Conceptual Data Warehouse Design

Helena Galhardas
DEI/IST

References

• A. Vaisman and E. Zimányi, Data Warehouse Systems: Design and Implementation, Springer, 2014 (chpt 4)
Outline

- Conceptual Modeling of Data Warehouses:
  - MultiDim
- Dimension Hierarchies
- Advanced Modeling Aspects

Conceptual Multidimensional Models

- **Conceptual models**: concise description of the user data requirements without taking into account implementation details
  - Allow better communication between designers and users to understand application requirements
  - More stable than implementation-oriented (logical) schema, which changes with the platform
- **No well-established** conceptual model for multidimensional data
  - Several proposals based on UML, on the ER model, or using specific notations
- Currently, data warehouses are designed using mostly **logical models** (star and snowflake schemas)
  - Difficult to express requirements
  - Limit users to define 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 (ER) 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 (1)

- **Level**: entity type (or entity set)
  - Has a set of attributes that describe characteristics of their members
- **Hierarchy**: several related levels
- **Dimension**: composed of one level or one or more hierarchies
- **Member**: every instance of a level
- **Child and parent levels**: the lower and higher levels
MultiDim Model: Notation (2)

- **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
  
  - [0,1]
  - (1,1)
  - (0,0)
  - < (0,1)

- **Criterion**: Expresses different hierarchical structures used for analysis

- **Key (identifier) attribute**: Indicates how child members are grouped

- **Descriptive attributes**: Describe characteristics of members

MultiDim Model: Notation (3)

- **Fact**: Relates measures to leaf levels in dimensions
  
  - **Additive**: can be meaningfully summarized along all the dimensions, using addition (most common type)
  
  - **Semiadditive**: can be meaningfully summarized using addition along some dimensions (example: inventory quantities, which cannot be added along the Time dimension)
  
  - **Nonadditive measures** cannot be meaningfully summarized using addition across any dimension (Ex: item price, cost per unit, and exchange rate)

- Dimensions can be related to a fact with **one-to-one, one-to-many, or many-to-many relationships**

- Dimension can be related several times to a fact with different **roles**
MultiDim Model: Notation (4)

- Distributing attribute

- Exclusive relationship

MultiDim Conceptual Schema of the Northwind Data Warehouse
Outline

• Conceptual Modeling of Data Warehouses:
  – MultiDim

  ➢ Dimension Hierarchies
    ➢ Balanced Hierarchies
    ➢ Unbalanced Hierarchies
    ➢ Generalized Hierarchies
    ➢ Alternative Hierarchies
    ➢ Parallel Hierarchies
    ➢ Nonstrict 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
  – Must limit their analysis to the predefined set of hierarchies supported by the tools
• MultiDim includes classification of hierarchies at the schema and instance level and proposes a graphical notation
Balanced Hierarchies

- **At schema level**: has 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

![Balanced Hierarchy: schema level](image-url)
Unbalanced Hierarchies

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

![Unbalanced Hierarchy Diagram](image)

• **At instance level:** members form a unbalanced tree (parent members may not have associated child members)

![Recursive (Parent-child) Hierarchy Diagram](image)

Recursive (Parent-child) 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)

• **At instance level:** each member belongs to only one path
Recursive Hierarchy: schema 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:
  - For companies: Customer -> Sector -> Branch
  - For people: Customer -> Profession -> Branch

- At instance level: each member belongs to only one path
Generalized Hierarchies: observations

- **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 without a Joining level

- In generalized hierarchies, it is not necessary that splitting levels must be joined
Noncovering (Ragged or Level-skipping) Hierarchies

- A special case of generalized hierarchies
- Example: City -> State -> Region -> Country -> Continent
- At the schema level: Alternative paths are obtained by skipping one or several intermediate levels
- At instance level: Path length from the leaves to the same parent can be different for different members

Noncovering Hierarchy: schema level
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 a graph

- **Needed to analyze measures from a unique perspective** (e.g., time) using alternative paths
- Measures will participate totally in each component hierarchy => conventional aggregation procedures
- It is not semantically correct to simultaneously combine different component hierarchies
- Users must choose only one of the alternative paths for their analysis and switch to other one if required
Generalized vs. Alternative Hierarchies

- **Both hierarchies**
  - Share some levels
  - Use one analysis criterion

- **A child member**
  - Related to only one of the paths 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 Dependent Hierarchies

- Composed of several hierarchies that account for different analysis criteria and **share some levels**
- Component hierarchies may be of different kinds
- It is possible to **combine levels from different hierarchies**
  - Example of query: sales figures for stores in city A that belong to the sales district B

![Diagram of Parallel Dependent Hierarchies]

Parallel Dependent Hierarchies

- Parallel dependent hierarchies leading to different parent members of the shared level
  - Ex: Traversing the hierarchies Lives and Territory from the Employee to the State level will lead to different states for employees that live in one state and are assigned to another.

![Diagram of Parallel Dependent Hierarchies Example]
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
    - Aggregated measure for the Month level can be reused between both paths
    - 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
  - Cannot be done for parallel hierarchies
    - Aggregated measure for State level cannot be reused between both paths
    - Traversing the hierarchies Live and Territory from the Employee to the State level will lead to different states for employees who live in one state and work in another.

Nonstrict Hierarchies

- At schema level: At least one many-to-many cardinality
- At instance level: Members form an acyclic graph
Strict Hierarchy: schema level

Nonstrict Hierarchies: Double Counting

- Problem: Double counting of measures when a roll-up operation reaches a many-to-many relationship
- Examples of aggregation:

```
<table>
<thead>
<tr>
<th>Strict Hierarchy</th>
<th>Nonstrict Hierarchy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sales by employee</td>
<td>Sales by employee</td>
</tr>
<tr>
<td>Aggregation by city</td>
<td>Aggregation by city</td>
</tr>
<tr>
<td>Aggregation by state</td>
<td>Aggregation by state</td>
</tr>
<tr>
<td>Atlanta 170</td>
<td>Atlanta 170</td>
</tr>
<tr>
<td>Orlando 100</td>
<td>Orlando 200</td>
</tr>
<tr>
<td>Tampa 100</td>
<td>Florida 400</td>
</tr>
<tr>
<td>Florida 200</td>
<td></td>
</tr>
</tbody>
</table>
```
Nonstrict Hierarchies: Solutions for Double Counting

1. Include a distributing factor
2. Calculate approximate values of a distributing factor
3. 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
   - 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 (Solutions 1 and 2)

- 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

![Diagram of nonstrict hierarchy with payroll, employee, section, and division dimensions]
Nonstrict Hierarchies: Splitting the Hierarchy (Solution 3)

- 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

Outline

- Conceptual Modeling of Data Warehouses:
  - MultiDim
- Dimension Hierarchies
  - Advanced Modeling Aspects
    - Facts with multiple granularities
    - Many-to-many dimensions
Facts with Multiple Granularities

- Measures can be captured at multiple granularities
  - Ex: Sales reported at the city level or at the state level
- Use exclusive relationships between the various granularity levels

```
+-----+     +-----+     +-----+
| Date|     | Quantity|     | ProductNo|
+-----+     +-------+  +-------+
| Day|     | UniPrice|     | ProductName|
| No|     |         |     |            |
| Week|     | SalesAmount|   |            |
+-----+     +-------+  +-------+
```

```
+-----+     +-----+     +-----+
| City|     | State|     | Country|
+-----+     +------|     +-----+
| CityName| | StateName| | CountryName|
|        | | EnglishStateName| | CountryCode|
|        | |                   |     |
+-----+     +------|     +-----+
```

Many-to-Many Dimensions

- Schema level: Several members of a dimension participate in the same fact member
  - Ex: an account can be jointly owned by several clients; aggregation of the balance according to the clients will count this balance as many times as the number of account holders

```
+-----+     +-----+     +-----+
| Time|     | Account|     | Agency|
+-----+     +-------+  +-----+
| Date|     | AccountNo|     | AgencyName|
| Event|     | Type|     | Address|
| Week|     | Description|   | Area|
| Day|     | OpeningDate|   | NoEmployees|
| No|     |            |   |        |
| Week|     |            |   |        |
| Flag|     |            |   |        |
|        |     |            |   |        |
+-----+     +-------+  +-----+
```

```
+-----+     +-----+     +-----+
| Client|     | Balance|     | BankDetails|
+-----+     +-------+  +-----+
| ClientId| | Amount|     |          |
| ClientName| |       |     |          |
| ClientAddress| |       |     |          |
+-----+     +-------+  +-----+
```
Many-to-Many Dimensions: instance level

- Example of double-counting problem

<table>
<thead>
<tr>
<th>Time</th>
<th>Account</th>
<th>Client</th>
<th>Balance</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>A1</td>
<td>C1</td>
<td>100</td>
</tr>
<tr>
<td>T1</td>
<td>A1</td>
<td>C2</td>
<td>100</td>
</tr>
<tr>
<td>T1</td>
<td>A1</td>
<td>C3</td>
<td>100</td>
</tr>
<tr>
<td>T1</td>
<td>A2</td>
<td>C1</td>
<td>500</td>
</tr>
<tr>
<td>T1</td>
<td>A2</td>
<td>C2</td>
<td>500</td>
</tr>
</tbody>
</table>

Many-to-Many Dimensions: solution to the double counting problem (2)

- Two possible decompositions of the fact
  2. Including a nonstrict hierarchy

Diagram with entity relationships:
- Client
  - ClientId
  - ClientName
  - ClientAddress
  - ...
- Time
  - Date
  - Event
  - WeekdayFlag
  - WeekendFlag
  - ...
- Balance
  - Amount
- Account
  - AccountNo
  - Type
  - Description
  - OpeningDate
  - ...
- Agency
  - AgencyName
  - Address
  - NoEmployees
  - ...

Diagram showing relationships between entities and attributes.
Many-to-Many Dimensions: solution to the double counting problem (1)

- Two possible decompositions of the fact
  1. Creating two facts

```
Many-to-Many Dimensions: solution to the double counting problem (3)

- Alternative decomposition of the schema: an additional level is created to represent the groups of clients participating in joint accounts
```
Next Lecture

• Logical Data Warehouse Design