Data-Warehouse-, Data-Mining- und OLAP-Technologien

Chapter 3: Data Warehouse Design

Bernhard Mitschang
Universität Stuttgart

Winter Term 2014/2015
Overview

• Data Warehouse Design Process
• Conceptual Design
• Logical Design
• Details of Logical Design
• Physical Design
Database Design

• Process Model:
  - information collection
  - semantical data modeling
  - logical data modeling
  - database installation

  - analysis of meaning
  - rough modeling
  - precise modeling

  - interview
  - noun analysis
  - brainstorming
  - document analysis
  - ...

  - ERM
  - NIAM
  - EXPRESS-G
  - IDEF1X
  - STEP
  - UML
  - ...

  - conceptual schema

  - hierarchical
  - network-like
  - relational
  - object-oriented
  - ...
  - XML

  - DBMS independent
  - DBMS dependent

  - conceptional schema design
  - logical schema design
  - physical schema design

  - DB2
  - INGRES
  - ORACLE
  - ONTOS
  - ...

  - time
Data Warehouse Design Process

**Requirement analysis and -specification**
- Operational database schema

**Conceptual design**
- Semiformal business concept
- Formal conceptual schema

**Logical design**
- Formal logical schema

**Physical design**
- Physical database schema

- time

- interview
- noun analysis
- brainstorming
- document analysis
- ...

- starER
- dimensional fact model
- ME/R
- mUML
- ...

- multidimensional
- relational
- object-relational
- ...

- DB2
- ORACLE
- MS Server
- Essbase
- MS Analysis Services
- ...

Source: [HLV00]
Overview

• Data Warehouse Design Process
• Conceptual Design
  ▪ Multidimensional Model
  ▪ Dimensional Fact Model
  ▪ starER
• Logical Design
• Details of Logical Design
• Physical Design
Conceptual Design

Transformation of the semi-formal business requirements specification into a formalized conceptual multidimensional schema.

• Most approaches produce a graphical multidimensional schema: Dimensional Fact Model, starER, UML-based, …

• Conceptual Design is based on:
  ▪ business requirement specification
  ▪ ER schema of the operational systems

• Process model to conceptual data warehouse design:
  ▪ context definition of measures
  ▪ dimensional hierarchy design
  ▪ definition of summarizability constraints
A data warehouse is based on a multidimensional data model which views data in the form of a **data cube**. A data cube, such as sales, allows data to be modeled and viewed in multiple dimensions.

In data warehousing literature, an n-D base cube is called a base cuboid. The top most 0-D cuboid, which holds the highest-level of summarization, is called the apex cuboid. The lattice of cuboids forms the data cube.
Data Cube

• Example Data Cube for SALES modeled and viewed in multiple dimensions (item, time, location, and supplier)

0-D (apex) cuboid

1-D cuboids

2-D cuboids

3-D cuboids

4-D (base) cuboid
Basic Elements of a Conceptual Model

Fact data
- Mostly numeric data that is observed or measured
- Example: turnover/sales, number of pieces, …

Qualities
- Represent a state, a status or a mode
- Cannot be aggregated (in contrast to fact data)
- Example: shipping mode, status

Attributes
- Describe dimension objects
- Mostly text-based descriptions
- Example: product descriptions, customer profiles, addresses

Dimensions
- Strongly associated attributes
- Typically 5 to 20 attributes
- Showing mostly hierarchical relationships
- Example: product information, customer information, time references
Conformed Dimensions

A conformed dimension is a dimension that means the same thing with every possible fact to which it can be joined.

- A conformed dimension is identically the same dimension in each data mart.
- Conformed dimensions support:
  - A single dimension can be used against multiple facts in the same database space.
  - User interfaces and data content are consistent whenever the dimension is used.
  - There is a consistent interpretation of attributes and, therefore, rollups across data marts.
Dimensional Fact Model (DFM)

- Part of work on a complete and consistent design methodology.
- Multidimensional conceptual model including a graphical representation.
- Representation of reality consists of a set of fact schemes.
- Basic elements of a fact scheme \( f = (M, A, N, R, O, S) \) are facts, measures, attributes, dimensions, and hierarchies.
- Other features which may be represented on fact schemes are the additivity of fact attributes along dimensions, the optionality of dimension attributes, and the existence of non-dimension attributes.
- The multidimensional model may be mapped on the logical level differently depending on the underlying DBMS.

Source: [GMR98]
DFM: Basic Definitions

- Let $g=(V, E)$ be a directed, acyclic and weakly connected graph.
- $g$ is a **quasi-tree** with root in $v_0 \in V$ if each other vertex $v_j \in V$ can be reached from $v_0$ through at least one directed path. $\text{path}_{0j}(g) \subseteq g$ denotes a directed path starting in $v_0$ and ending in $v_j$.
- Given $v_i \in \text{path}_{0j}(g)$,
  - $\text{path}_{ij}(g) \subseteq g$ denotes a directed path starting in $v_i$ and ending in $v_j$,
  - $\text{sub}(g, v_i) \subseteq g$ denotes the quasi-tree rooted in $v_i \neq v_0$. 
**DFM: Facts and Measures**

- **A fact** is a focus of interest for the decision-making process. It models an event occurring in the enterprise world (e.g., sales and shipments).
- **M** is a set of measures. Each **measure** $m_i \in M$ is defined by a numeric or Boolean expression which involves values acquired from the operational information systems.
- **Graphical representation:**
  - A fact is represented by a box which reports the fact name and, typically, one or more measures.

```
<table>
<thead>
<tr>
<th>fact</th>
<th>measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>SALE</td>
<td>qty sold</td>
</tr>
<tr>
<td></td>
<td>no. of customers</td>
</tr>
</tbody>
</table>
```
DFM: Attributes and Dimensions

- **Dimensions** are discrete attributes which determine the minimum granularity adopted to represent facts.
- **A** is a set of **dimension attributes**.
  - Each dimension attribute $a_i \in A$ is characterized by a discrete domain of values, $\text{Dom}(a_i)$.
- **N** is a set of **non-dimension attributes**.
  - A non-dimension attribute contains additional information about a dimension attribute and is connected by a -to-one relationship.
  - Unlike dimension attributes, non-dimension attributes cannot be used for aggregation.

- **Graphical representation:**
  - Dimension attributes are represented by circles, non-dimension attributes by lines.

![Graphical representation diagram]

- **Dimension attribute**: sales manager
- **Non-dimension attribute**: address
- **Dimension**: store
DFM: Hierarchies

- **Hierarchies** are made up of discrete dimension attributes linked by -to-one relationships, and determine how facts may be aggregated and selected significantly for the decision-making process.

- **R** is a set of ordered tuples, each having the form \((a_i, a_j)\) where \(a_i \in A \cup \{a_0\}\) and \(a_j \in A \cup N\) (\(a_i \neq a_j\)), such that the graph \(qt(f) = (A \cup N \cup a_0, R)\) is a quasi-tree with root \(a_0\).

- \(a_0\) is a **dummy attribute** playing the role of the fact \(f\) on which the scheme is centered. The couple \((a_i, a_j)\) models a -to-one relationship between attributed \(a_i\) and \(a_j\).

- Each element in \(\text{Dim}(f) = \{a_i \in A \mid (a_0, a_i) \in R\}\) is a **dimension**.

- The **hierarchy** on dimension \(d_i \in \text{Dim}(f)\) is the quasi-tree rooted in \(d_i\): \(\text{sub}(qt(f), d_i)\).

- \(O \subseteq R\) is a set of **optional relationships**. The domain of each dimension attribute \(a_j\) such that \(\exists (a_i, a_j) \in O\) includes a NULL value.
DFM: Hierarchies

• Graphical representation:
  ▪ Subtrees rooted in dimensions are hierarchies. The arc connecting two attributes represents a -to-one relationship between them.
  ▪ Optional relationships are represented by marking with a dash the corresponding arc.

• Example:
DFM: Aggregation

- $S$ is a set of aggregation statements.
  - Each aggregation statement consists of a triple $(m_j, d_i, \Omega)$ where $m_j \in M$, $d_i \in \text{Dim}(f)$ and $\Omega \in \{'\text{SUM}', '\text{AVG}', '\text{COUNT}', '\text{MIN}', '\text{MAX}', '\text{AND}', '\text{OR}', \ldots\}$ (aggregation/grouping operator).
  - Statement $(m_j, d_i, \Omega) \in S$ declares that measure $m_j$ can be aggregated along dimension $d_i$ by means of the operator $\Omega$.
  - If no aggregation statement exists for a given pair $(m_j, d_i)$, then $m_j$ cannot be aggregated at all along $d_i$. 
DFM: Aggregation

- A measure is **additive** on a dimension if its values can be aggregated along the corresponding hierarchy by the 'SUM' operator.

- Graphical representation:
  - If \((m_j, d_i, 'SUM') \notin S\), \(m_j\) and \(d_i\) are connected by a dashed line labelled with all aggregation operators \(\Omega\) such that \((m_j, d_i, \Omega) \in S\).
  - If \((m_j, d_i, 'SUM') \in S\):
    - If \(\forall \omega \neq 'SUM' \mid (m_j, d_i, \omega) \in S\), \(m_j\) and \(d_i\) are not graphically connected.
    - Otherwise, \(m_j\) and \(d_i\) are connected by a dashed line labelled with the symbol '+' followed by all the other operators \(\omega \neq 'SUM'\) such that \((m_j, d_i, \omega) \in S\).
SALE Fact Schema

- **SALE**
  - qty sold
  - revenue
  - no. of customers

- **Product**
  - brand
  - type
  - city
  - weight
  - diet

- **Manager**
  -营销组
  -部门

- **Sales**
  -销售经理
  -销售区域

- **Store**
  -城市
  -县
  -地址
  -电话

- **Promotion**
  -促销
  -价格减免
  -广告类型
  -成本

- **Date**
  -年
  -季度
  -月
  -日
  -周
  -假期
  -季节

- **Begin Date**
- **End Date**
INVENTORY Fact Schema

- INVENTORY
  - qty
  - MIN
  - AVG
  - store
    - city
    - state
  - address
  - season
  - year
  - month
  - week
  - product
    - type
    - brand
    - category
    - units per pallet
    - weight
    - package size
    - package type
starER

- Combines the star structure with the semantically rich constructs of the ER model.
- Adds special types of relationships to support hierarchies.
- starER vs. DFM:
  - starER allows many-to-many relationships between dimensions and facts.
  - starER allows objects participating in the data warehouse, but not in the form of a dimension.
  - Specialized relationships on dimensions are permitted in starER (specialization/generalization).

Source: [TBC99]
starER: Mortgage Company Data Warehouse
UML Profile for Multidimensional Modeling

- **UML profile:** extend UML by
  - stereotypes represented by icons e.g. 📊 🗑️ 📊
  - tagged values e.g., {name = value}
  - constraints, e.g., {Quantity Is Not SUM Along Salesperson}

**Main characteristics:**
- **Accurate**
  - major features of multidimensional models supported
- **No redundancy**
  - concepts and elements are only defined once in the model and imported where needed
- **Simplicity**
  - minimal subset of UML and minimal extensions
- **Understandable**
  - define three levels of abstraction
  - use packages as grouping mechanism
  - support conformed dimensions

Source: [LTS06]
UML Profile: Overview

Stereotype icons of packages

- StarPackage
- DimensionPackage
- FactPackage

Stereotype icons of classes

- Fact
- Dimension
- Base
- Degenerate Fact

Stereotype icons of attributes

- DD: Degenerate Dimension
- OID: OID
- FA: Fact Attribute
- D: Descriptor
- DA: Dimension Attribute
Three Levels of Detail

Level 1: Model definition
- A package represents a star schema
- A dependency between two packages indicates that star schemas share at least one dimension

Level 2: Star schema definition
- A package represents a fact or a dimension
- A dependency between two packages indicates that they share at least one level of the dimension hierarchy

Level 3: Dimension Definition
- Set of classes representing the hierarchy levels, or
- Set of classes representing the entire star schema
Running Example

• Company having several dealerships that sell automobiles (cars and vans) across several states
• Data warehouse should support analysis of
  ▪ sales of automobiles
  ▪ sales of parts such as spare wheels or light bulbs
  ▪ service works such as change of lubricating oil or brake oil
Level 1: Model Definition

- **Auto-sales schema**: contains the sales of automobiles.
- **Parts schema**: contains the sales of parts, such as spare wheels or light bulbs.
- **Services schema**: contains the services provided, such as the change of the lubricating oil or the brake fluid.

**Stereotype icons**

- **StarPackage**
- **DimensionPackage**
- **FactPackage**
Level 2: Auto-Sales Schema

Data Warehouse Design

- Dealership dimension
- Customer dimension
- Auto dimension
- Salesperson dimension
- Time dimension

Stereotype icons of packages

StarPackage

DimensionPackage

FactPackage
They share some hierarchy levels: City, Region, and State

Level 2: Services Schema

Services fact

Dealership dimension

Customer dimension

Parts dimension

Service dimension

Mechanic dimension

Time dimension
Dimension Hierarchies

- Classification hierarchy levels are specified by base classes

- Associations show hierarchical relationship
  - drill-down direction is marked by +d
  - roll-up direction is marked by +r
  - default roll-up or drill-down directions are marked by arrows

- Multiplicity may be defined for each association

- Various attribute types:
  - DA: dimension attribute
  - D: description
  - OID: object id
Level 3: Customer Dimension

Customer personal data:
- Fullname
- Name
- Surname
- BornDate

Fullname is concatenation of Name and Surname
Types of Classification Hierarchies

- **Strict hierarchy:**
  Each object of the lower level (+d) belongs to only one object at the higher level (+r).

- **Non-strict hierarchy:**
  An object of the lower level (+d) may belong to several objects at the higher level (+r).

- **Completeness for drill-down:**
  For each object at the higher level there exists an object at the lower level.

- **Completeness for roll-up:**
  For each object at the lower level there exists an object at the higher level.

Non-strict and incomplete hierarchies may yield to inconsistent totals!
Summarizability Issues

- **Drill-Down**
  - Country (B) ➔ Region (B)
    - +r
    - +d
    - 0..*  
    - Germany 50  
    - France 40  
    - Andorra 15  
  - Total = 105

- **Roll-up**
  - Category (B) ➔ Product (B)
    - +r
    - +d
    - 0..1  
    - Dairy 55  
    - Bakery 25  
  - Total = 80

- **Non-strict**
  - Season (B) ➔ Month (B)
    - +r
    - +d
    - 1..*  
    - Winter 560  
    - Spring 500  
    - Summer 100  
  - Total = 1160

- **Country Sales**
  - Germany 50  
  - France 40  
  - Andorra 15  
  - Total = 105

- **Region Sales**
  - NRW 20  
  - BW 10  
  - BAY 20  
  - Paris 30  
  - Province 10  
  - Total = 90

- **Category Sales**
  - Dairy 55  
  - Bakery 25  
  - Total = 80

- **Product Sales**
  - Milk 15  
  - Butter 10  
  - Bread 25  
  - Yogurt 30  
  - Glasses 100  
  - Total = 180

- **Month Sales**
  - Jan 200  
  - Feb 180  
  - Mar 180  
  - Apr 160  
  - May 160  
  - Jun 100  
  - Total = 980
Categorization Hierarchies

- Allows to model additional features for subtypes of a class, e.g., cars and vans have some common and some specific attributes.
Level 3: Auto-Sales Fact

- Many-to-many-Relationships between facts and dimensions
  - Several salespersons for a sale

- Summarizability is defined as a constraint

- Degenerate Fact:
  - for m:n-relationships between facts and dimensions
  - specific attribute that is provided for each instance combination in such a relationship

- Degenerate Dimension:
  - Most of the properties are already presented by other elements (facts, dimensions)
  - Remaining attributes are necessary to uniquely identify fact instances.
Important Features of a Conceptual Model

- Explicit hierarchies in dimensions
- Multiple hierarchies in a dimension
- Non-strict and non-complete hierarchies
- Conformed dimensions

- Support correct aggregation (summarizability)
- Degenerated facts
- Degenerated dimensions

- Many-to-many relationships between facts and dimensions
- Symmetric treatment of dimensions and measures

- Changes of data over time
- Reflect uncertainty
- Different levels of granularity

Sources: [PJ99, LTS06]
Overview

• Data Warehouse Design Process
• Conceptual Design

• Logical Design
  ▪ TPCH Benchmark Schema
  ▪ Star Schema
  ▪ Snowflake Schema
  ▪ Informix Schema

• Details of Logical Design
• Physical Design
Logical Design

Convert the conceptual schema to a logical one with respect to the target logical data model.

• Logical Design is based on:
  ▪ conceptual diagrams
  ▪ summarizability constraints
  ▪ transformation rules

• Logical data models:
  ▪ relational
  ▪ multidimensional
Sample Scenario

- Company that manages, sells, or distributes a product worldwide (e.g., car rental, food distribution, parts, suppliers, etc.).
- Scenario taken from TPC-H benchmark

- Each order entry refers to a part, an order, a customer, a supplier and the dates for the ordering, the commitment, the shipment and the receipt.
- Order entries are further characterized by a line number, a status, shipment instructions and a return flag.
Conceptual Schema: Overview
Logical Schema Types

- **Star Schema**
  - Each dimension is modeled by a single table
  - Redundancy within the dimension tables, e.g. nationname, regionname
  - Qualities are combined into one or a few ‘junk’ dimensions

- **Snowflake Schema**
  - Dimensions get normalized (3NF)
  - One dimension table per hierarchy level

- **Informix Schema**
  - Normalized attribute tables
  - Hierarchies are partly represented in dimension tables
Informix Schema

**part**
- Attribute Table
  - partkey
  - name
  - mfg
  - brand
  - type
  - size
  - container

**shipdate**
- Attribute Table
  - shipdatekey
  - shipmonth
  - shipyear

**shipdate**
- Dimension Table
  - shipdatekey
  - shipmonthkey
  - shipyearkey

**shipyear**
- Attribute Table
  - shipyearkey
  - shipyearname

**shipmonth**
- Attribute Table
  - shipmonthkey
  - shipmonthname

**commitdate**
- Dimension Table
  - commitdatekey

**order**
- Attribute Table
  - orderkey
  - orderstatus
  - orderpriority
  - shippriority
  - clerk

**lineitem_orders**
- Fact Table
  - orderkey
  - partkey
  - suppkey
  - custkey
  - shipdatekey
  - commitdatekey
  - receiptdatekey
  - orderdatekey
  - junkkey
  - linenumber
  - quantity
  - extendedprice
  - endprice
  - discount
  - tax

**supplier**
- Dimension Table
  - suppkey
  - suppnationkey
  - suppregionkey

**suppregion**
- Attribute Table
  - suppregionkey
  - suppregionname

**suppnation**
- Attribute Table
  - suppnationkey
  - suppnationname

**customer**
- Attribute Table
  - custkey
  - custnationkey
  - custregionkey
  - mktsegment

**custregion**
- Attribute Table
  - custregionkey
  - custregionname

**mktsegment**
- Attribute Table
  - mktsegmentkey
  - name

**custnation**
- Attribute Table
  - custnationkey
  - custnationname

**receiptdate**
- Dimension Table
  - receiptdatekey
Comparing Logical Schema Types

- Is all the information of the conceptual model representable in the logical schema?
  - Explicit hierarchies in dimensions
  - Multiple hierarchies in a dimension
  - Non-strict and non-complete hierarchies
  - Many-to-many relationships between facts and dimensions
  - Symmetric treatment of dimensions and measures
  - Support correct aggregation (summarizability)

- What effort is needed to reflect changes of the conceptual design in the logical schema:
  - Insert hierarchy level
  - Delete hierarchy level
  - Insert measure
  - Delete measure
  - Insert dimension
  - Delete dimension
  - Modify granularity

arrow logical schema?

arrow metadata?
## Comparison

<table>
<thead>
<tr>
<th></th>
<th><strong>Star Schema</strong></th>
<th><strong>Snowflake Schema</strong></th>
<th><strong>Informix Schema</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Clearness</strong></td>
<td>one table per dimension</td>
<td>multiple tables per dimension</td>
<td>multiple tables per dimension</td>
</tr>
<tr>
<td><strong>Redundancy</strong></td>
<td>in the dimension table</td>
<td>normalization takes care of</td>
<td>normalization takes care of</td>
</tr>
<tr>
<td><strong>Data Volume</strong></td>
<td>high(er)</td>
<td>low</td>
<td>low</td>
</tr>
<tr>
<td><strong>Hierarchies</strong></td>
<td>not represented</td>
<td>represented in dimension tables</td>
<td>only one hierarchy</td>
</tr>
<tr>
<td><strong>Summarizability</strong></td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>Adding a Dimension</strong></td>
<td>one additional table</td>
<td>several additional tables</td>
<td>several additional tables</td>
</tr>
<tr>
<td><strong>Adding an Attribute</strong></td>
<td>one attribute appended to dimension table</td>
<td>changes to several tables</td>
<td>changes to several tables</td>
</tr>
<tr>
<td><strong>Performance (Queries)</strong></td>
<td>max. one join per dimension</td>
<td>several joins among the dimension tables</td>
<td>several joins among the dimension tables</td>
</tr>
</tbody>
</table>
Overview

- Data Warehouse Design Process
- Conceptual Design
- Logical Design

Details of Logical Design
- Extended Dimension Table Design
  - Production keys / surrogate keys
  - Roles of Dimensions
  - Hierarchies
  - Slowly changing dimensions
  - Time stamping in large dimensions
  - Large dimensions with frequent changes
  - Many-to-Many Dimensions
  - Time Dimensions
- Extended Fact Table Design
  - Modeling Events and Coverage (Factless Fact Tables)
  - Multinational Currency Tracking

Physical Design
Production Keys / Surrogate Keys

- **Production key**: Attributes that make up a primary key in the production system (source system).
  - Keys may be reused in the production system.
  - Rules for building production keys may change.
  - Production keys are kept although the dimension has changed.

- **Surrogate key**: Single attribute (INTEGER) that is used as primary key in the data warehouse.
  - Results in small keys.
  - Surrogate keys have no meaning.
  - Keys may be changed independent of the source system (this is especially important for type two slowly changing dimensions)
Roles of Dimensions

• In some situations a single dimension appears several times in the same fact table.

• **Example:**
  A fact table on customer orders may include:
  - Order date
  - Packaging date
  - Shipping date
  - Delivery date
  - Payment date
  - Return date
  - Order status
  - Customer
  - ...

  time in several roles

• **Problems:**
  - Using the same dimension for each role may result in wrong query results.
  - Result may contain many columns with identical names.
Single Dimension Tables for all Roles

SELECT ...  
FROM Orders AS O,  
      Time AS T  
WHERE O.OrderDate = T.TimeID  
AND O.PackagingDate = T.TimeID  
AND O.ShippingDate = T.TimeID  
AND O.OrderStatus = 'completed'  
GROUP BY ...  

SELECT ...  
FROM Orders AS O,  
      Time AS OT,  
      Time AS PT,  
      Time AS ST  
WHERE O.OrderDate = OT.TimeID  
AND O.PackagingDate = PT.TimeID  
AND O.ShippingDate = ST.TimeID  
AND O.OrderStatus = 'completed'  
GROUP BY ...
Single Dimension Table for each Role

Role-playing dimension tables can be provided as physical tables or as views.
Hierarchies

- Organization hierarchies
- Parts explosion hierarchies

- Compact way to represent the hierarchy.
- Structure results in complex queries.

- Goals:
  - Keep the original grain of the customer dimension.
  - Support aggregation for the entire hierarchy.
  - Support aggregation for the immediate subsidiaries as well as for the lowest-level subsidiaries.
  - Support the efficient search for the immediate parent or the top-most parent.
Bridge Table for Hierarchies

- Dimension table and fact table are kept.
- Bridge table contains:
  - All combinations of CustomerKeys and all their SubsidiaryCustomers
  - Path-length of all combinations including paths of length zero.
  - A flag identifying the bottom level of the hierarchy.
Usage of Bridge Tables

- Aggregation ignoring the customer hierarchy.

- Descending the customer hierarchy.

- Ascending the customer hierarchy.
Slowly Changing Dimensions

- **Example:** The description of a product changes.
- **Modeling alternatives:**
  - **Type One:** Substitution of the old description by the new one.
  - **Type Two:** Product together with its new description gets a new ID.
  - **Type Three:** Associated product entry has an additional attribute that holds the old description.
# Slowly Changing Dimensions

<table>
<thead>
<tr>
<th></th>
<th>Type One</th>
<th>Type Two</th>
<th>Type Three</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keeps history</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Schema changed</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Volume of data changes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Effect on queries</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Usage</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Time Stamping in Large Dimensions

• **Example:** A human resource department has to store many attributes on many employees and precisely track all the changes on these attributes.

• Reports that should be provided:
  - R1: Summary status of the entire employee base on a regular (monthly) basis.
  - R2: Profile the employee population at any precise instant of time.
  - R3: Provide every action taken on a given employee, with the correct transaction sequence and the correct timing of the transaction.

• Slowly changing dimensions, type 2:
  - R1, R2: Does not provide the valid information for an employee for a certain point in time.
  - R3: Does not track the transaction time for actions taken on employees.
Human Resource Environment

- Employee transaction table contains a complete snapshot of the employee record for each transaction.
- Human resource table contains the regular facts stored for each employee each month.
- Which tables are needed for reports R1, R2 and R3?
Large Dimensions with Frequent Changes

• **Example:**
  - For an insurance company the customer dimension might hold millions of tuples; large amounts of demographic data are kept for each customer.
  - Customer data change constantly and these changes have to be propagated into the warehouse.

<table>
<thead>
<tr>
<th>Customer Dimension Table</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>custkey</strong></td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>original_address</td>
</tr>
<tr>
<td>date_of_birth</td>
</tr>
<tr>
<td>first_order_date</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>income</td>
</tr>
<tr>
<td>education</td>
</tr>
<tr>
<td>number_children</td>
</tr>
<tr>
<td>marital_status</td>
</tr>
<tr>
<td>credit_score</td>
</tr>
<tr>
<td>purchase_score</td>
</tr>
</tbody>
</table>
Large Dimensions with Frequent Changes

- Design goals for this kind of dimensions:
  - Support rapid browsing, e.g. of low cardinality attributes.
  - Support efficient browsing of cross-constrained values in the dimension table.
  - Do not penalize the fact table query for using a large dimension.
  - Find and suppress duplicate entries in the dimension.
  - Do not create additional records to handle the changing dimension problem.
Large Dimensions with Frequent Changes

- Modeling approach:
  Break off dimension into several dimension tables.
Large Dimensions with Frequent Changes

• One (or several) additional dimension table (Demographics) includes all changing attributes.
• Attributes in this new dimension are forced to have a relatively small number of discrete values:
  - e.g., income is mapped to income bands
• The additional dimension table
  - is populated with all possible discrete attribute combinations, and
  - comes with its own surrogate key.
• The surrogate key of the additional dimension is used in the fact table.
• Changes in the demographics dimension are reflected by entries in the fact table.
Many-to-Many Dimensions

- Dimensions may have zero, one or many values for a given fact record.
- The number of dimension values is not known in advance.
- **Example:**
  A fact table record for each line item of a hospital bill or each line item of a treatment performed in a physician's office. The design has to cover several diagnoses per patient.
Bridge Tables

- Bridge table between the fact table and the dimension table.
- Use a special diagnosis group key in the fact table.
- The diagnosis group table contains a set of records for each patient. Each record points to a specific diagnosis and provides a weighting factor.
- Over time a patient may have different diagnosis groups.

Questions that can be answered:
- Correctly weighted summary of all charges grouped by diagnosis.
- Impact report that totals the impact each diagnosis has in terms of total amounts associated with that diagnosis.
Time Dimension

- Some transactions need a fine-scale tracking to the minute or even the second.
- Solution 1:
  - Time dimension with one record for every minute or second.
- Solution 2:
  - Time dimension on a daily basis.
  - Transaction time is treated as a fact.
  - Timestamp may be provided for various time zones.

<table>
<thead>
<tr>
<th>Sales Fact Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>DateTimeKey</td>
</tr>
<tr>
<td>ProductKey</td>
</tr>
<tr>
<td>StoreKey</td>
</tr>
<tr>
<td>CustomerKey</td>
</tr>
<tr>
<td>ClerkKey</td>
</tr>
<tr>
<td>RegisterKey</td>
</tr>
<tr>
<td>PromotionKey</td>
</tr>
<tr>
<td>DollarsSold</td>
</tr>
<tr>
<td>UnitsSold</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sales Fact Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>DateKey</td>
</tr>
<tr>
<td>ProductKey</td>
</tr>
<tr>
<td>StoreKey</td>
</tr>
<tr>
<td>CustomerKey</td>
</tr>
<tr>
<td>ClerkKey</td>
</tr>
<tr>
<td>RegisterKey</td>
</tr>
<tr>
<td>PromotionKey</td>
</tr>
<tr>
<td>TimeOfDay</td>
</tr>
<tr>
<td>GMTTimeOfDay</td>
</tr>
<tr>
<td>DollarsSold</td>
</tr>
<tr>
<td>UnitsSold</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>
Various Units of Measure

- Several business processes monitor the flow of products or the inventory. For some of these processes numbers should be expressed in different units of measure.

- Example:
  In the value chain several quantity facts have five different unit-of-measure interpretations and four valuation schemas.

- Solution 1:
  - The unit-of-measure interpretations as well as the valuation schemas are presented in the dimension table.
Various Units of Measure

- **Solution 2:**
  - The unit-of-measure interpretations as well as the valuation schemas are provided in the fact table.

- **Advantages of this solution:**
  - Eliminates the possibility of choosing the wrong factors.
  - Changes in factors do not lead to changes in the dimension table.
Modeling Events and Coverage

• How to model events?
• **Example:**
  - A fact table should express the fact that students participate in courses and exercises (student’s attendance event).
• What are the dimensions? What are the facts?
• Dimensions: Time, Student, Course, Teacher, Facility
• No obvious fact: Combination of all relevant keys express the fact.

• How to model coverage?
• **Example:**
  - A fact table should record the sales of products in stores on particular days under each promotion condition.
• Sales fact table may be sparse. How to handle products that did not sell?
• Dimensions: Time, Store, Product, Promotion
• No obvious fact: Combination of all relevant keys express the fact.
Factless Fact Tables

- Dummy fact 'attendance' is optional but makes SQL more readable.
- Many questions can be answered:
  - Which classes were the most heavily attended?
  - Which classes were the most consistently attended?
  - Which teachers taught the most students?
  - Which teachers taught classes in facilities belonging to other departments?
  - Which facilities where the most lightly used?
Factless Fact Tables

- Answering the question of which products were on promotion but did not sell requires a two-step application.
  - first, consult coverage table
  - second, consult sales table
- Use sales fact table instead and fill in records representing zero sales for all possible products?
  - The coverage factless fact table can be made much smaller.
Multinational Currency Tracking

- Transactions have to be expressed in a multitude of currencies.
- Solution:
  - Value is reported in the local currency as well as in the global currency for a multinational enterprise (e.g. US dollars).
  - Conversion table provides conversion rates on a daily basis in both directions.
Overview

• Data Warehouse Design Process
• Conceptual Design
• Logical Design
• Details of Logical Design
• Physical Design
Physical Design

Physical implementation of the logical schemata with respect to the individual properties of the target database system.

- What to consider in physical design:
  - indexing
  - partitioning
  - denormalization
  - pre-aggregation
  - ...

- See also 'Database Support for Data Warehousing'
Summary

- Main steps of the Data Warehouse Design Process:
  - req. analysis, conceptual, logical and physical design
- Different conceptual models are available. Common elements:
  - attributes, dimensions, hierarchies, facts, measures
- Conceptual models usually provide a graphical notation.
- Logical design is based on the multidimensional model or the relational model.
- Conceptual schema may be mapped to different logical schema types (relational model):
  - star schema, snowflake schema, vendor-specific schemes
- Physical design deals with optimization steps for the target database system.
Papers


Appendix: starER

- Combines the star structure with the semantically rich constructs of the ER model.
- Adds special types of relationships to support hierarchies.
- starER vs. DFM:
  - starER allows many-to-many relationships between dimensions and facts.
  - starER allows objects participating in the data warehouse, but not in the form of a dimension.
  - Specialized relationships on dimensions are permitted in starER (specialization/generalization).

Source: [TBC99]
Appendix: starER: Fact Sets

- **Fact set:**
  - Represents a set of real-world facts sharing the same characteristics or properties.
  - Semantically, a fact set points to the process of generating data over time, i.e., data is generated in terms of facts, each time an event related to the fact takes place.

- **Graphical representation:**

  repayment  

  ![Graphical representation of repayment](image-url)
Appendix: starER: Fact Properties

- **Fact properties:**
  - Properties that characterize a fact.
  - Usually numerical data that can be summarized.
  - There are three different types of properties:
    - **stock (S):** records the state of something at a specific point in time.
    - **flow (F):** records the commutative effect over a period of time.
    - **value-per-unit (V):** like 'stock', but the unit of the property is different.

- **Graphical representation:**

![Graphical representation of repayment and installment properties](image)
Appendix: starER: Entity sets and relationship sets

- **Entity sets:**
  - Represent a set of real-world objects with similar properties.
  - It has the same meaning as in traditional application modeling.

- **Relationship sets:**
  - Represents a set of associations among entity sets or among entity sets and fact sets.
  - Its cardinality can be many-to-many, many-to-one or one-to-many.

- **Attributes:**
  - Static properties of entity sets, relationship sets, and fact sets.

- **Graphical representation:**
  - ![Customer has relationship with address](image1)
  - ![Customer pays relationship with repayment](image2)
Appendix: starER: Relationships

- Specialization / generalization

- Aggregation

- Complete membership

- Non-complete membership

- Strict membership
Appendix: starER: Dimensions and Hierarchies

- **Dimension:**
  - Entity sets associated to a fact are the dimensions of that fact.

- **Hierarchies:**
  - Dimensions consist of hierarchies and other relationships among other entity sets.

```
repaayment payback loan
             \________\       /__________\
            /          \              /
           /            \            /    \\
           \            \            /     \\
            \          \          /      \\
             \        \        /       \\
              \      \      /        \\
               \    \    /         \\
                \  \  /           \\
                 \ \ /             \\
                  \//               \\
                    loan id         loan type
```

```
week -> day -> month -> quarter -> year
```

```
time dimension
```

```
loan dimension
```
Appendix: Mortgage Company DW