

Diplomatura en Big Data

Data Warehousing y OLAP

Alejandro VAISMAN

Departamento de Ingeniería Informática

Instituto Tecnológico de Buenos Aires

avaisman@itba.edu.ar

Conceptual Data Warehouse Design

Outline

- ◆ MultiDim: A Conceptual Model for Data Warehouses
- ◆ Dimension Hierarchies
- ◆ Advanced Modeling Aspects

Conceptual Multidimensional Models

◆ Conceptual models

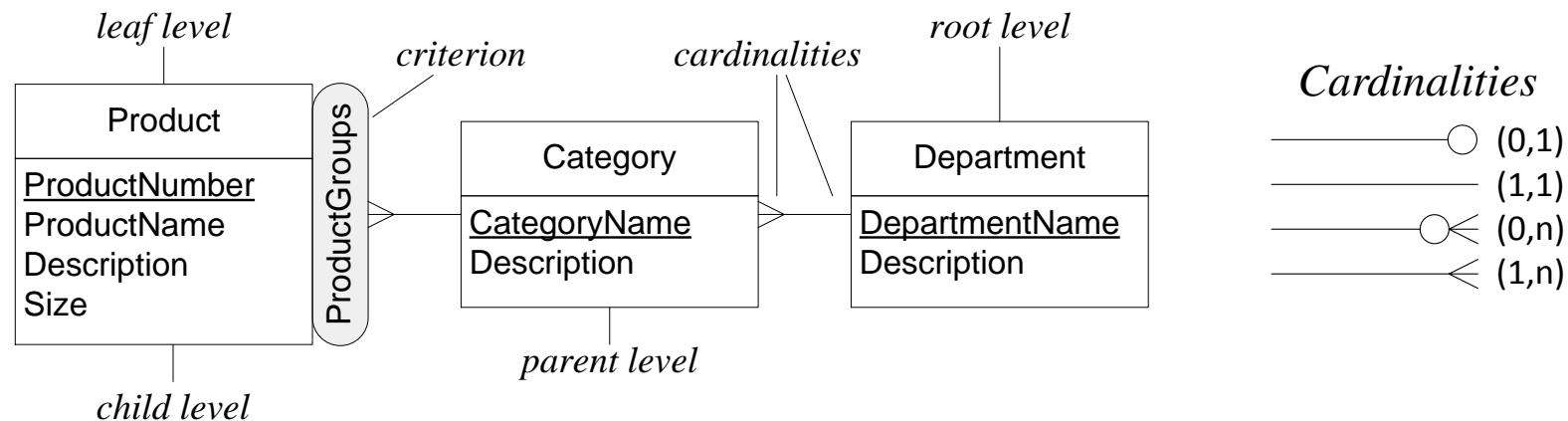
- Allow better communication between designers and users to understand application requirements
 - More stable than implementation-oriented (logical) schema, which changes with the platform
 - Provide better support for visual user interfaces
- ◆ No well-established conceptual model for multidimensional data
- ◆ Several proposals based on UML, on the ER model, or using specific notations
- ◆ Problems:
- Cannot express complex kinds of hierarchies
 - Lack of a mapping to the implementation platform
- ◆ Currently, data warehouses are designed using mostly logical models (star and snowflake schemas)
- Difficult to express requirements (technical knowledge required)
 - Limit users to defining only elements that the underlying implementation systems can manage
 - Example: Users constrained to use only the simple hierarchies supported in current tools

MultiDim: A Conceptual Multidimensional Model

- ◆ Based on the entity-relationship model
- ◆ Includes concepts like:
 - **dimensions**
 - **hierarchies**
 - **facts**
 - **measures**
- ◆ Supports various kinds of hierarchies existing in real-world applications
- ◆ Can be mapped to star or snowflake relational structures

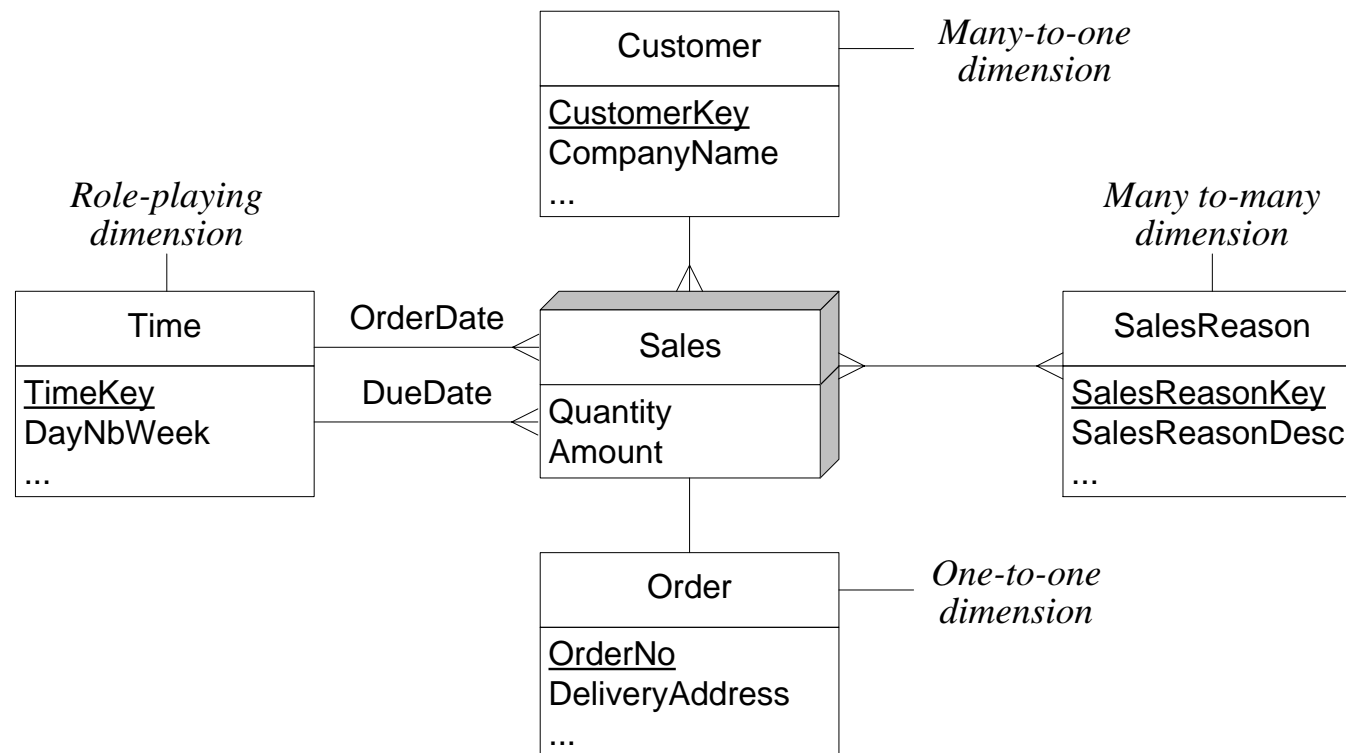
MultiDim Model: Notation

- ◆ **Dimension**: level or one or more hierarchies
- ◆ **Hierarchy**: several related levels
- ◆ **Level**: entity type
- ◆ **Member**: every instance of a level
- ◆ **Child** and **parent** levels: the lower and higher levels
- ◆ **Leaf** and **root** levels: first and last levels in a hierarchy
- ◆ **Cardinality**: Minimum/maximum numbers of members in a level related to members in another level
- ◆ **Criterion**: Expresses different hierarchical structures used for analysis
- ◆ **Key attribute**: Indicates how child members are grouped
- ◆ **Descriptive attributes**: Describe characteristics of members

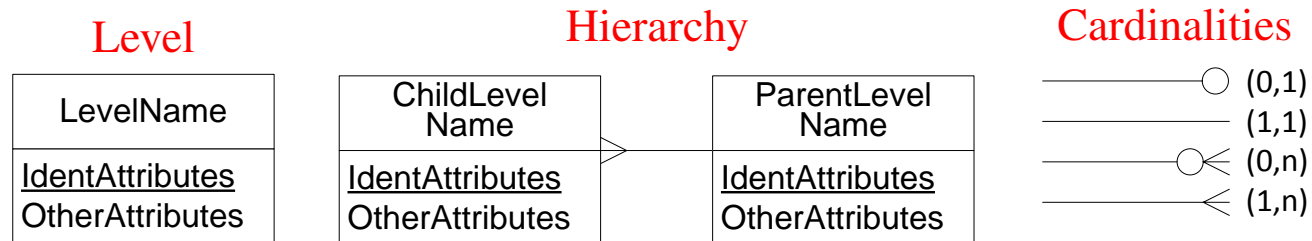


MultiDim Model: Notation

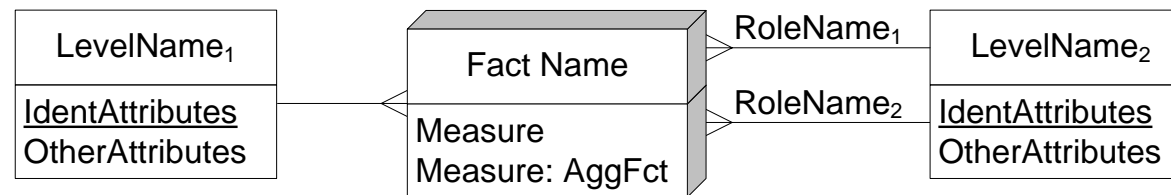
- ◆ **Fact**: Relates measures to leaf levels in dimensions
- ◆ Dimensions can be related to fact with **one-to-one**, **one-to-many**, or **many-to-many**
- ◆ Dimension can be related several times to a fact with **different roles**



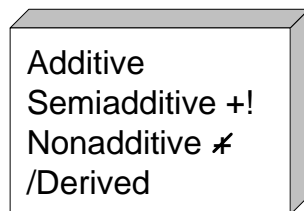
MultiDim Model: Notation (Summary)



Fact with measures and associated levels



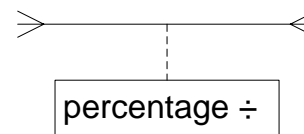
Types of measures



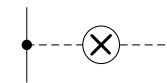
Analysis criterion



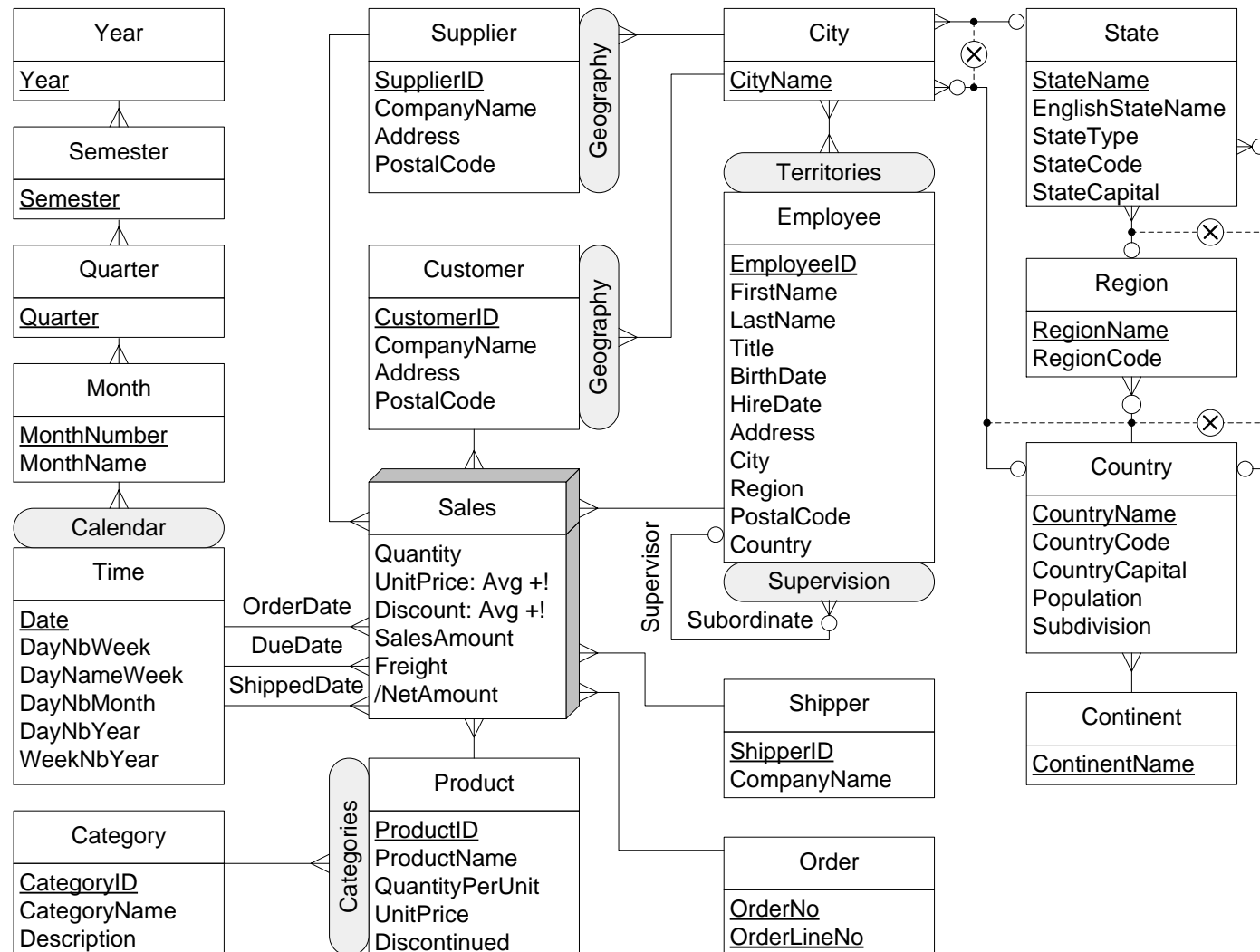
Distributing factor



Exclusive relationships



MultiDim Conceptual Schema of the Northwind Data Warehouse



Conceptual Data Warehouse Design

Outline

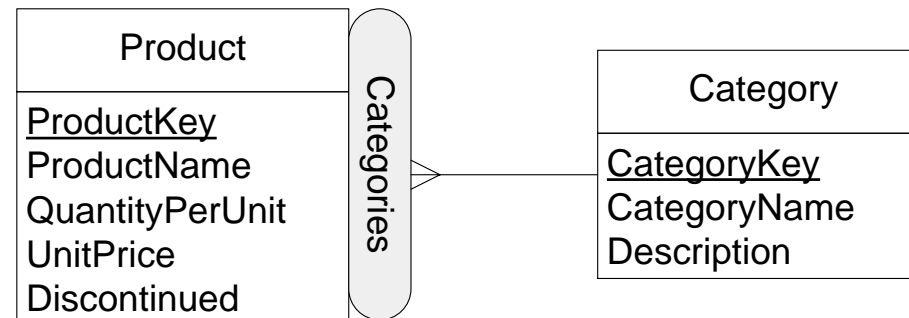
- ◆ A Conceptual Model for Data Warehouses
- ➡ **Dimension Hierarchies**
- ◆ Advanced Modeling Aspects

Dimension Hierarchies

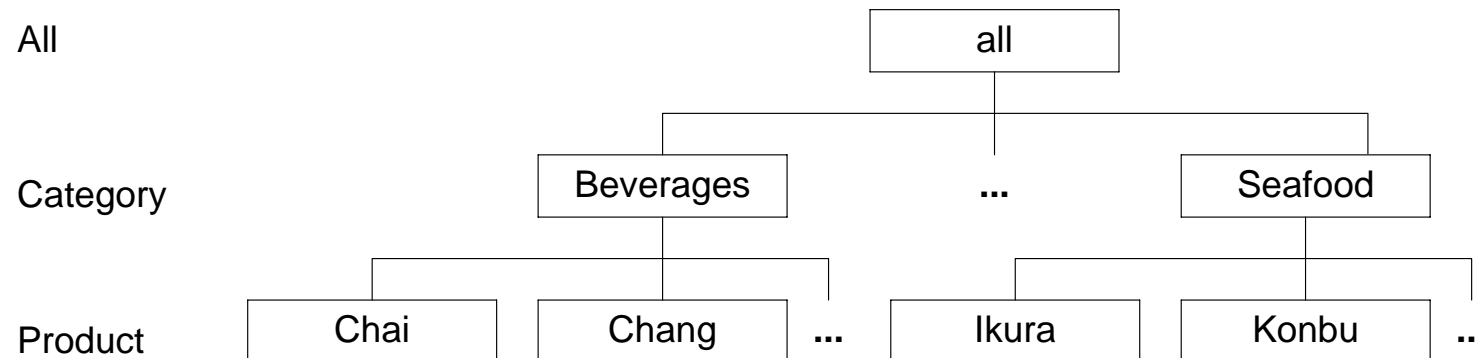
- ◆ Crucial in analytical applications
- ◆ Enable analysis at various abstraction levels
- ◆ In real-world situations, users must deal with complex hierarchies of various kinds
- ◆ Logical models of current DW and OLAP systems allow only a limited set of kinds of hierarchies
 - Users unable to capture the essential semantics of multidimensional applications
 - They must limit their analysis to the predefined set of hierarchies supported by the tools
- ◆ At the conceptual level, focus is to establish sequences of levels that should be traversed during roll-up and drill-down
- ◆ Distinction between the various kinds of hierarchies should also be made at the instance level
- ◆ Cardinalities in parent-child relationships must be considered
- ◆ MultiDim includes classification of hierarchies at the schema and instance level and proposes a graphical notation

Balanced Hierarchies

- ◆ At **schema level**: only one path where all parent-child relationships are many-to-one and mandatory

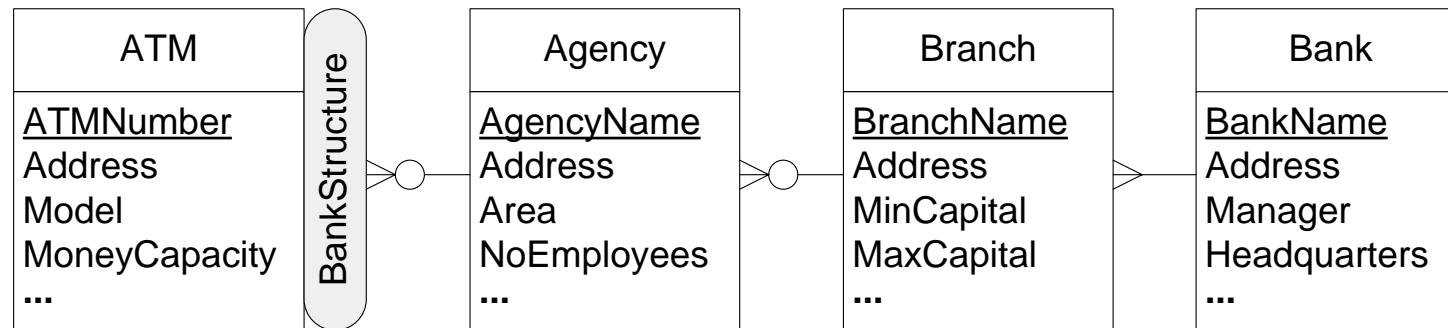


- ◆ At **instance level**: members form a balanced tree (all the branches have the same length)
- ◆ All parent members have at least one child member, and a child belongs exactly to one parent

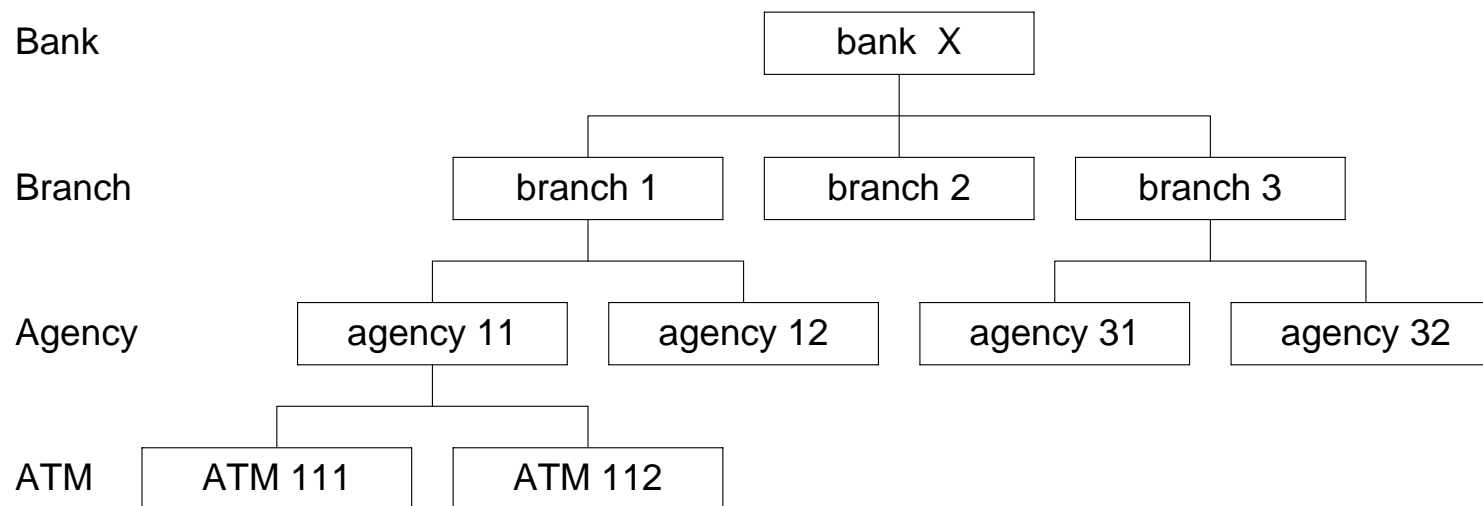


Unbalanced Hierarchies

- ◆ At **schema level**: one path where all parent-child relationships are many-to-one, but some are optional

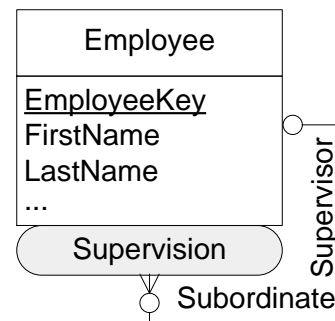


- ◆ At **instance level**: members form a unbalanced tree

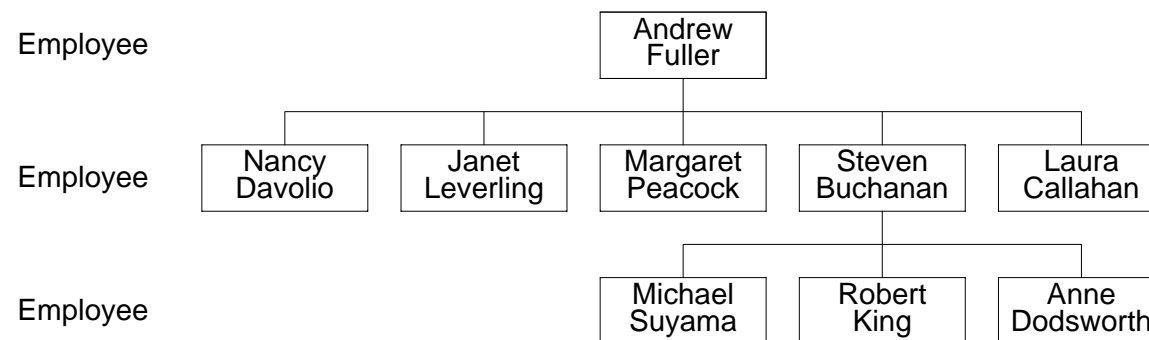


Recursive Hierarchies

- ◆ A special case of unbalanced hierarchies
- ◆ The **same level** is linked by the two roles of a parent-child relationship
- ◆ Used when all hierarchy levels express the same semantics
- ◆ The characteristics of the parent and child are similar (or the same)
- ◆ **Schema level**

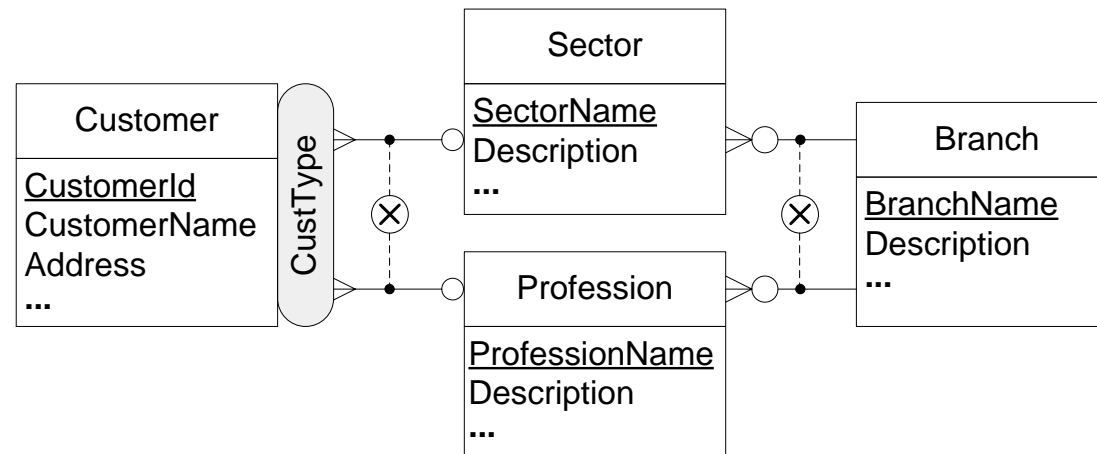


- ◆ **Instance level**

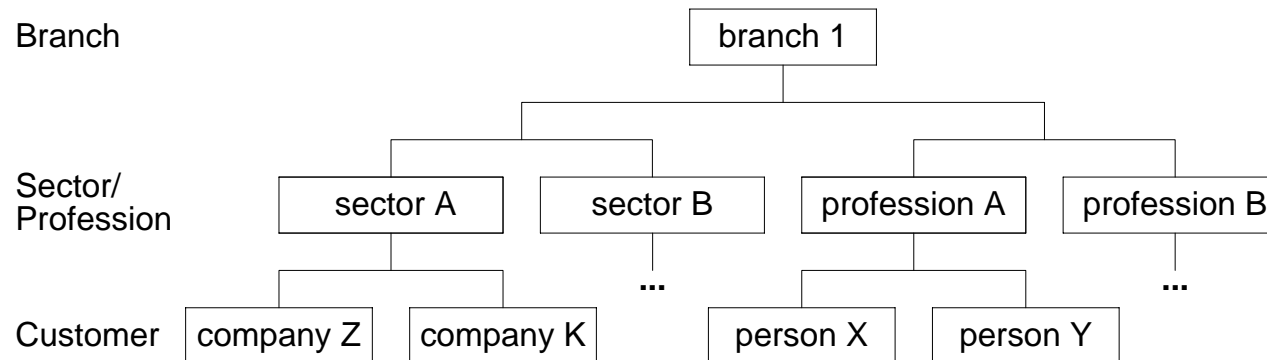


Generalized Hierarchies

- ◆ At **schema level**: multiple **exclusive** paths sharing at least the leaf level; may also share other levels
- ◆ Two aggregation paths, one for each type of customer



- ◆ At **instance level**: each member belongs to only one path

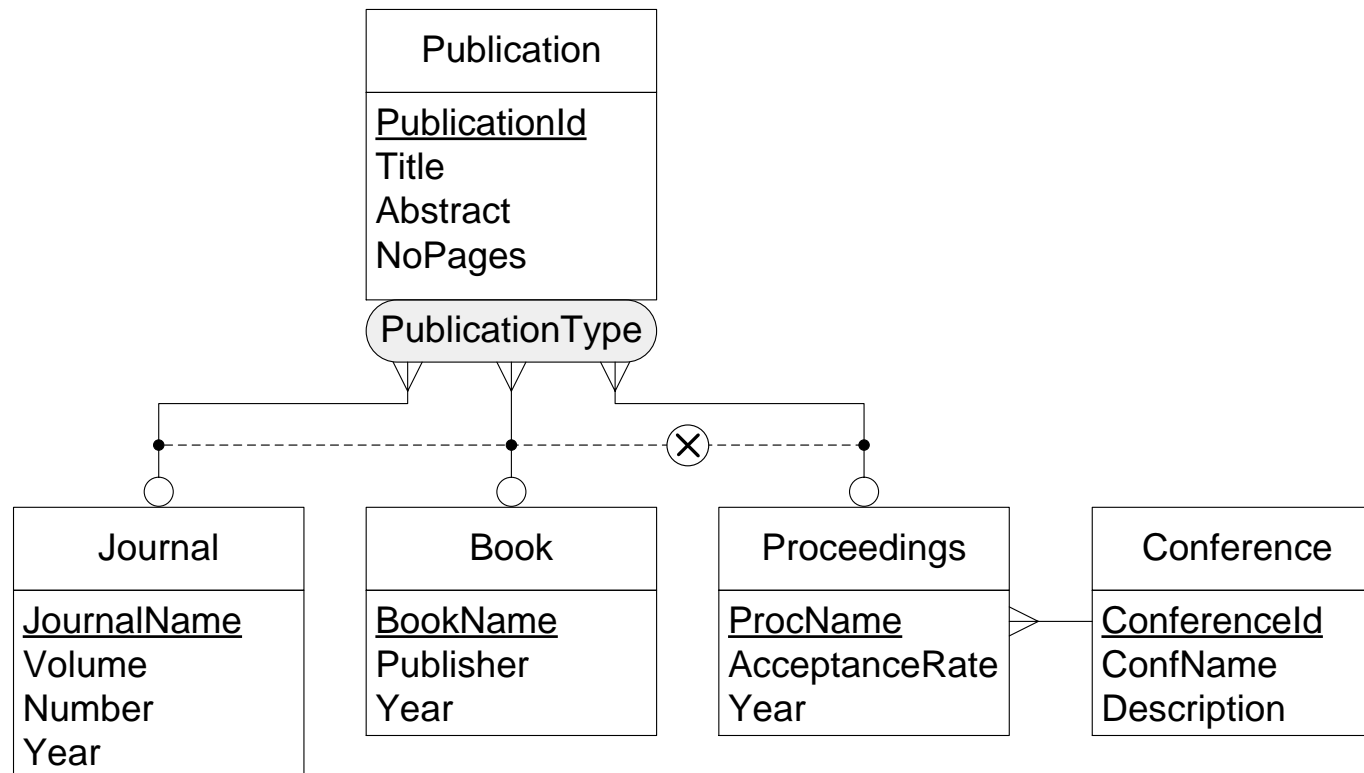


Generalized Hierarchies

- ◆ **Supertype** of the generalization/specialization relationship is used in generalized hierarchies for representing a **leaf level**
- ◆ It only includes those attributes that represent concepts at the lowest granularity
 - E.g., **CustomerId**, **CustomerName**, and **Address**
- ◆ This kind of hierarchy **does not satisfy the summarizability conditions**
 - The mapping from the splitting level to the parent levels is incomplete
 - * E.g., not all customers roll up to the **Sector** level
 - * E.g., not all customers are mapped to the **Profession** level
- ◆ Conventional aggregation mechanism should be modified when a splitting and joining levels are reached in a drill-down and roll-up operations
- ◆ Traditional approach can be used for aggregating measures for common hierarchy levels

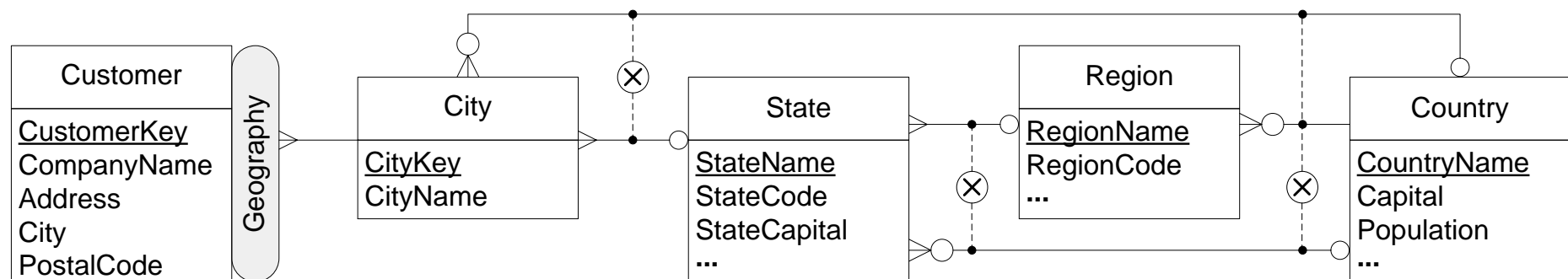
Generalized Hierarchies

- ◆ In generalized hierarchies, it is not necessary that splitting levels must be joined at a higher level



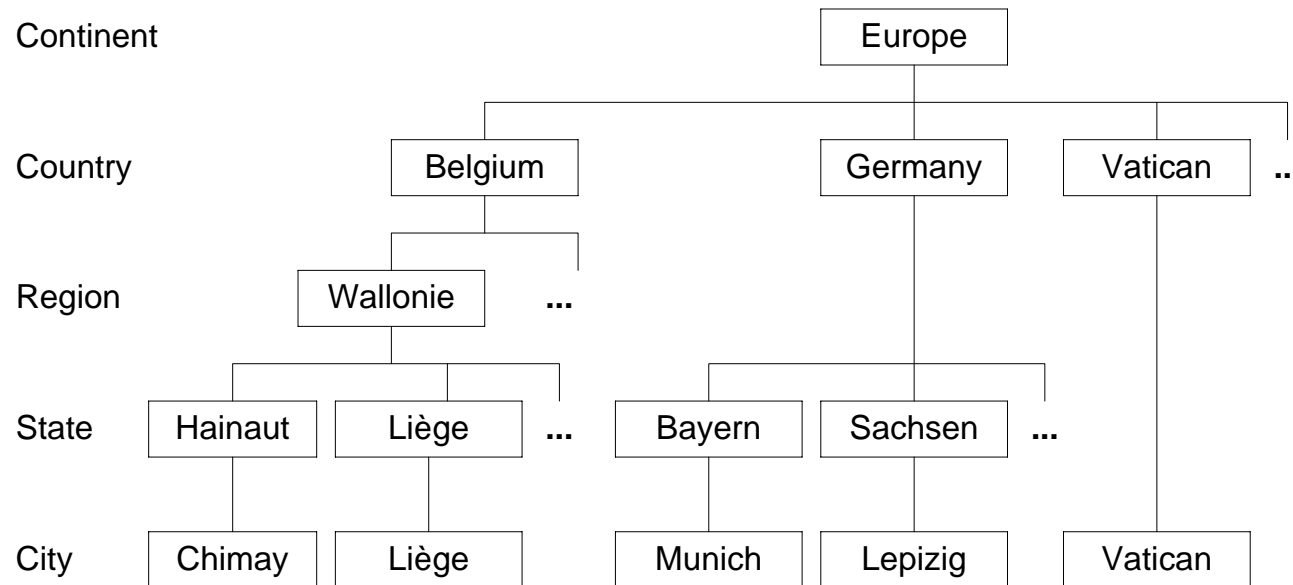
Noncovering Hierarchies

- ◆ Also known as **ragged** or **level-skipping hierarchies**
- ◆ A **special case of generalized hierarchies**
- ◆ At the **schema level**: Alternative paths are obtained by skipping one or several intermediate levels



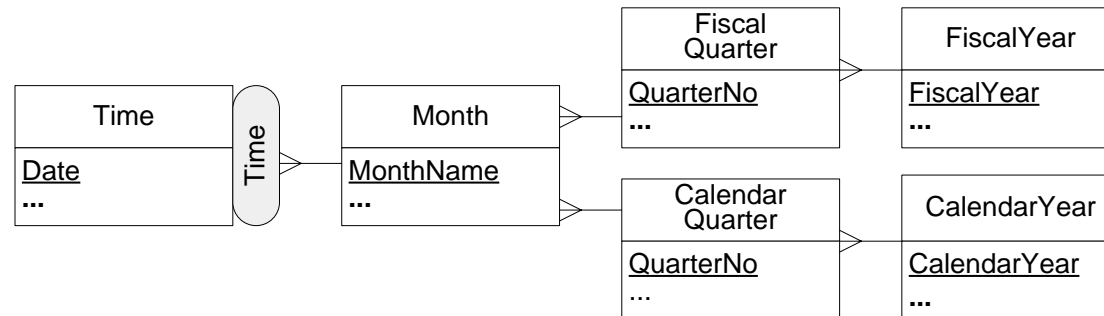
Noncovering Hierarchies

- ◆ At **instance level**: Path length from the leaves to the same parent can be different for different members

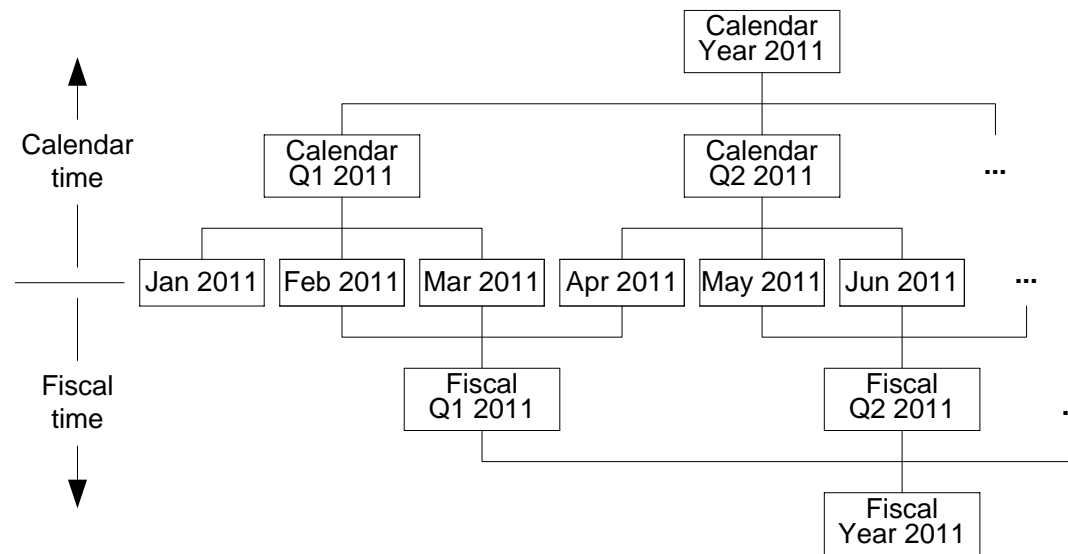


Alternative Hierarchies

- ◆ At **schema level**: Multiple **nonexclusive** hierarchies that share **at least the leaf level** and account for the same analysis criterion

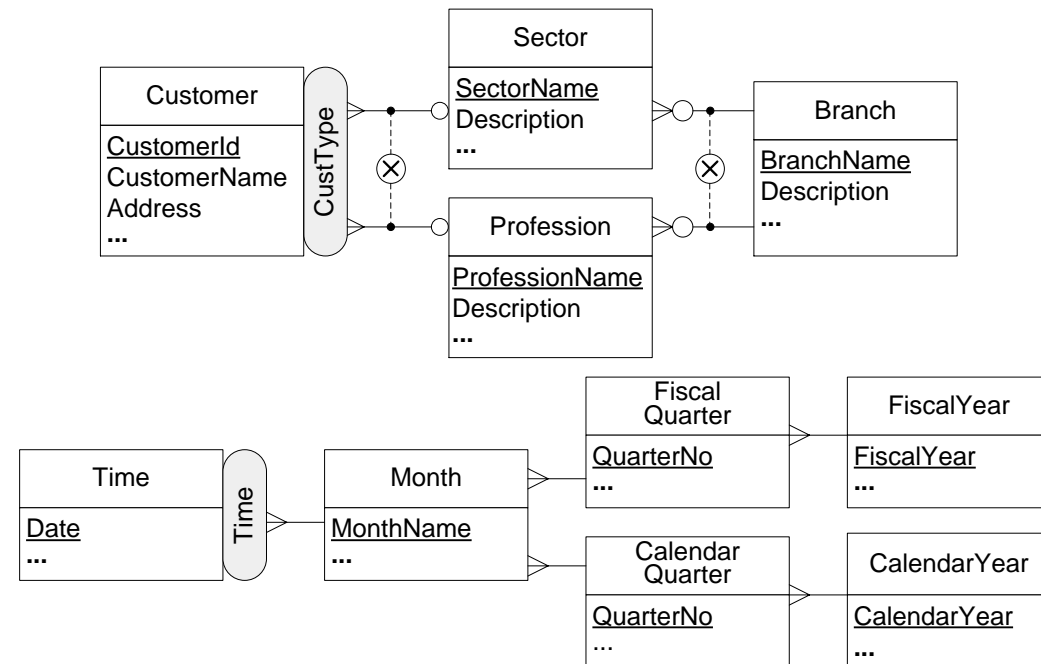


- ◆ At **instance level**: Members form graph



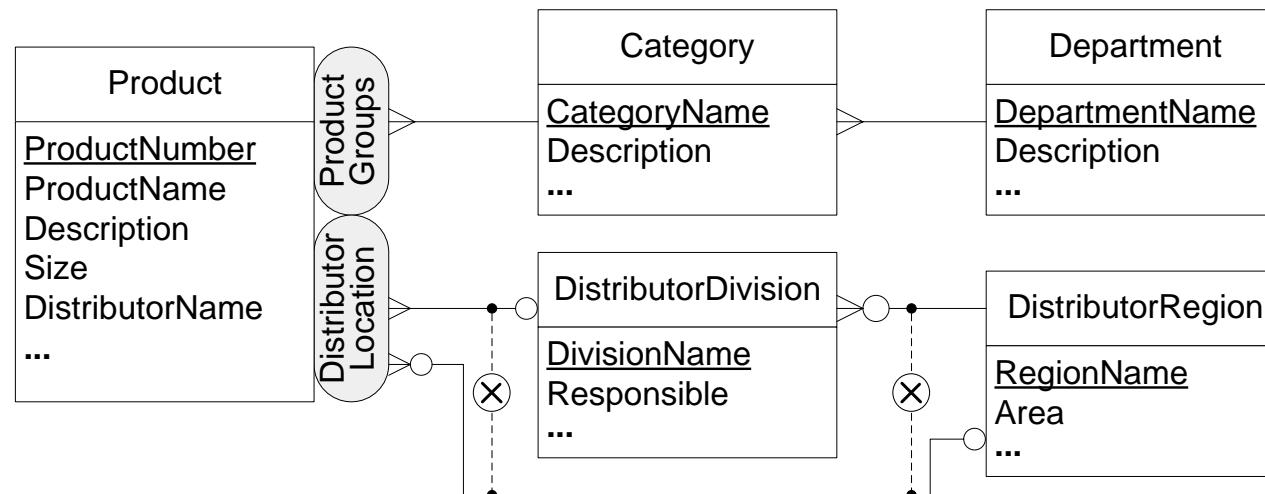
Generalized vs. Alternative Hierarchies

- ◆ Both hierarchies
 - Share some levels
 - Use one analysis criterion
- ◆ A child member
 - Related to only one path in generalized hierarchies
 - Related to all paths in alternative hierarchies and users must choose one for analysis



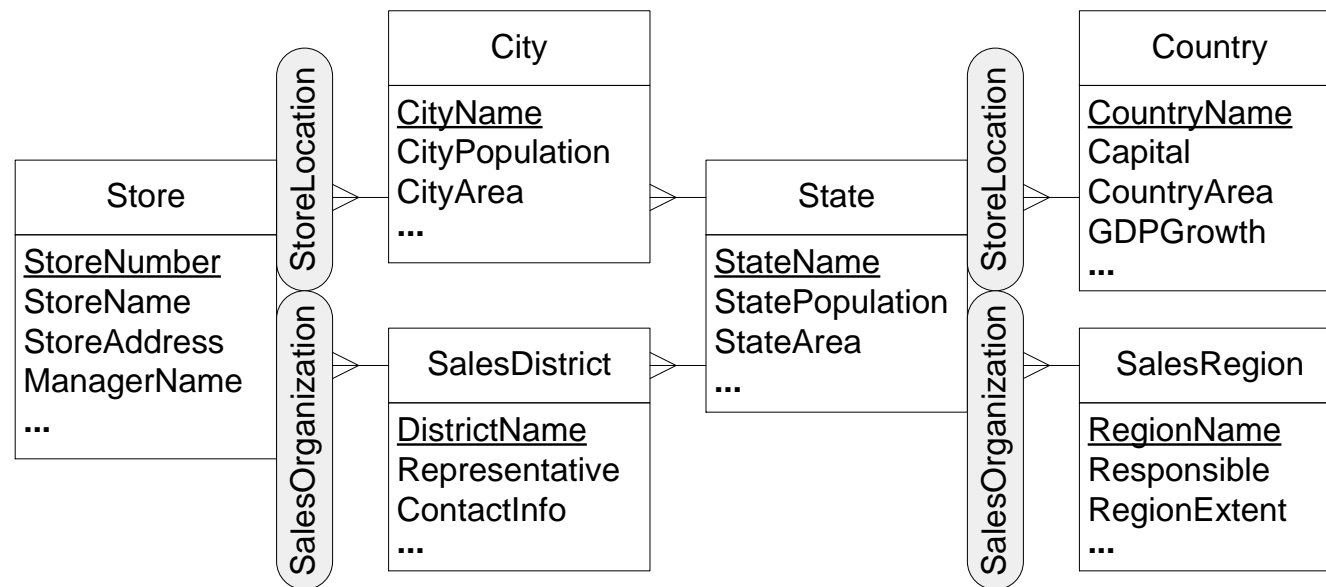
Parallel Hierarchies

- ◆ Dimension has associated **several hierarchies** accounting for **different analysis criteria**
- ◆ Two different types
 - Parallel **independent** hierarchies
 - Parallel **dependent** hierarchies
- ◆ Parallel **independent** hierarchies
 - Composed of disjoint hierarchies, i.e., hierarchies that **do not share levels**
 - Component hierarchies may be of different kinds



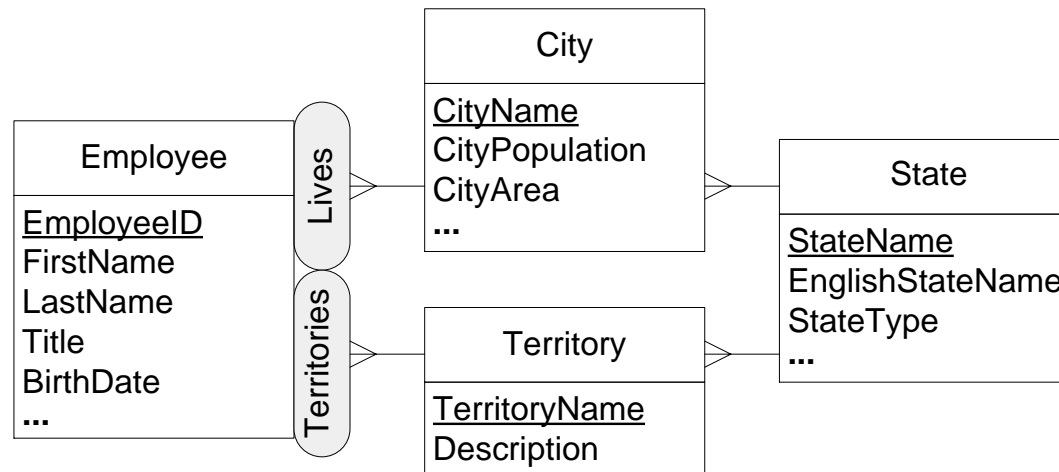
Parallel Hierarchies

- ◆ Parallel **dependent** hierarchies
- ◆ Composed of several hierarchies that account for different analysis criteria and **share some levels**
- ◆ Component hierarchies may be of different kinds



Parallel Hierarchies

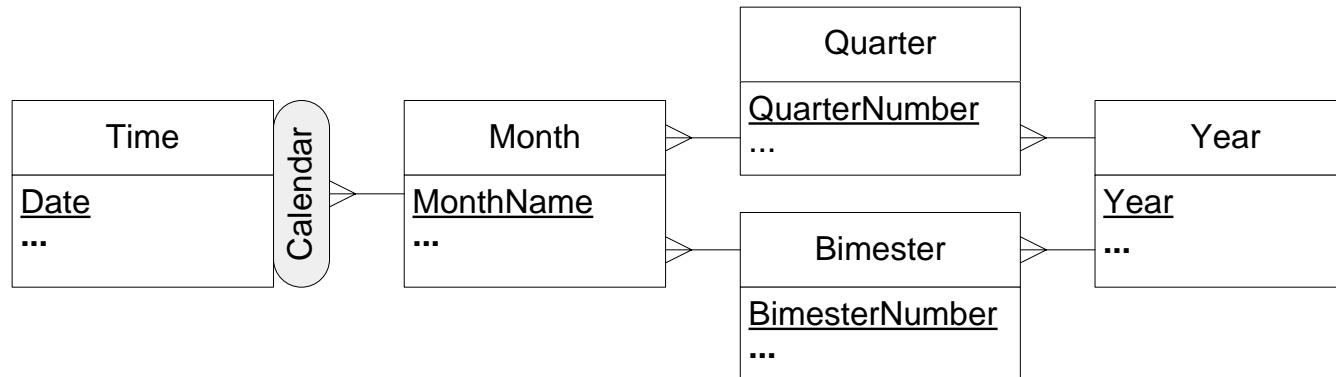
- ◆ Parallel dependent hierarchies leading to different parent members of the shared level



Alternative vs. Parallel Hierarchies

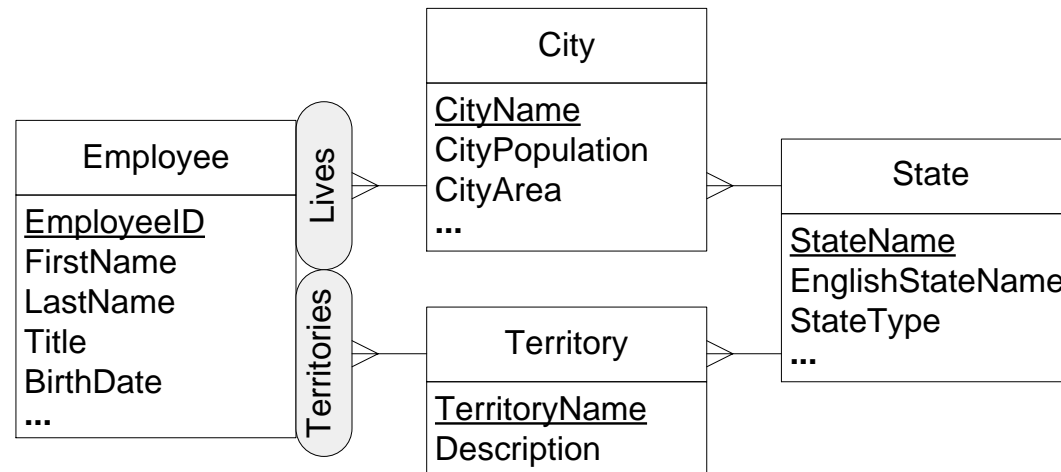
- ◆ Both hierarchies
 - Share some levels
 - May include several simple hierarchies
- ◆ Criterion
 - Only one for alternative hierarchies
 - Several for parallel hierarchies
- ◆ Combining hierarchies
 - Meaningless for alternative hierarchies
 - Useful for parallel hierarchies
- ◆ Reusing aggregated measures for common levels
 - Can be done for alternative hierarchies
 - Cannot be done for parallel hierarchies

Alternative vs. Parallel Hierarchies



- ◆ Aggregated measure for the **Month** level **can be reused** regardless the path
- ◆ Traversing the **Calendar** hierarchy from a specific day in the **Time** level will end up in the same year independently of which path is used

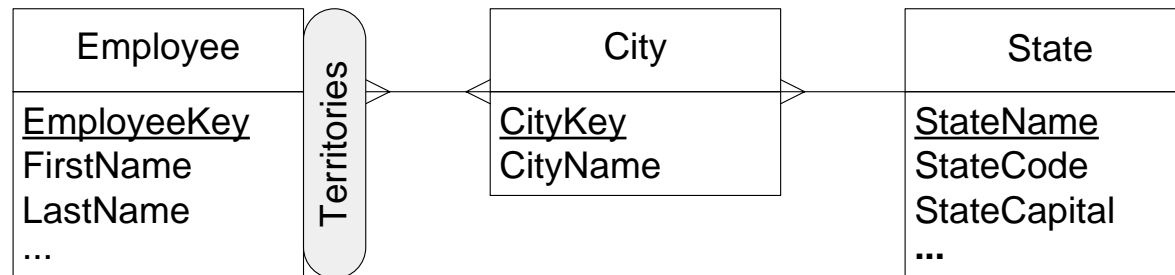
Alternative vs. Parallel Hierarchies



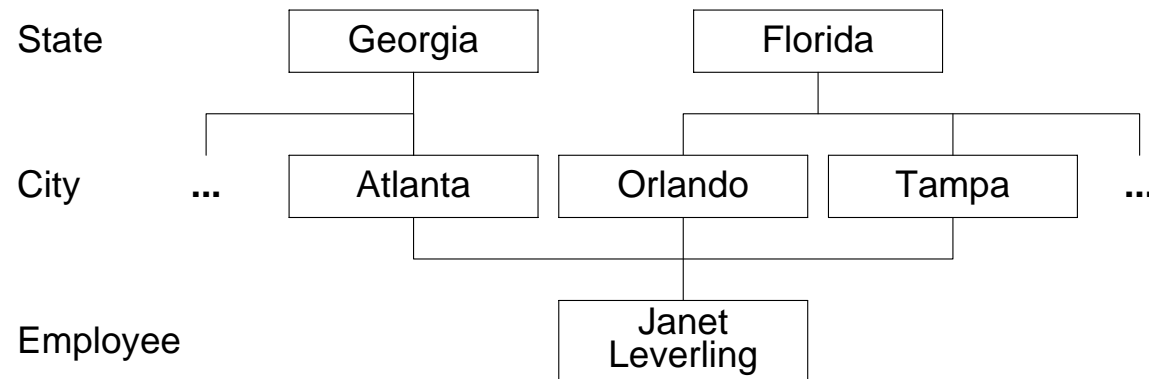
- ◆ Aggregated measure for **State** level **cannot be reused** regardless the path
- ◆ Traversing the hierarchies **Live** and **Work** from the **Employee** to the **State** level will lead to different states for employees who live in one state and work in another
- ◆ Example:
 - Sales generated by E1, E2, and E3: \$50, \$100, and \$150, respectively
 - All employees live in state A, but only E1 and E2 work in A
 - Aggregating sales to the **State** through hierarchy **Lives**: \$300
 - Aggregating sales to the **State** through hierarchy **Territories**: \$150
 - Both results are correct, since the two hierarchies represent different analysis criteria

Nonstrict Hierarchies

- ◆ At **schema level**: At least one many-to-many cardinality

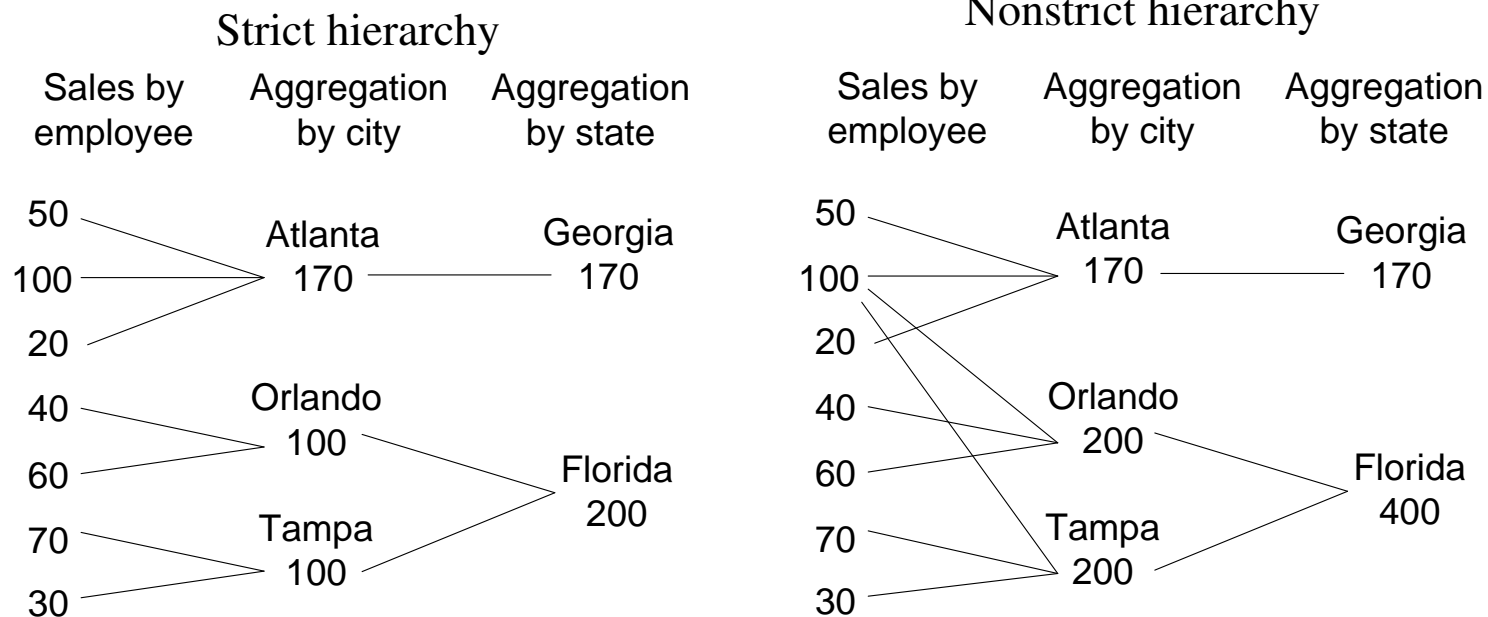


- ◆ At **instance level**: Members form a graph



Nonstrict Hierarchies: Double Counting

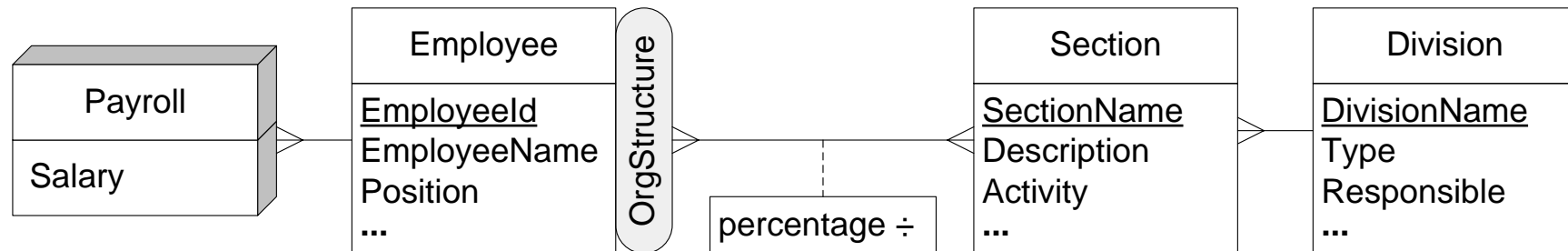
- ◆ Problem: **Double counting** of measures when a roll-up operation reaches a many-to-many relationship
- ◆ Example
 - Left: Janet L. works in Atlanta; Right: Janet L. works in three cities



Nonstrict Hierarchies: Solutions for Double Counting

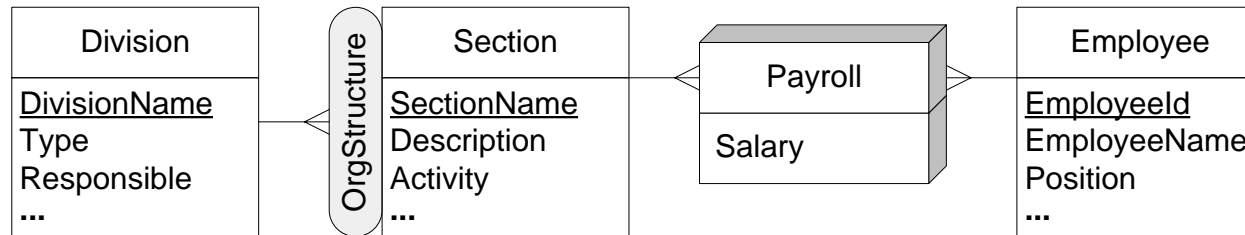
- ◆ Include a **distributing factor**
- ◆ Calculate **approximate** values of a distributing factor
- ◆ **Transform** a nonstrict hierarchy into a strict one:
 - **Create a new parent member** for each group of parent members linked to a single child member in a many-to-many relationship
 - * Example: Create a member Atlanta-Orlando-Tampa, and assign Janet's sales (\$100)
 - Problem: Must create a new member also at **State** level
 - **Choose one parent member as primary** and ignore the existence of other parent members
 - **Split the hierarchy in two** at the many-to-many relationship, where the levels from the parent level and beyond become a new dimension
- ◆ Each solution has its advantages and disadvantages and requires special aggregation procedures
- ◆ Appropriate solution must be chosen according to the situation at hand and user's requirements

Nonstrict Hierarchies: Distributing Factor

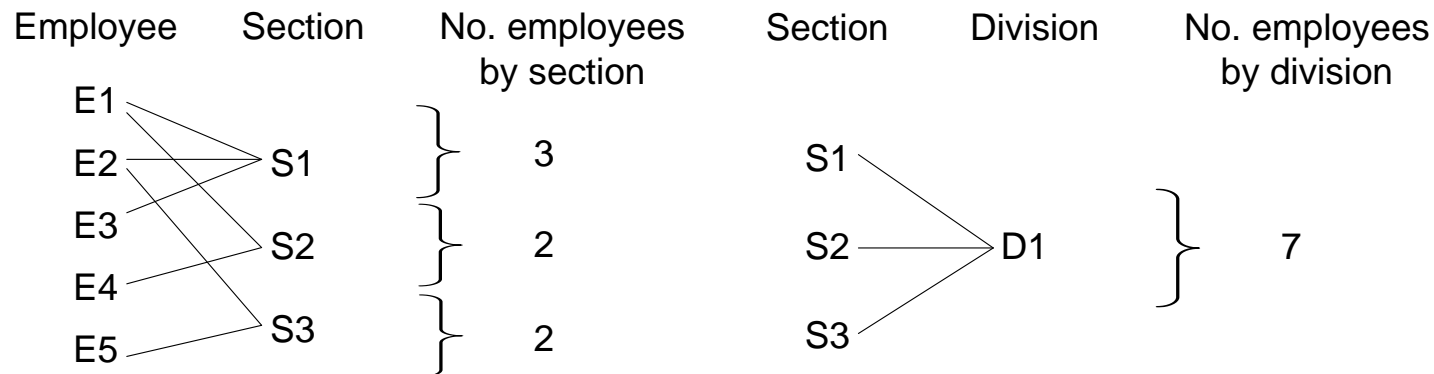


- ◆ Employees may work in several sections
- ◆ A measure represents an employee's overall salary, i.e., the sum of the salaries paid in each section
- ◆ Distributing factor determines how measures are divided between several parent members
- ◆ Distributing factor is **not always known**
 - Percentage of time that an employee works in a section must be added to schema
- ◆ Sometimes this distribution is **impossible to specify**
 - E.g., participation of customer in joint account
- ◆ Distributing factor can be **approximated** by considering the total number of parent members with which the child member is associated
 - If an employee works in three sections, 1/3 of the value of the measure aggregated for each one

Nonstrict Hierarchies: Splitting the Hierarchy



- ◆ Transform a nonstrict hierarchy into a strict one with an **additional dimension**
- ◆ **Focus of analysis** has changed from employee's salaries to employee's salaries by section
- ◆ Can only be applied when the **measure distribution is known**
- ◆ Nevertheless, double counting problem still remains
- ◆ Example: calculate the number of employees by section or by division

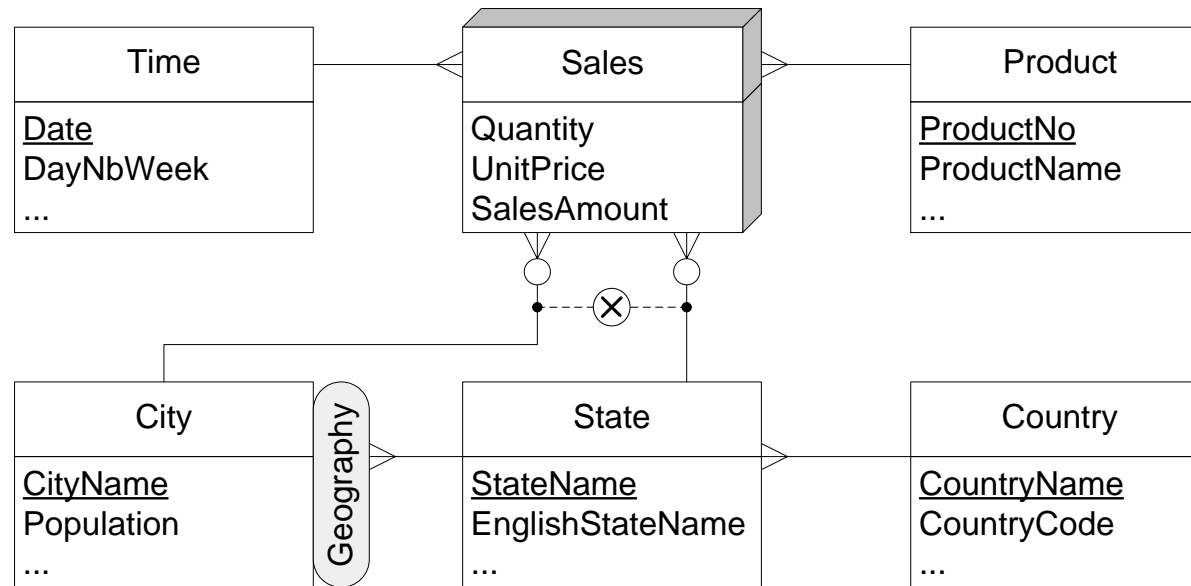


Conceptual Data Warehouse Design

Outline

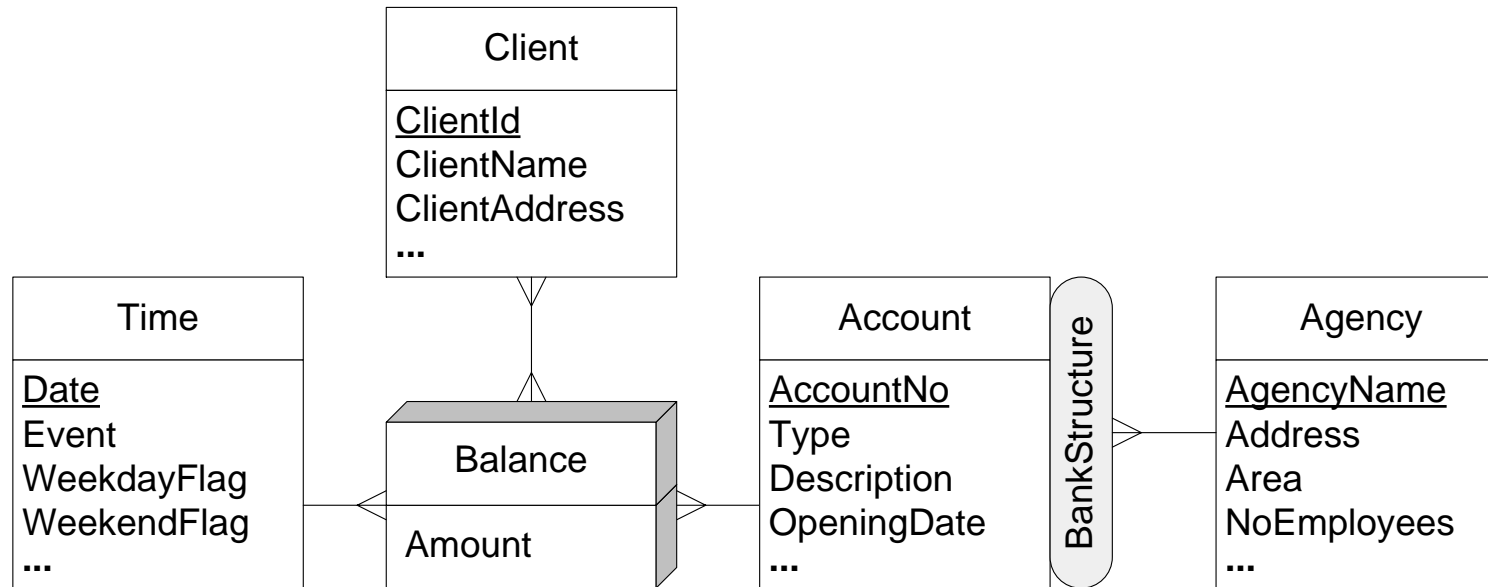
- ◆ A Conceptual Model for Data Warehouses
- ◆ Dimension Hierarchies
- ➡ **Advanced Modeling Aspects**

Advanced Modeling Aspects: Facts with Multiple Granularities



- ◆ Sales for USA captured at the city level, for Europe at the state level
- ◆ Or a patient may be related to a diagnosis at the lowest granularity, but, of not sure, at (more imprecise) diagnosis family

Advanced Modeling Aspects: Many-to-Many Dimensions



- ◆ Several members of the dimension participate in the **same fact member** (different from nonstrict hierarchies, where **levels** have m:n relationships)
- ◆ Multidimensional schema for the analysis of bank accounts

Advanced Modeling Aspects: Many-to-Many Dimensions

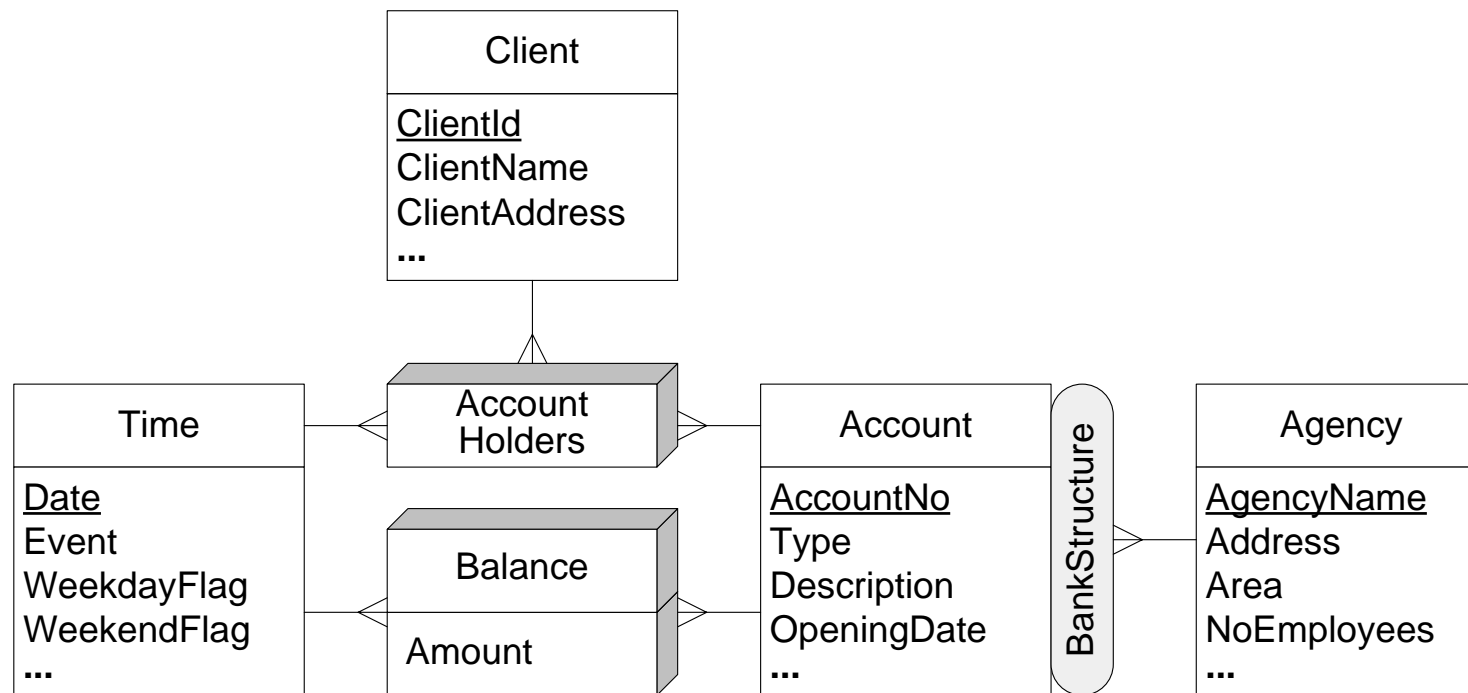
◆ Example of double-counting problem

Time	Account	Client	Balance
T1	A1	C1	100
T1	A1	C2	100
T1	A1	C3	100
T1	A2	C1	500
T1	A2	C2	500

Advanced Modeling Aspects: Many-to-Many Dimensions

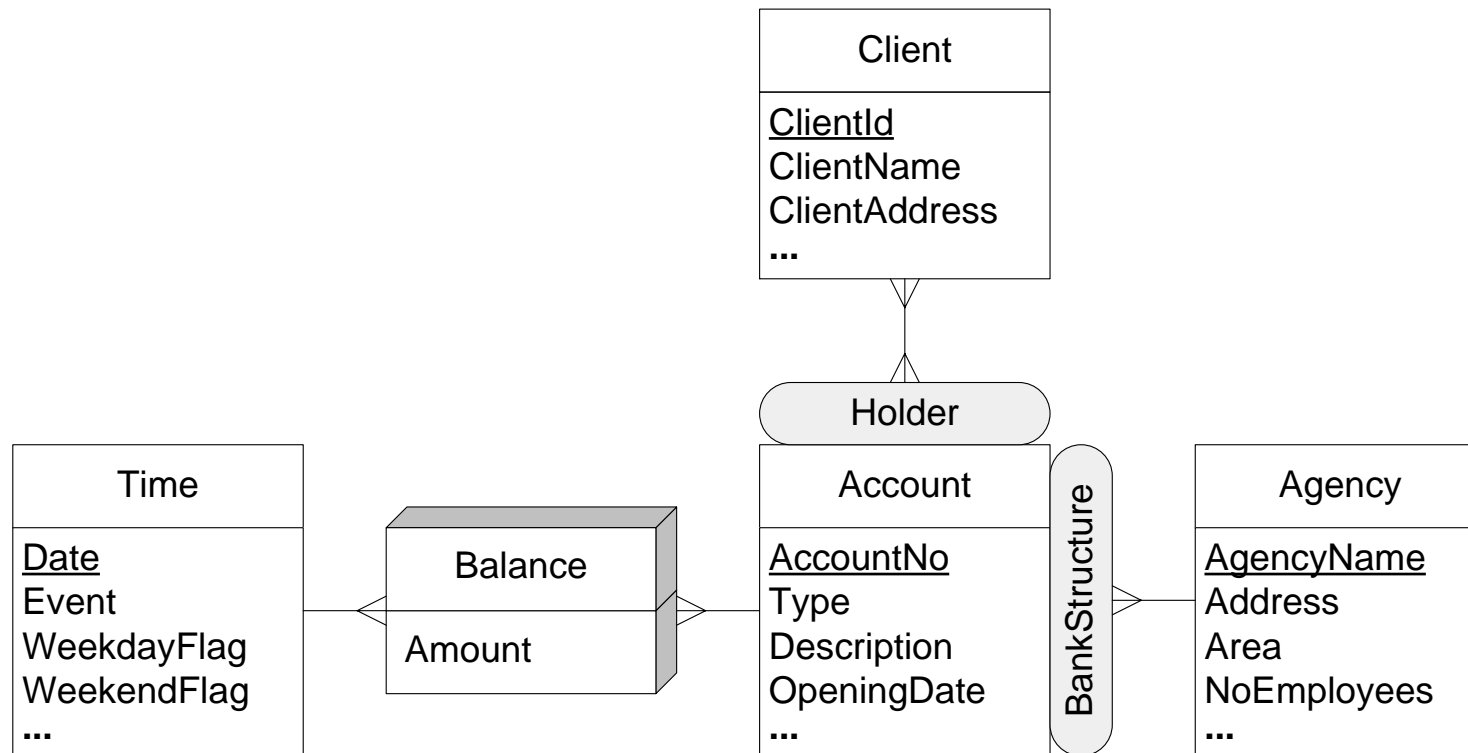
- ◆ Two possible decompositions of the fact

(1) Create two facts



Advanced Modeling Aspects: Many-to-Many Dimensions

- ◆ Two possible decompositions of the fact
 - (2) Include a nonstrict hierarchy



Advanced Modeling Aspects: Many-to-Many Dimensions

◆ Alternative decomposition of the schema

