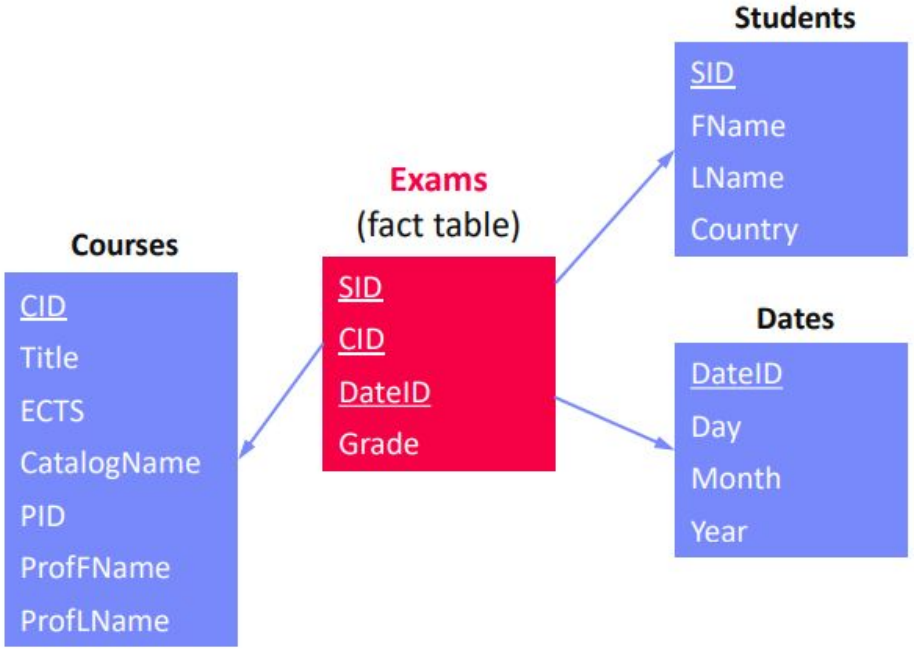
Task: Given below ER diagram, create a **ROLAP star schema.** Data types can be ignored, but indicate **PK and FK** constraints.



DWH: Star schema example

Task: Given below ER diagram, create a ROLAP star schema. Data types can be ignored, but indicate PK and FK constraints.





Extraction, Transformation, Loading (ETL)

ETL: Overview

- **Concept**
  - ETL process refers to the overall process of obtaining data from the source systems, cleaning and transforming it, and loading it into the DWH.
  - Includes many integration and cleaning techniques

ETL: Overview

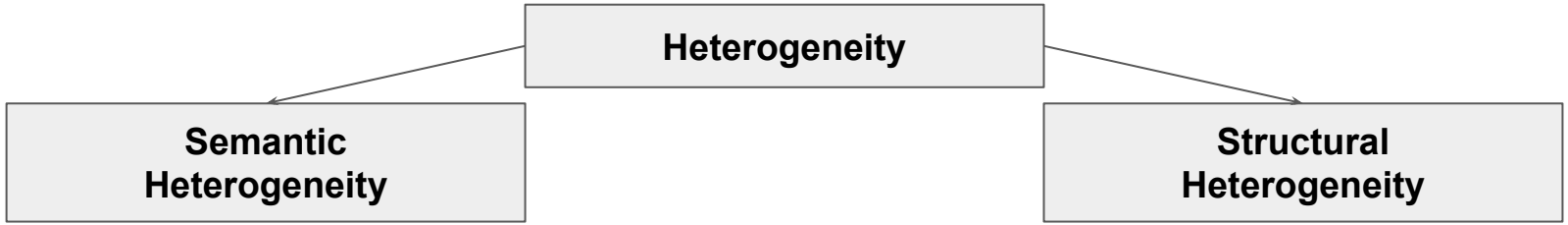
- **Concept**
  - ETL process refers to the overall process of obtaining data from the source systems, cleaning and transforming it, and loading it into the DWH.
  - Includes many integration and cleaning techniques
- **ETL**
  - **E**xtract data from **heterogeneous sources**
  - **Transform** data via dedicated data flows or in **staging area**
  - **Load** cleaned and transformed data into **DWH**

ETL: Overview

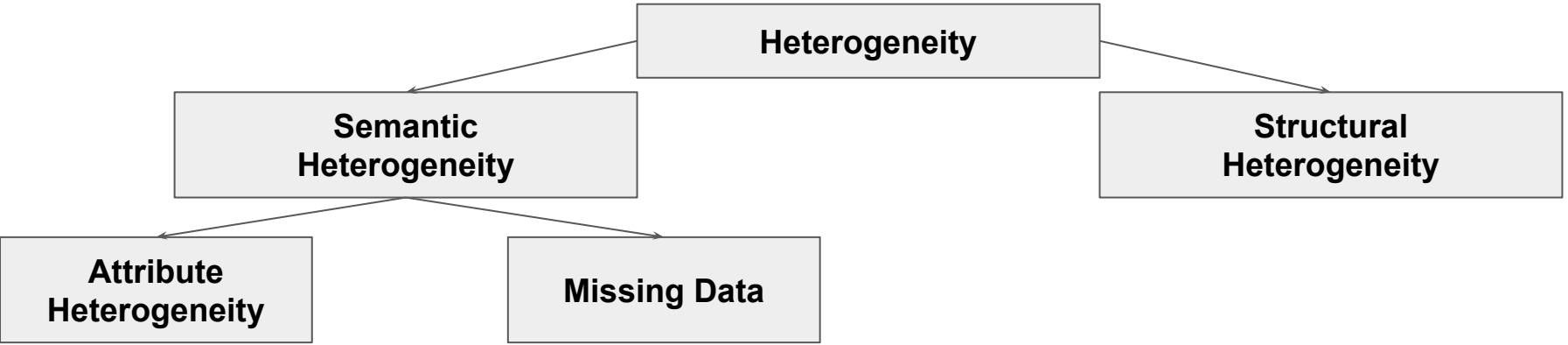
- **Concept**
  - ETL process refers to the overall process of obtaining data from the source systems, cleaning and transforming it, and loading it into the DWH.
  - Includes many integration and cleaning techniques
- **ETL**
  - Extract data from heterogeneous sources
  - **Transform** data via dedicated data flows or in staging area
  - **Load** cleaned and transformed data into DWH
- **ELT**
  - **E**xtract data from **heterogeneous sources**
  - **Load** raw data directly into DWH
  - **Transform** inside the DWH via SQL

ETL: TYPES OF HETEROGENEITY

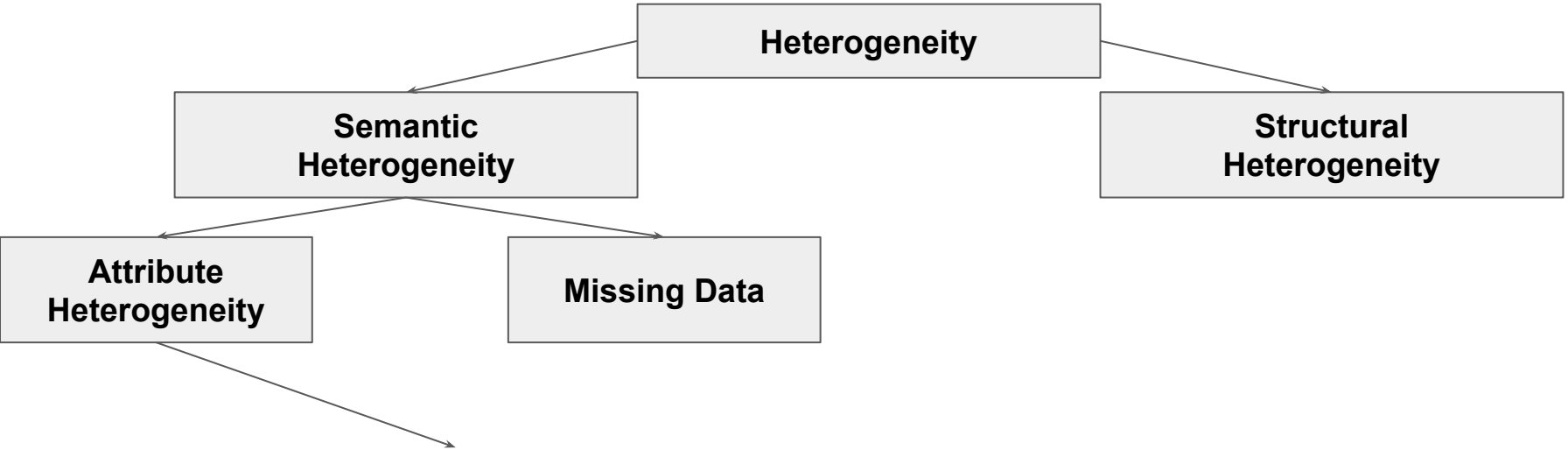
ETL: TYPES OF HETEROGENEITY



ETL: TYPES OF HETEROGENEITY

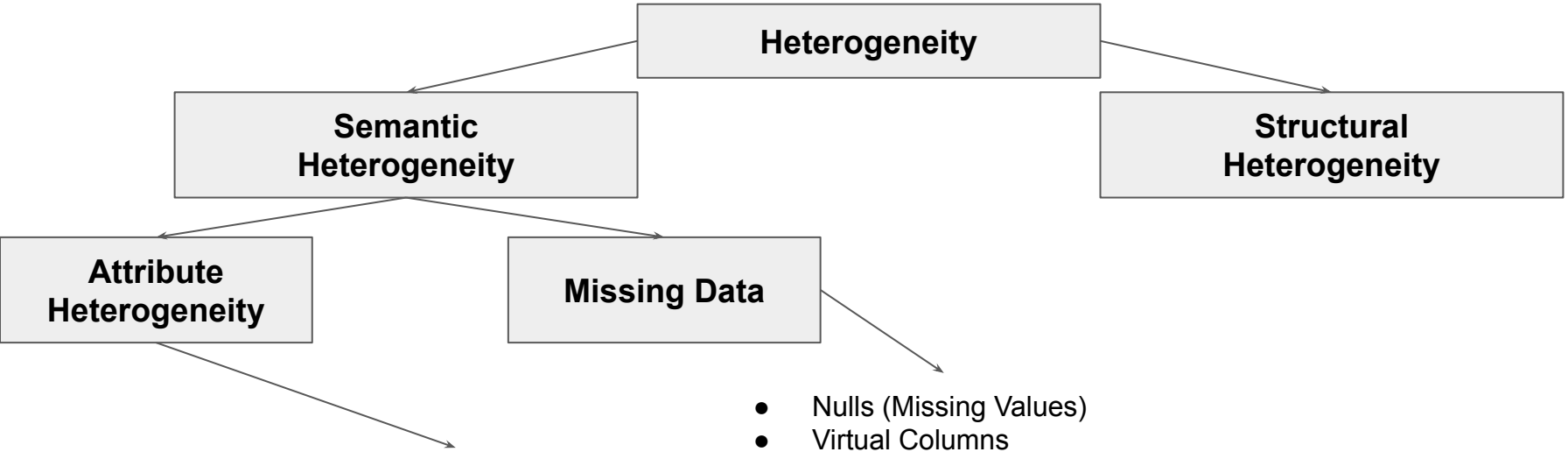


ETL: TYPES OF HETEROGENEITY



- Synonyms / Homonyms: (Instructor vs lecturer)
- Simple Mapping: (12hr vs 24 hs)
- Language Expressions: (Database vs Datenbank)

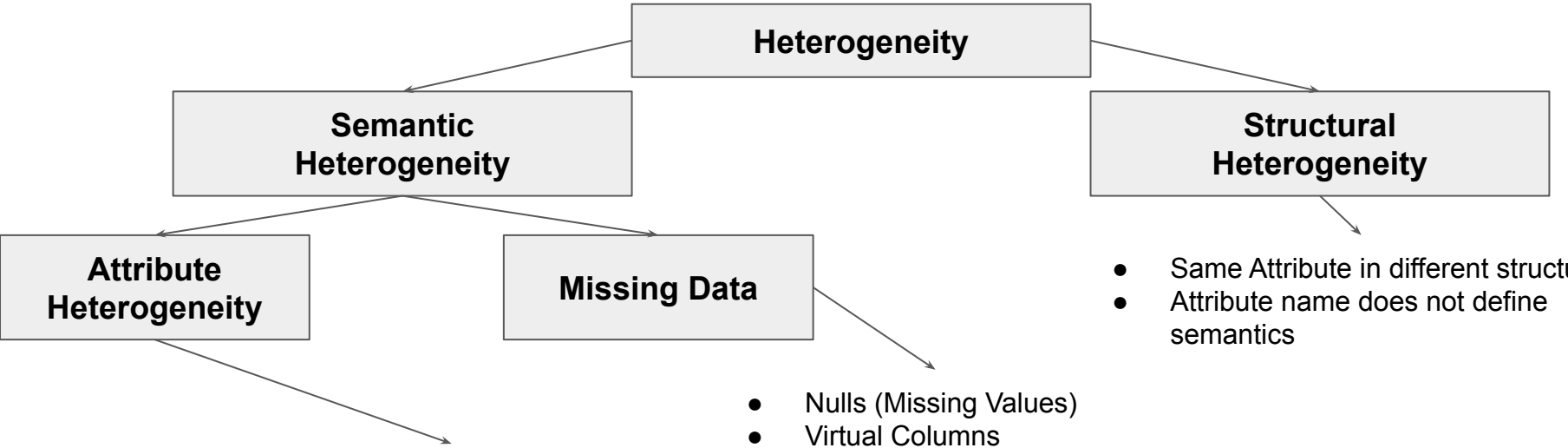
ETL: TYPES OF HETEROGENEITY



- Synonyms / Homonyms: (Instructor vs lecturer)
- Simple Mapping: (mat 12hr vs 12 hs)
- Language Expressions: (Database vs Datenbank)

- Nulls (Missing Values)
- Virtual Columns
- Semantic Incompatibility (Concept does not exist)

ETL: TYPES OF HETEROGENEITY

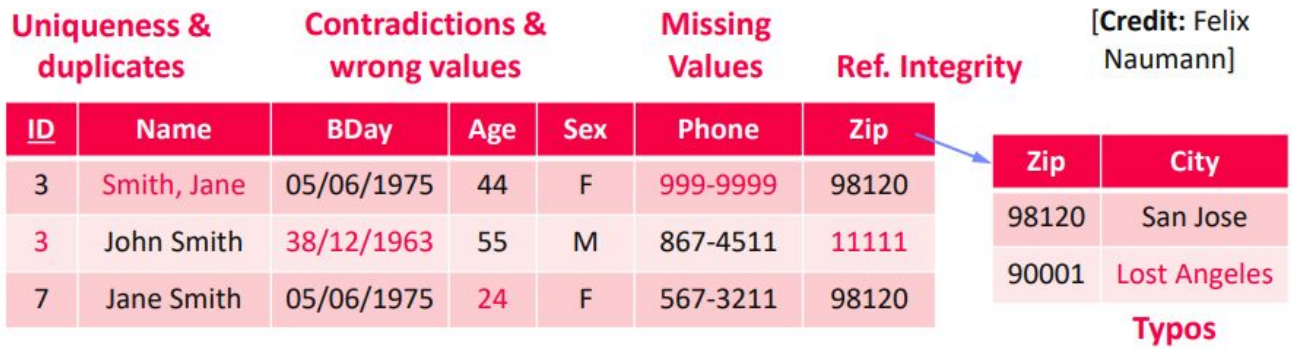


ETL: Corrupted data

- **Heterogeneity of Data Sources**
  - Data comes from many different systems **--->** Disorder.
- **Human Errors**
  - Data is **incomplete, missing, or wrongly labeled** (sometimes due to laziness or bias!).
- **Shifting Formats:**
  - The way data is saved **changes over time ----->** Inconsistency. (e.g. YYYY MM DD, DD MM YYYY)
- **Equipment Failure:**
  - Faulty hardware (like batteries or sensors)

ETL: Corrupted data

- **Heterogeneity of Data Sources**
  - Data comes from many different systems ---> Disorder.
- **Human Errors**
  - Data is incomplete, missing, or wrongly labeled (sometimes due to laziness or bias!).
- **Shifting Formats:**
  - The way data is saved changes over time ----> Inconsistency.
- **Equipment Failure:**
  - Faulty hardware (like batteries or sensors)



ETL: Planning and Design Phase

- **Architecture, Flows, and Schemas**
  - Plan requirements, architecture, tools
  - Design high-level integration flows (systems, integration jobs)
  - Data understanding (copy/code books, metadata)
  - Design dimension & loading (static, dynamic incl keys)
  - Design fact table & loading

ETL: Planning and Design Phase

- **Architecture, Flows, and Schemas**
  - Plan requirements, architecture, tools
  - Design high-level integration flows (systems, integration jobs)
  - Data understanding (copy/code books, metadata)
  - Design dimension & loading (static, dynamic incl keys)
  - Design fact table & loading
- **Data Integration and Cleaning**
  - Types of data sources (snapshot, APIs, query language, logs)
  - Prepare schema mappings → see **04 Schema Matching and Mapping**
  - Change data capture and incremental loading (diff, aggregates)
  - Transformations, enrichments, and deduplication → **05 Entity Linking**
  - Data validation and cleansing → see **06 Data Cleaning and Data Fusion**

ETL: Events and Change Data Capture

- **Architecture, Flows, and Schemas**
  - Plan requirements, architecture, tools
  - Design high-level integration flows (systems, integration jobs)
  - Data understanding (copy/code books, metadata)
  - Design dimension loading (static, dynamic incl keys)
  - Design fact table loading
- **Data Integration and Cleaning**
  - Types of data sources (snapshot, APIs, query language, logs)
  - Prepare schema mappings → see **04 Schema Matching and Mapping**
  - Change data capture and incremental loading (diff, aggregates)
  - Transformations, enrichments, and deduplication → **05 Entity Linking**
  - Data validation and cleansing → see **06 Data Cleaning and Data Fusion**
- **Optimization**
  - Partitioning schemes for loaded data (e.g., per month)
  - Maintenance

ETL: Events and Change Data Capture

Goal: Monitoring operations of data sources for detecting changes

- **Explicit Messages/Triggers**
  - Setup update propagation from the source systems to middleware (e.g. every night at 23:00)
  - **Asynchronously** propagate the updates into the DWH

ETL: Events and Change Data Capture

Goal: Monitoring operations of data sources for detecting changes

- **Explicit Messages/Triggers**
  - Setup update propagation from the source systems to middleware (e.g. every night at 23:00)
  - **Asynchronously** propagate the updates into the DWH
- **Log-based Capture**
  - Parse system logs / provenance to **retrieve changes since last loading**
  - Leverage explicit audit columns or internal timestamps

ETL: Events and Change Data Capture

Goal: Monitoring operations of data sources for detecting changes

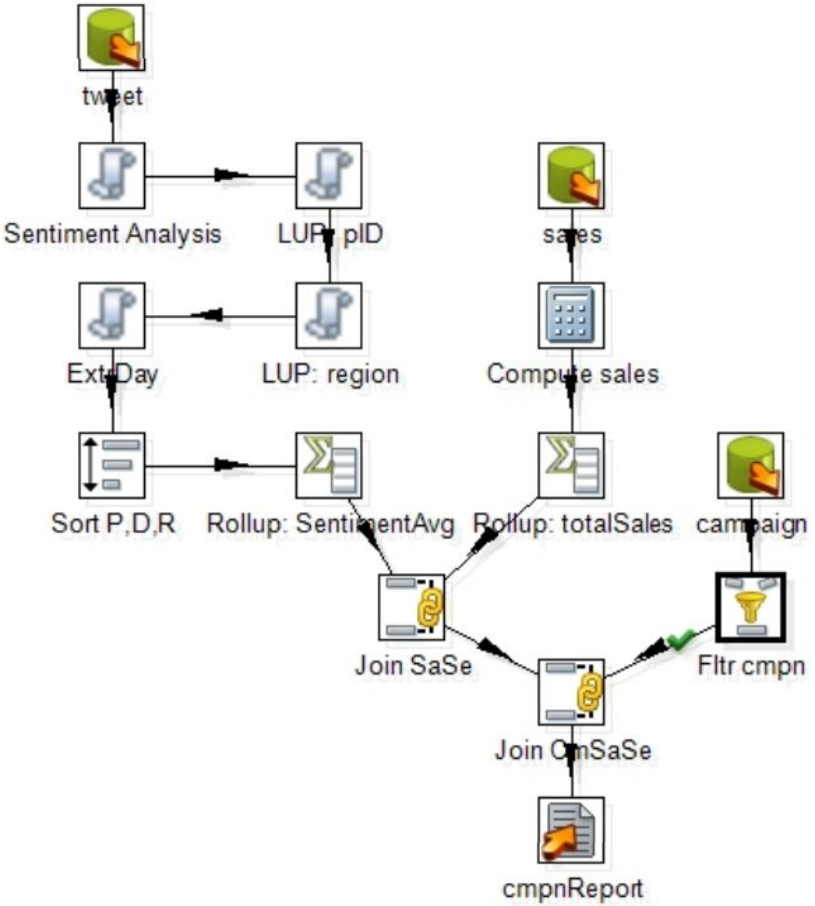
- **Explicit Messages/Triggers**
  - Setup update propagation from the source systems to middleware
  - Asynchronously propagate the updates into the DWH
- **Log-based Capture**
  - Parse system logs / provenance to retrieve changes since last loading
  - Sometimes combined w/ replication ➔ **03 MoM, EAI, and Replication**
  - Leverage explicit audit columns or internal timestamps
- **Snapshot Differences**
  - Compute **difference between old and new snapshot** (e.g., files) before loading
  - Broadly applicable but **more expensive**

ETL: Example Flow

xPAD: A Platform for Analytic Data Flows

Alkis Simitsis, Kevin Wilkinson, Petar Jovanovic

Simitsis et.al. XPAD: a platform for analytic data flows. In Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data (SIGMOD '13). Association for Computing Machinery,



SQL/OLAP Extensions

SQL/OLAP Extensions: Multi-Groupings

- **Recap: GROUP BY**
  - Group tuples by categorical variables (e.g. year)
  - Aggregate per group (SUM of Revenue)

Year	Quarter	Revenue
2004	1	10
2004	2	20
2004	3	10
2004	4	20
2005	1	30

```
SELECT Year, SUM(Revenue)
  FROM Sales
  GROUP BY Year
```

Year	SUM
2004	60
2005	30

SQL/OLAP Extensions: Multi-Groupings

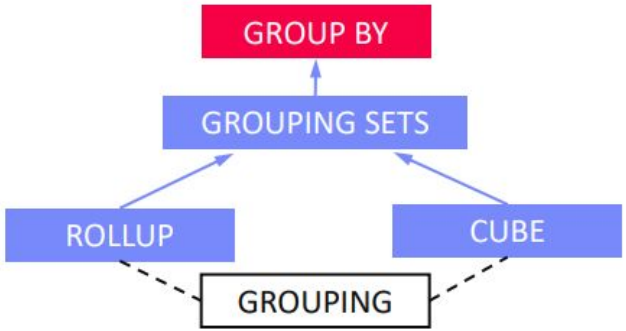
- **Recap: GROUP BY**
  - Group tuples by categorical variables
  - Aggregate per group

Year	Quarter	Revenue
2004	1	10
2004	2	20
2004	3	10
2004	4	20
2005	1	30

```
SELECT Year, SUM(Revenue)
  FROM Sales
  GROUP BY Year
```

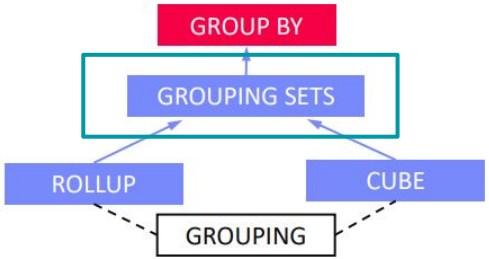
Year	SUM
2004	60
2005	30

- **Grouping Extensions**



SQL/OLAP Extensions: Multi-Groupings - Grouping Sets

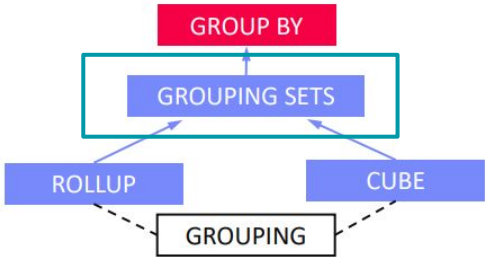
- **Semantics**
  - **What It Does:** It allows you to generate aggregated results (sums, averages, etc.) for multiple, specific combinations of columns within a single query.
  - **Why Use It?** It's a great shortcut that often offers better performance than writing many separate GROUP BY clauses and manually joining them with UNION ALL.
  - **In Short:** You get multiple reports (views) with just one command.
- **Example**



SQL/OLAP Extensions: Multi-Groupings - Grouping Sets

- **Example**

```
SELECT Year, Quarter, SUM(Revenue)
  FROM R
  GROUP BY GROUPING SETS
    ((), (Year), (Year,Quarter))
```



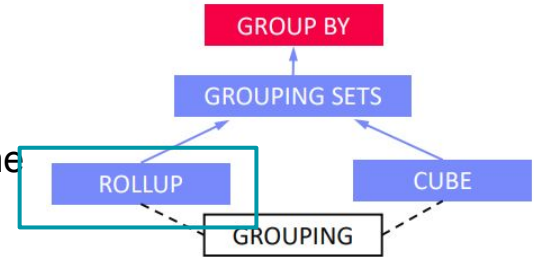
Year	Quarter	Revenue
2004	1	10
2004	2	20
2004	3	10
2004	4	20
2005	1	30

Granularity

Year	Quarter	SUM	
-	-	90	()
2004	-	60	(Year)
2005	-	30	
2004	1	10	(Y, Q)
2004	2	20	
2004	3	10	
2004	4	20	
2005	1	30	

SQL/OLAP Extensions: Multi-Groupings - Rollup

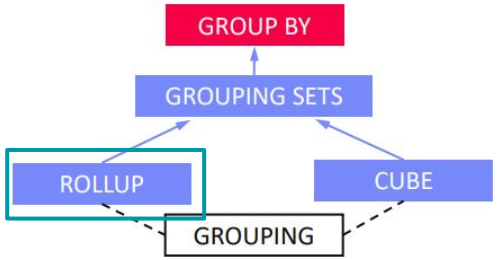
- **Semantics**
  - **What it does:** It's a command that generates sequential subtotals across a hierarchy you define, culminating in a **Grand Total** of all the data.
  - **Why use It?** Perfect for reports where you need to see sales by country, then by country and city, and finally the worldwide total.
  - **In short:** The ROLLUP syntax is just a much cleaner way to write out a long and repetitive GROUPING SETS command.



SQL/OLAP Extensions: Multi-Groupings - Rollup

- **Example**

```
SELECT Year, Quarter, SUM(Revenue)
  FROM R
  GROUP BY ROLLUP(Year,Quarter)
```



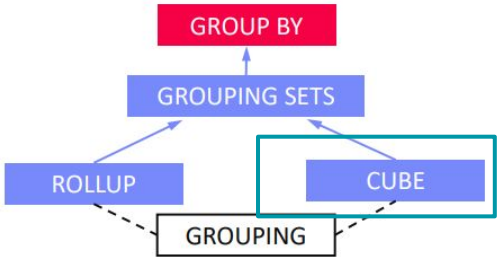
Year	Quarter	Revenue
2004	1	10
2004	2	20
2004	3	10
2004	4	20
2005	1	30

Granularity

Year	Quarter	SUM	
-	-	90	()
2004	-	60	(Year)
2005	-	30	
2004	1	10	(Y, Q)
2004	2	20	
2004	3	10	
2004	4	20	
2005	1	30	

SQL/OLAP Extensions: Multi-Groupings - Cube

- **Semantics**
  - **What it does:** It creates all possible subtotals from the columns you define, including the **grand total**. It gives you **every single combination of the data.**
  - **Why use It?** It's essential for Multi-Dimensional Analysis (OLAP) where users might need to **slice the data by any combination of fields.**
  - **In short:** Like ROLLUP, the CUBE syntax is a concise, high-performance way to write out an enormous, exhaustive GROUPING SETS command.

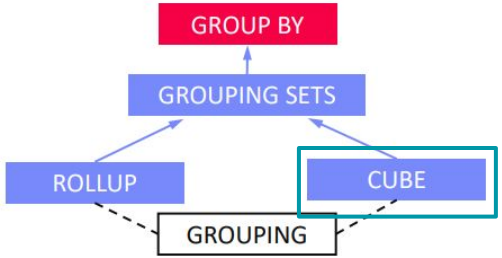


SQL/OLAP Extensions: Multi-Groupings - Cube

- **Example**

```
SELECT Year, Quarter, SUM(Revenue)
  FROM R
  GROUP BY CUBE(Year,Quarter)
```

= GROUP BY ((), (Y), (Q), (Y,Q))



Year	Quarter	Revenue
2004	1	10
2004	2	20
2004	3	10
2004	4	20
2005	1	30

Year	Quarter	SUM
-	-	90
2004	-	60
2005	-	30
-	1	40
-	2	20
-	3	10
-	4	20
2004	1	10
2004	2	20
2004	3	10
2004	4	20
2005	1	30

Summary and Q&A

Summary and Q&A

- **Data Warehousing (DWH)**
  - DWH architecture
  - Multidimensional modeling
- **Extraction, Transformation, Loading (ETL)**
  - ETL process, errors, (and data flows → )
- **SQL/OLAP Extensions**
  - Multi-grouping operations
- **Next lecture: Data Integration Architectures**
  - October 17. Message-oriented Middleware, EAI, and Replication + **Project Presentation**

Many thanks!