Multidimensional Databases

Conceptual Design

Peter Scheuermann

ER Model vs. Multidimensional Model

- Why don't we use the ER model in data warehousing?

- ER model: a data model for **general** purposes
  - All types of data are "equal", difficult to identify the data that is important for business analysis
    - No difference between:
      - What **is** important
      - What just **describes** the important
    - Normalized databases **spread** information
    - When analyzing data, the information must be **integrated** again
  - Hard to overview a **large** ER diagram (e.g., over 100 entities/relations for an enterprise)

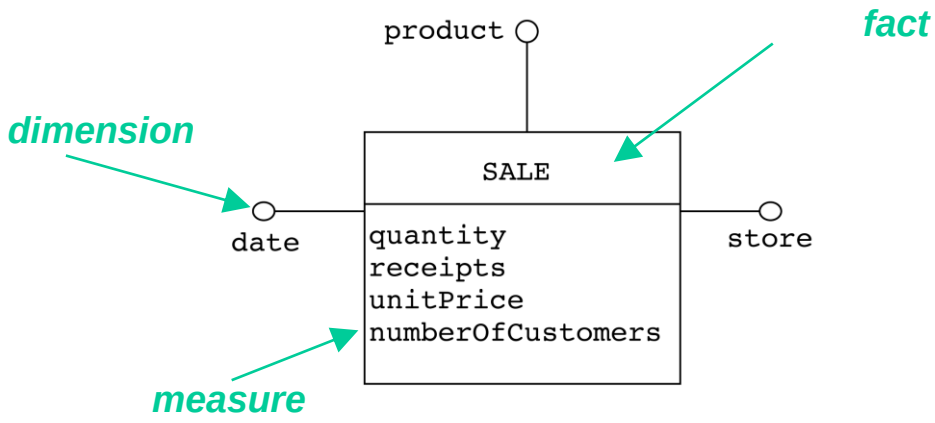
ER Model vs. Multidimensional Model

- The multidimensional model
  - Its only purpose: **data analysis**
    - It is not suitable for OLTP systems
  - More **built in** “meaning”
    - What **is** important
    - What **describes** the important
    - What we want to **optimize**
    - Easy for query operations
- Recognized by OLAP/BI tools
  - Tools offer powerful query facilities based on MD design

Multidimensional model: basic concepts

- A *fact* is a concept relevant to decision-making processes. It typically models a set of events taking place within a company (e.g., sales, shipments, purchases, ...). It is essential that a fact have dynamic properties or evolve in some way over time
- A *measure* is a numerical property of a fact and describes a quantitative fact aspect that is relevant to analysis (e.g., every sale is quantified by its receipts)
- A *dimension* is a fact property with a finite domain and describes an analysis coordinate of the fact. Typical dimensions for the sales fact are products, stores, and dates.
- *Facts* "live" in a multidimensional cube

A fact expresses a many-to-many relationship between its dimensions



Cubes

- A “cube” may have **many** dimensions!
  - More than 3 - the term ”hypercube” is sometimes used
  - Theoretically no limit for the number of dimensions
  - Typical cubes have 4-12 dimensions
- But only 2-4 dimensions can be viewed at a time
  - Dimensionality reduced by queries via projection/aggregation
- A cube consists of **cells**
  - A given combination of dimension values
  - A cell can be empty (no data for this combination)
  - A **sparse** cube has few non-empty cells
  - A **dense** cube has many non-empty cells
  - Cubes become sparser for many/large dimensions

Dimensions

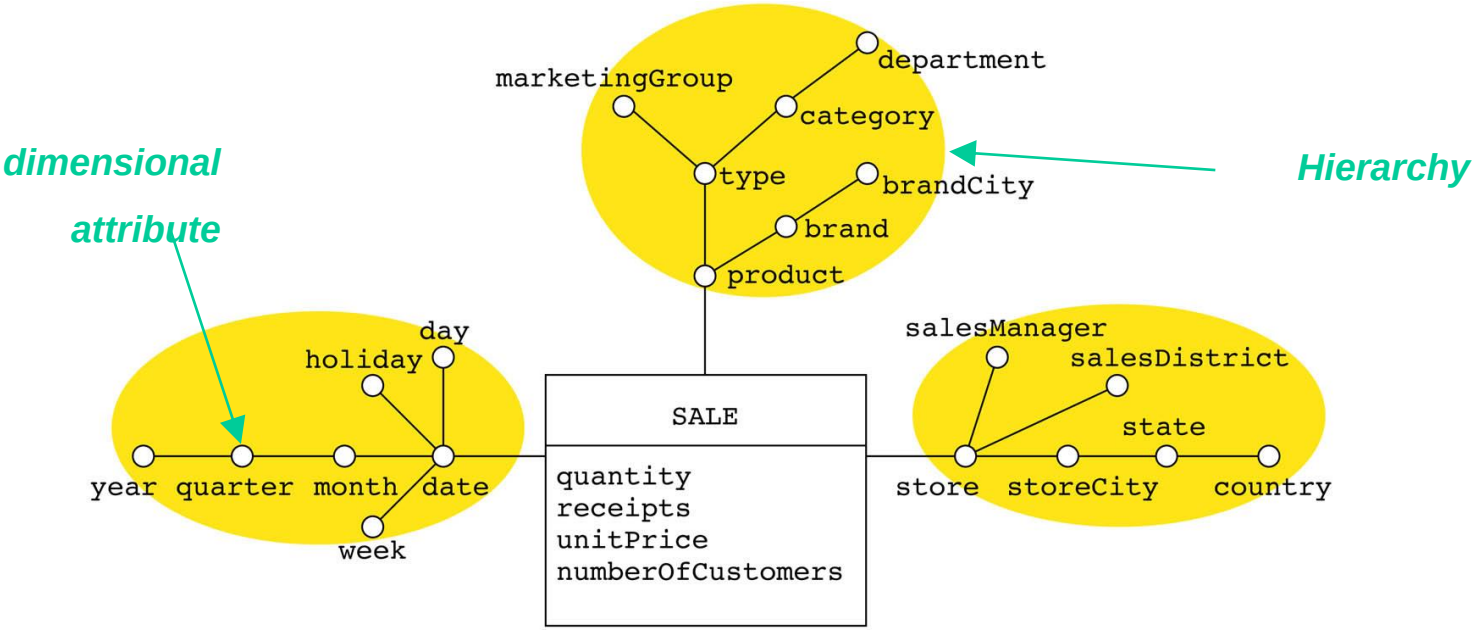
- Dimensions are the core of multidimensional databases
  - Other types of databases do not support dimensions
- Dimensions are used for
  - **Selection** of data
  - **Grouping** of data at the right level of detail
- Dimensions consist of **dimension values**
  - Product dimension have values ”milk”, ”cream”, …
  - Time dimension have values ”1/1/2001”, ”2/1/2001”,…
- Dimension values may have an **ordering**
  - Used for comparing cube data across values
  - Example: ”percent sales increase compared with last month”
  - Especially used for Time dimension

Dimensions -continued

- Dimensions have **hierarchies** with **levels**
  - Typically 3-5 levels (of detail)
  - Dimension values are organized in a **tree structure**
  - **Product**: Product->Type->Category
  - **Store**: Store->Area->City->County
  - **Time**: Day->Month->Quarter->Year
  - Dimensions have a **bottom level** and a **top level** (ALL)
- Levels may have **attributes**
  - Simple, non-hierarchical information
  - Day has Workday as attribute
- Dimensions should contain much information
  - Time dimensions may contain holiday, season, events,…
  - Good dimensions have 50-100 or more attributes/levels

MD: basic concepts

- The general term *dimensional attributes* stands for the dimensions and other possible attributes, always with discrete values, that describe them (e.g., a product is described by its type, by the category to which it belongs, by its brand, and by the department in which it is sold)
- A *hierarchy* is a directed tree whose nodes are dimensional attributes and whose arcs model *many-to-one associations* between dimensional attribute pairs. It includes a dimension, positioned at the tree's root, and all of the dimensional attributes that describe it

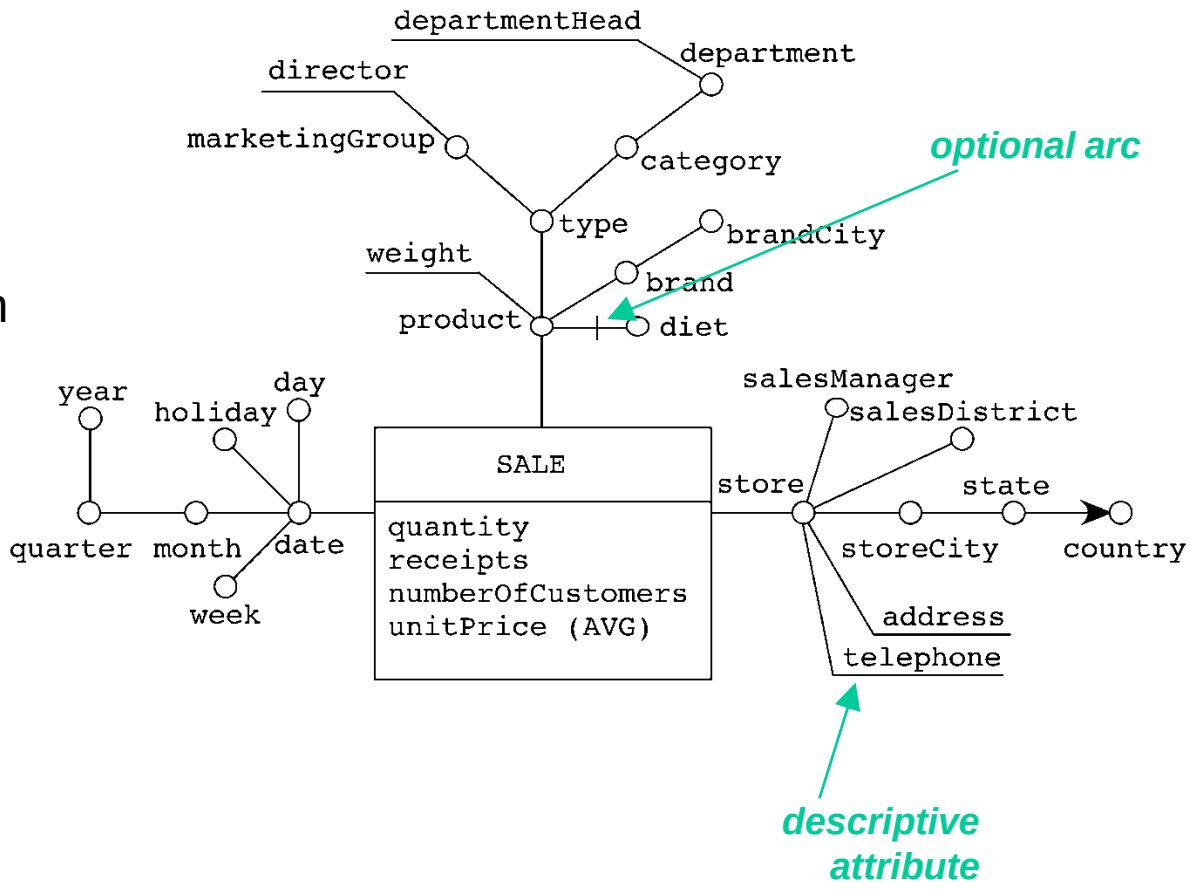


Events and aggregation

- A *primary event* is a particular occurrence of a fact, identified by one tuple made up of a value for each dimension. A value for each measure is associated with each primary event
  - In reference to the sales example, a possible primary event records that 10 packages of Shiny detergent were sold for total sales of $25 on 10/10/2008 in the SmartMart store
- Given a set of dimensional attributes (**group-by set**), each n-tuple of their values identifies a *secondary event* that aggregates all of the corresponding primary events. Each secondary event is associated with a value for each measure that sums up all the values of the same measure in the corresponding primary events
  - This makes it possible to use hierarchies to define the way you can aggregate primary events and effectively select them for decision-making processes. While the dimension in which a hierarchy takes root defines its finest aggregation granularity, the other dimensional attributes correspond to a gradually increasing granularity

DFM: advanced concepts

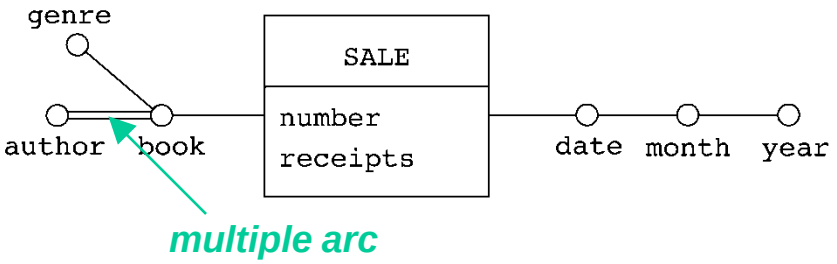
- A *descriptive attribute* stores additional information about a dimensional attribute. It is not used for aggregation because it has a dense domain and/or it is a child of a one-to-one association
- Some arcs in a fact schema can be *optional*



DFM (dimensional fact model)

Advanced concepts

- A *multiple arc* models a many-to-many association between two dimensional attributes

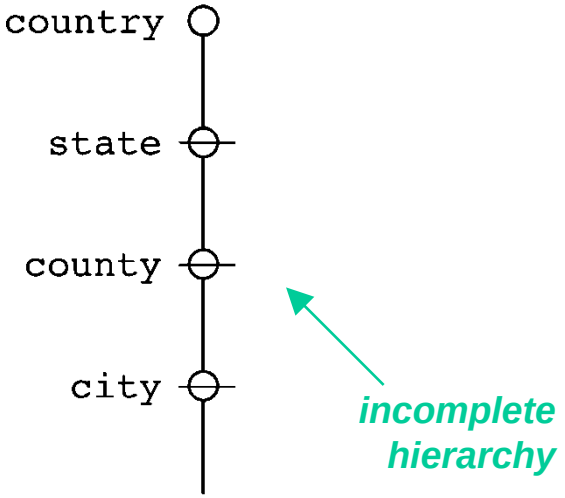


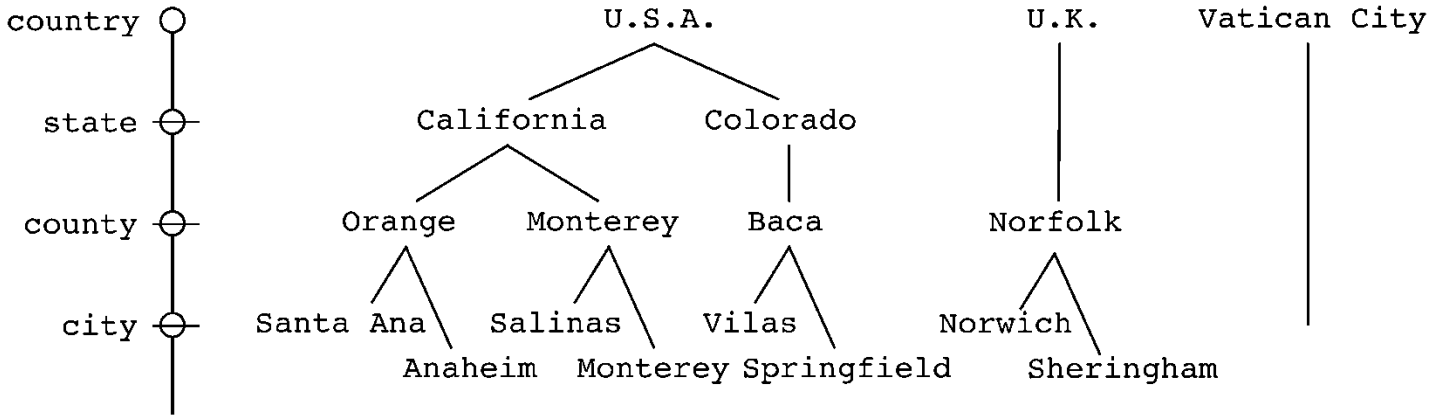
Facts & Crimes	Golfarelli, Rizz	3
Sounds Logical	Golfarelli	5
The Right Measure	Rizzi	10
Facts: How and Why	Golfarelli, Rizz	4
The Fourth Dimension	Golfarelli	8

How much did Rizzi sell?

DFM: advanced concepts

- An *incomplete hierarchy* is a hierarchy where, for some instances, one ore more aggregation levels are missing (because they are unknown or undefined)



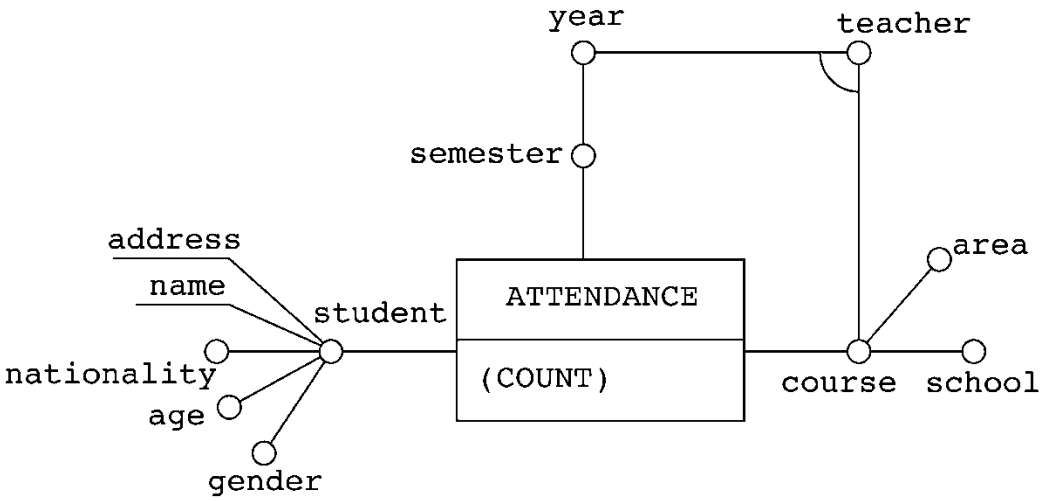


Types of Facts

- **Event** fact (transaction)
  - A fact for every **business event** (sale)
- ”**Fact-less**” facts
  - A fact per event (customer contact)
  - **No** numerical measures
  - An event has happened for a given dimension value combination
- **Snapshot** fact
  - A fact for every dimension combination at given time intervals
  - Captures **current** status (inventory)
- **Cumulative snapshot** facts
  - A fact for every dimension combination at given time intervals
  - Captures **cumulative** status up to now (sales in year to date)
- Every type of facts answers **different** questions
  - Often both event facts and both kinds of snapshot facts exist

Empty fact (Fact-less) schema

- A fact schema is said to be *empty* if it does not have any measures
  - primary events only record the *occurrence* of events in an application domain



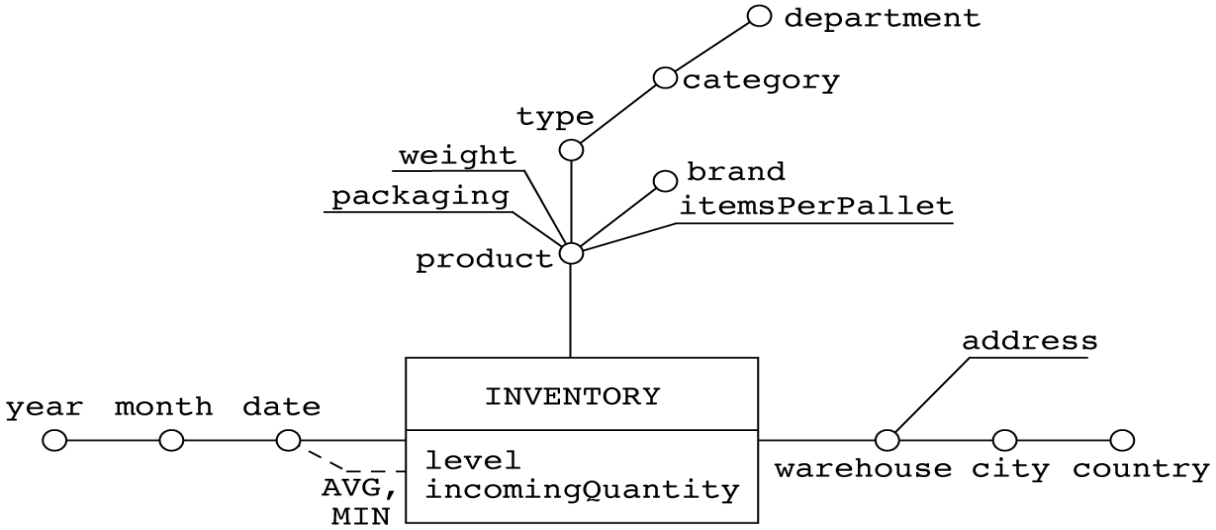
Types of Measures

- Three types of measures

- Additive
  - Can be aggregated over **all** dimensions (using SUM operator)
  - Example: **sales price**
  - Often occur in event facts
- Semi-additive
  - **Cannot** be aggregated over **some** dimensions - typically time
  - Example: **inventory level**
  - Often occur in snapshot facts
- Non-additive
  - **Cannot** be aggregated over **any** dimensions
  - Example: **average sales price**
  - Occur in all types of facts

Additivity

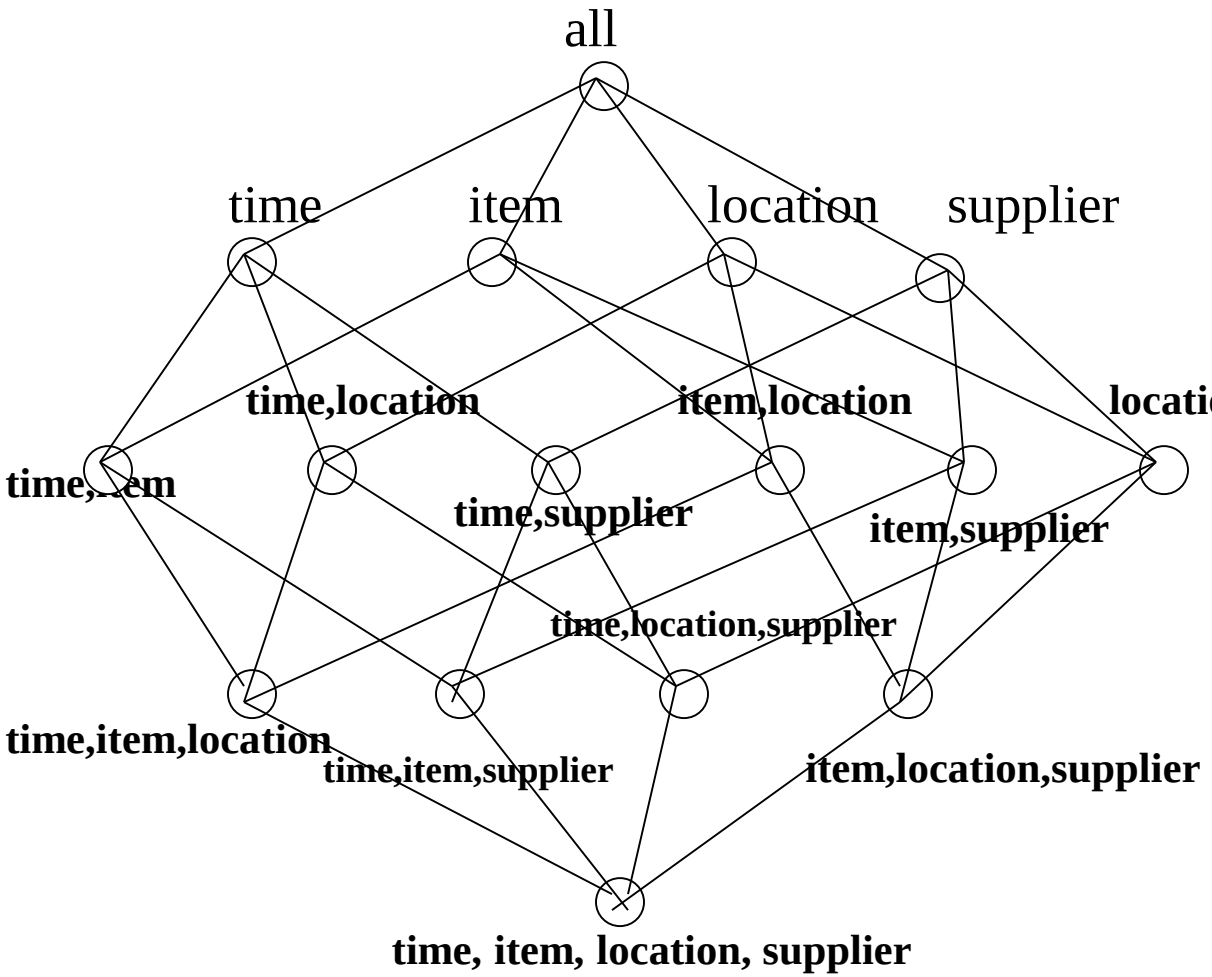
- Additivity in fact requires that we can use the SUM operator to aggregate the values along the dimension hierarchy.
- Note: a measure can be semi-additive, but you can still use the AVG, MAX, Min operators to aggregate it.



Why a schema cannot answer question X

- Possible reasons
  - Certain **measures** not included in fact table
  - **Granularity** of facts too coarse
  - Particular **dimensions** not in DW
  - Descriptive **attributes** missing from dimensions
  - **Meaning** of attributes/measures deviate from the expectation of data analysts (users)

Cube: A Lattice of Cuboids



0-D (*apex*) cuboid

1-D cuboids

2-D cuboids

3-D cuboids

4-D (*base*) cuboid

Star vs. snowflake

- Snowflaking may be useful when:
  - The ratio between the cardinalities of the primary and secondary DTs is high, because in this case it leads to a relevant space savings