Acknowledgements: I am indebted to Michael Böhlen and Stefano Rizzi for providing me their slides, upon which these lecture notes are based.
Outline

1. The Grocery Store Example
2. More about Multidimensional Modeling
3. Inventory Management Example
4. Order Management
5. OncoNet
6. MEDAN
Outline

1. The Grocery Store Example
2. More about Multidimensional Modeling
3. Inventory Management Example
4. Order Management
5. OncoNet
6. MEDAN
A grocery chain with 500 stores spread over a five-state area.
- Each of the stores is a typical modern supermarket with a full complement of departments including grocery, frozen foods, dairy, meat, bakery, hard goods, liquor, and drugs.
- Each store has roughly 60,000 individual products on its shelves.

The individual products are called stock keeping units (SKUs).

About 40,000 of the SKU come from outside manufacturers and have bar codes imprinted on the product package.
- These bar codes are called universal product codes (UPCs).
- UPCs are at the same grain as individual SKUs.
- Each different package variation of a product has a separate UPC and hence is a separate SKU.

The remaining 20,000 SKUs come from departments like meat or bakery departments and do not have nationally recognized UPC codes.
- The grocery store assigns SKU numbers to these products by sticking scanner labels on the items.
- Although the bar codes are not UPCs they are certainly SKU numbers.
Data is collected at several places in a grocery store.

- Some of the most useful data is collected at the cash registers as customers purchase products.
- Our modern grocery store scans the bar codes directly into the point-of-sale (POS) system.
- The POS system is at the front door of the grocery store where customer takeaway is measured.
- The back door, where vendors make deliveries, is another interesting data-collection point.
At the grocery store, management is concerned with the logistics of ordering, stocking the shelves, and selling the products while maximizing the profit at each store.

The profit ultimately comes from:
- charging as much as possible for each product,
- lowering costs for product acquisition and overhead, and
- at the same time attracting as many customers as possible.

The most significant decisions have to do with pricing and promotions:
- Both store management and headquarters marketing spend a great deal of time tinkering with pricing and running promotions.
- Promotions in a grocery store include temporary price reductions, ads in newspapers and newspaper inserts, displays in the grocery store, and coupons.
A somehow simplified DM design process consists of the following 4 steps:

1. Choose the business process(es) to model
   - e.g., Sales,

2. Choose the granularity of the business process
   - e.g., Items by Store by Promotion by Day.
   - Low granularity is needed.
   - Are individual transactions necessary/feasible?

3. Choose the dimensions
   - Time, Store, . . .

4. Choose the measures
   - Dollar_sales, unit_sales, dollar_cost, customer_count
Step 1: Choose the Business Process

- A **business process** is an activity in the organization that typically is supported by a source data management system
  - raw material purchasing, orders, shipments, invoicing, inventory, bank transfers, patient transfers, . . .
- Business processes are not necessarily limited to a single department
  - e.g., sales and marketing departments might be interested in the orders
- **Focusing on the business process** rather than the department avoids duplication of work and keeps data more consistent
- The **first dimensional model** built should be the one with the **most impact**
- It should answer the **most pressing business questions** and be readily accessible for data extraction
Step 1: Choose the Business Process – Example

- Management wants to better understand customer purchases as captured by the POS system.
- Business process: POS retail sales
- Allows us to analyze:
  - What products are selling?
  - In which stores?
  - On what days?
  - Under what promotional conditions?
  - etc.
Step 2: Choose the Grain of the Business Process

- Preferably develop dimensional models for the most atomic information captured by a business process
  - Not because queries report individual rows, but queries need to cut through the details in very precise ways
- The more detailed/atomic data is, the more things we know
- Atomic data provides maximum analytic flexibility
- Can be constrained and rolled up in every possible way
- It is always possible to declare higher-level grains by aggregation of atomic data; the opposite is not true
- Less granular model is vulnerable to unexpected requests for more details
- Example grain declarations
  - Individual line item on a customer’s sales ticket as measured by a scanner
  - An individual boarding pass of a flight
  - A monthly snapshot for each bank account
Step 2: Choose the Grain of the Business Process – Example

- Individual line item on a POS transaction is the most detailed data, and we choose this as grain (event fact)
- Allows a very detailed analysis of sales
  - Difference in sales on Monday vs. Sunday
  - Is it worthwhile to stock so many individual sizes of certain brands?
  - How many shoppers took advantage of the 50-cents-off promotion on shampoo?
  - Impact in terms of increasing sales when a competitive diet soda product was promoted
  - etc.
- Note that none of these queries calls for data from a specific transaction, but require detailed ways to slice data
  - Could not be answered if only aggregated values would be stored, e.g., daily summaries
Step 3: Choose the Dimensions

- Dimensions can be derived by answering the question “How do business people describe the data resulting from the business processes?”
- “Decorate” fact tables with dimensions representing all possible descriptions of the facts/measures.
- A clear grain statement helps to identify the dimensions.
- Sometimes a revision of step 2 is required.
The Grocercty Store Example

Step 3: Choose the Dimensions – Example/1

- The **Date** dimension
  - Explicit date dimension is needed (events, holidays, ... )

- The **Product** dimension
  - Hierarchy allows drill-down/roll-up through category, brand, department, etc.
  - Many descriptive attributes (often more than 50)

- The **Store** dimension
  - Primary geographic dimension to specify location of the store
  - Many descriptive attributes

- The **Promotion** dimension
  - Used to see if promotions work and are profitable
  - Ads, price reductions, end-of-sale displays, coupons
    - Highly correlated (only 5000 combinations)
Step 3: Choose the Dimensions – Example/2

- Preliminary version of grocery store schema

**Date dimension**
- DateKey (PK)
- FullDateDesc
- DayOfWeek
- Month
- Year
- Quartal

**Product Dimension**
- ProductKey (PK)
- ProductDesc
- BrandDesc
- CategoryDesc
- DepartmentDesc

**POS Retail Sales Transaction Facts**
- DateKey (FK)
- ProductKey (FK)
- StoreKey (FK)
- PromotionKey (FK)
- Measures TBD

**Store Dimension**
- StoreKey (PK)
- StoreName
- StoreStreetAddress
- StoreCity
- StoreCountry

**Promotion Dimension**
- PromotionKey (PK)
- PromotionName
- PriceReductionType
- PromotionMediaType
- AdType

Step 4: Choose the Measures

- Identify the **business performance measures** for the selected processes
- Determined by answering *"What are we measuring?"*
- Measures are determined by the grain declaration
- Revision of the grain might be required
Step 4: Choose the Measures – Example

- Sales quantity
- Sales dollar amount
- Cost dollar amount
- Gross profit
  - Equals to sales dollar − cost dollar
  - Explicit storage avoids user errors

- All measures are additive
- Facts are classified as event facts since each POS transaction is stored (most detailed level)
MD Schema of the Grocery Store Example

**Date dimension**
- DateKey (PK)
- FullDateDesc
- DayOfWeek
- Month
- Year
- Quartal

**POS Retail Sales Transaction Facts**
- DateKey (FK)
- ProductKey (FK)
- StoreKey (FK)
- PromotionKey (FK)
- SalesQuantity
- SalesDollarAmount
- CostDollarAmount
- GrossProfitDollarAmount

**Product Dimension**
- ProductKey (PK)
- ProductDesc
- BrandDesc
- CategoryDesc
- DepartmentDesc

**Store Dimension**
- StoreKey (PK)
- StoreName
- StoreStreetAddress
- StoreCity
- StoreCountry

**Promotion Dimension**
- PromotionKey (PK)
- PromotionName
- PriceReductionType
- PromotionMediaType
- AdType
Outline

1. The Grocery Store Example
2. More about Multidimensional Modeling
3. Inventory Management Example
4. Order Management
5. OncoNet
6. MEDAN
Database Sizing

- Time dimension: 2 years = 730 days
- Store dimension: 300 stores reporting each day
- Product dimension: 30,000 products, only 3,000 sell per day
- Promotion dimension: 5,000 combinations, but a product only appears in one combination per day
- Number of fact records: \(730 \times 300 \times 3,000 \times 1 = 657,000,000\)
- Number of fields: 4 key + 4 measures = 8 fields
- Total DB size: \(657,000,000 \times 8 \text{ fields} \times 4 \text{ bytes} = 21 \text{ GB}\)
- Small database by today’s standards!
Date Dimension

- Date dimension is present in all DWs
- Can be created in advance
- “Meaningful” values are important for report generation, etc.
  - e.g., Holiday/Nonholiday vs. Yes/No
- Time-of-day a separate dimension
  - Separation keeps both dimensions small
- Date dimension vs. SQL date type
  - Many date attributes are not supported in SQL, e.g., fiscal month
  - Business user is not versed in SQL
- Date dimension is relatively small
  - 10 years = 3,650 rows
### Instance of Date Dimension

#### Date Dimension Table

<table>
<thead>
<tr>
<th>DK</th>
<th>Date</th>
<th>FullDateDescription</th>
<th>DayOfWeek</th>
<th>DayNum</th>
<th>HolidayInd</th>
<th>WeekdayInd</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>29.09.2013</td>
<td>September 29, 2013</td>
<td>Sunday</td>
<td>29</td>
<td>Nonholiday</td>
<td>Nonweekday</td>
</tr>
<tr>
<td>2</td>
<td>30.09.2013</td>
<td>September 30, 2013</td>
<td>Monday</td>
<td>30</td>
<td>Nonholiday</td>
<td>Weekday</td>
</tr>
<tr>
<td>3</td>
<td>01.10.2013</td>
<td>October 1, 2013</td>
<td>Tuesday</td>
<td>1</td>
<td>Nonholiday</td>
<td>Weekday</td>
</tr>
<tr>
<td>4</td>
<td>02.10.2013</td>
<td>October 2, 2013</td>
<td>Wednesday</td>
<td>2</td>
<td>Nonholiday</td>
<td>Weekday</td>
</tr>
</tbody>
</table>
Product Dimension

- Description of the products
- >50 attributes is typical for Product dimension
- Concept hierarchy
  - SKU → Brand → Category → Department
  - Many repetitions, but space of dimensions is not critical

Instance of Product dimension table

<table>
<thead>
<tr>
<th>PK</th>
<th>Product Description</th>
<th>Brand Desc</th>
<th>Cat Desc</th>
<th>Dept Desc</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Baked Well Light</td>
<td>Baked Well</td>
<td>Bread</td>
<td>Bakery</td>
</tr>
<tr>
<td>2</td>
<td>Fluffy Sliced Whole Wheat</td>
<td>Fluffy</td>
<td>Bread</td>
<td>Bakery</td>
</tr>
<tr>
<td>3</td>
<td>Fluffy Light Sliced Whole Wheat</td>
<td>Fluffy</td>
<td>Bread</td>
<td>Bakery</td>
</tr>
<tr>
<td>4</td>
<td>Fat Free Mini Cinnamon Rolls</td>
<td>Light</td>
<td>Sweeten Bread</td>
<td>Bakery</td>
</tr>
<tr>
<td>5</td>
<td>Diet Lovers Vanilla 2 Gallon</td>
<td>Coldpack</td>
<td>Frozen Desserts</td>
<td>Frozen Foods</td>
</tr>
<tr>
<td>6</td>
<td>Light and Creamy Butter Pecan 1 Pint</td>
<td>Freshlike</td>
<td>Frozen Desserts</td>
<td>Frozen Foods</td>
</tr>
</tbody>
</table>
Degenerate Dimensions

- **Degenerate dimensions** are “empty”
  - i.e., dimensions with a “hierarchy” of only one attribute
  - Values are directly stored in fact table, no dimension table is needed
- Examples are operational control numbers, e.g., order #, invoice #, POS transaction #, etc.
- Useful to serve as part of primary key in fact table or for grouping
  - e.g., grouping by POS transaction number to retrieve all products purchased in a single transaction

![Diagram of POS Retail Sales Transaction Facts and Dimensions](image)
Junk Dimensions

- **Junk dimension**: a dimension table that stores several degenerate dimensions
  - Within a junk dimension there is no functional dependency (hierarchy)
  - Only feasible if the number of distinct values for the attributes is small

- **Example**: 3 generate dimensions combined in a single junk dimension
How Many Dimensions?

- Dimensions are important to provide detailed information for the analysis!
- But, **too many** dimensions is bad!
  - A sign that dimensions are not independent, and hence should be combined
  - Significantly increases space requirements of fact table
    - Size of dimension table is not a problem
  - 15 dimensions should normally be enough

### POS Retail Sales Transaction Facts

- Date Key (FK)
- Week Key (FK)
- Month Key (FK)
- Quarter Key (FK)
- Year Key (FK)
- Product Key (FK)
- Brand Key (FK)
- Subcategory Key (FK)
- Category Key (FK)
- Store Key (FK)
- Store Country (FK)
- Store State Key (FK)
- PromotionKey (FK)
- SalesQuantity
- SalesDollarAmount
- CostDollarAmount
- GrossProfitDollarAmount
Working with a Dimensional Model/1

- Each dimension attribute is a **rich source** for constructing row headers.
- A common activity is to **drag and drop dimensional attributes** and measures into a simple report (+ specification of aggregate functions for measures).

### POS Retail Sales Transaction Facts
- **DateKey (FK):** DateKey (PK), FullDateDesc, Month, Year, ...
- **ProductKey (FK):** ProductKey (PK), ProductDesc, SKUNumber, BrandDesc, CategoryDesc, DepartmentDesc, ...
- **StoreKey (FK):** StoreKey (PK), Name, District, ZipCode, ...
- **PromotionKey (FK):** PromotionKey (PK), PromotionName, PriceReductionType, PromotionMediaType, ...
- **SalesQuantity:**
- **SalesDollarAmount:**
- **CostDollarAmount:**
- **GrossProfitDollarAmount:**

### Report
<table>
<thead>
<tr>
<th>District</th>
<th>BrandDesc</th>
<th>SalesAmount</th>
<th>QuantitySold</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gries</td>
<td>Kelloggs</td>
<td>3198</td>
<td>1023</td>
</tr>
<tr>
<td>Haslach</td>
<td>Danone</td>
<td>1099</td>
<td>671</td>
</tr>
<tr>
<td>Oswald</td>
<td>Kelloggs</td>
<td>2976</td>
<td>800</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Working with a Dimensional Model/2

- **Drilling down** is adding row headers from a dimension
- **Rolling up** is removing row headers

<table>
<thead>
<tr>
<th>Dept desc</th>
<th>Sales Dollar Amount</th>
<th>Sales Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bakery</td>
<td>$12.331</td>
<td>5088</td>
</tr>
<tr>
<td>Frozen Foods</td>
<td>$31.776</td>
<td>15.565</td>
</tr>
</tbody>
</table>

Roll up on Product dimension

Drill down on Product dimension

<table>
<thead>
<tr>
<th>Dept Desc</th>
<th>Brand Desc</th>
<th>Sales Dollar Amount</th>
<th>Sales Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bakery</td>
<td>Baked Well</td>
<td>$3.009</td>
<td>1.138</td>
</tr>
<tr>
<td>Bakery</td>
<td>Fluffy</td>
<td>$3.024</td>
<td>1.476</td>
</tr>
<tr>
<td>Bakery</td>
<td>Light</td>
<td>$6.298</td>
<td>2.474</td>
</tr>
<tr>
<td>Frozen Foods</td>
<td>Coldpack</td>
<td>$5.321</td>
<td>2640</td>
</tr>
<tr>
<td>Frozen Foods</td>
<td>Freshlike</td>
<td>$10.476</td>
<td>5.234</td>
</tr>
</tbody>
</table>
Surrogate Keys

- **Surrogate keys** are integers that are assigned sequentially in a dimension table, e.g., 1, 2, 3, \ldots
- Should be used instead of natural operational production codes
- Many advantages over operational codes.
  - Make the DW independent from operational changes
    - e.g., re-use of old operational keys after some time
  - Avoid key overlap problem when consolidating data
  - Dimension keys should not contain “intelligence”
    - Should be stored explicitly as additional attribute
  - Performance: Small integer vs. long alpha-numeric code
    - Results in smaller fact tables
    - 1 Byte in a 1 billion fact table translates into 1 GB disk space
Outline

1. The Grocery Store Example
2. More about Multidimensional Modeling
3. Inventory Management Example
4. Order Management
5. OncoNet
6. MEDAN
Consider a large grocery chain with a central warehouse and several retail stores

Advanced retail business requires inventory information

- Making sure the right product is in the right store at the right time minimizes out-of-stocks and reduces overall inventory carrying costs
- The retailer needs the ability to analyze daily quantity-on-hand inventory levels by product and store

Design dimensional models that support the analysis of inventories for retail businesses (grocery stores)
The Value Chain

- The **value chain** identifies the natural, logical flow of an organization’s primary activities
- Provides useful information for the **identification of business processes**
- Operational systems provide snapshots at each step with interesting **data** and **performance metrics**

```
Retailer Issues  
Purchase Order  

Deliveries at Retailer Warehouse

Retailer Warehouse Inventory

Deliveries at Retail Store

Retail Store Inventory

Retail Store Sales
```

ADMT 2018/19 — Unit 6
J. Gamper
Inventory Models

3 different inventory models
- Model 1: Inventory periodic snapshot model
- Model 2: Inventory transactions model
- Model 3: Inventory accumulating snapshot model
Model 1: Inventory Periodic Snapshot: Every day (or at some other regular time interval) the inventory levels of each product is measured and stored as a new row in the fact table.

Example: Inventory of retail store
- Business process: Analysis of retail store inventory
- Granularity: Daily inventory by product at each individual store
- Dimensions: Date, product, and store
- Facts/measures: Quantity on hand
Inventory Periodic Snapshot Model/2

- Inventory generates **dense snapshot tables** (i.e., entry for each product)
  - In contrast, POS Retail Sales table was **sparse**

- Hence, inventory fact table is growing fast!
  - 60,000 products $\times$ 100 stores $= 6$ Mio. rows each time
  - With a row width of 14 bytes, this is 84 MB each time
  - 1 year of daily snapshots would be 30 GB

- Reduce **snapshot frequencies** over time
  - Last 60 days of inventory at daily level
  - Weekly snapshots for older data
  - Instead of 1,095 snapshots in 3 years, only 208 snapshots would be required
The quantity on hand is a **semi-additive measure**

- Can be summarized across products and stores, but not across time
- Different in POS Retail Sales table: Sold entities are counted only once

All measures that record a **static level** (inventory, financial account balance, measures of intensity, e.g., temperature) are inherently **non-additive across time** and possibly other dimensions

- Can be aggregated along time dimension by averaging

A note about SQL AVG function:

- Cannot be used to compute the average over time, since it averages over the number of rows
- Avg inventory over a cluster of 3 products in 4 stores across 7 days would divide the summed value by 84
Model 2: Inventory Transactions: Every transaction that affects the inventory is recorded

Example: Inventory transactions in the store chain
- Receive product
- Place product in to inspection hold
- Release product from inspection hold
- Return product to vendor due to inspection failure
- Place product in bin
- Authorize product for sale
- Pick product for shipment
- Ship product to customer
- Receive product from customer
- Return product to inventory from customer return
- Remove product from inventory
Inventory Transactions Model/2

- Star schema of the inventory transaction model

  - **Date Dimension**
    - DateKey (PK)
    - FullDateDesc
    - ...

  - **Product Dimension**
    - ProductKey (PK)
    - ...

  - **Vendor Dimension**
    - VendorKey (PK)
    - ...

  - **Warhouse Dimension**
    - WarehouseKey (PK)
    - WarehouseName
    - WarehouseAddress
    - WarehouseCity
    - WarehouseZip
    - ...

  - **Store Inventory Snapshot Facts**
    - DateKey (FK)
    - ProductKey (FK)
    - WarehouseKey (FK)
    - VendorKey (FK)
    - InventoryTransactionTypeKey (FK)
    - InventoryTransactionQuantity

- Contains most detailed information, e.g.,
  - How many shipments from a given vendor?
  - On which products more than one round of inspection?
- Reconstruction of exact inventory numbers is possible, but not practical!
- Used in combination with other fact table
Model 3: Inventory Accumulating Snapshot: One row in the fact table for each shipment of a particular product to the warehouse.

Assumption that the inventory goes through a series of events, e.g., receiving, inspection, bin placement, authorization to sell, picking, boxing, and shipping.

A row tracks the disposition of a shipment through these events in the warehouse.

Row is updated as the shipment moves through the warehouse until it leaves the warehouse.

Characterized by many date dimensions and many updates.
Inventory Accumulative Snapshot Model/2

- **Date Received Dimension**
  - DateKey (PK)
  - ...

- **Date Inspected Dimension**
  - DateKey (PK)
  - ...

- **Date Placed in Inventory Dimension**
  - DateKey (PK)
  - ...

- **Date authorized to Sell Dimension**
  - DateKey (PK)
  - ...

- **Date Picked Dimension**
  - DateKey (PK)
  - ...

- **Date Boxed Dimension**
  - DateKey (PK)
  - ...

- **Date Shipped Dimension**
  - DateKey (PK)
  - ...

- **Date of Last Return Dimension**
  - DateKey (PK)
  - ...

- **Warehouse Inventory Accumulating Facts**
  - Date Received Key (FK)
  - Date Inspected Key (FK)
  - Date Placed in Inventory Key (FK)
  - Date Authorized to Sell key (FK)
  - Date Picked Key (FK)
  - Date Boxed Key (FK)
  - Date Shipped Key (FK)
  - Date of Last Return Key (FK)
  - Product Key (FK)
  - Warehouse Key (FK)
  - Vendor Key (FK)
  - Quantity Received
  - Quantity Inspected
  - Quantity Returned to Vendor
  - Quantity Placed in Bin
  - Quantity Authorized to Sell
  - Quantity Picked
  - Quantity Boxed
  - Quantity Shipped
  - Quantity Returned by Customer
  - Quantity Returned to Inventory
  - Quantity Damaged
  - Quantity Lost
  - Unit Cost
  - Unit List Price
  - Unit Average Price
  - Unit Recovery Price

- **Product Dimension**
  - ProductKey (PK)
  - ...

- **Warehouse Dimension**
  - WarehouseKey (PK)
  - ...

- **Vendor Dimension**
  - VendorKey (PK)
  - ...
Integration across the value chain is important for the analysis

- Allows analysis across the business to better evaluate the overall performance (not just at the individual department level)
- End-to-end perspective high-level management to customer

This requires the integration and consistent handling/use of data

**Solution:** Individual fact tables for processes + shared dimensions

**Shared dimensions** are used by different data marts

![Diagram showing relationships between dimensions and facts]
Data Warehouse Bus Architecture

- Data Warehouse Bus Architecture is a standard bus interface that supports the incremental development of a DW
- Based on conformed (similar) dimensions that are shared by the DMs
- Useful tool for the design process as it breaks down the process into small chunks (DMs)
- DMs can be realized at different times and by different groups
Data Warehouse Bus Matrix is a way to document the bus architecture
- Rows represent business processes (translate into DMs)
- Columns represent a suite of standardized, common and shared dimensions

<table>
<thead>
<tr>
<th>Business Processes</th>
<th>Date</th>
<th>Product</th>
<th>Store</th>
<th>Promotion</th>
<th>Warehouse</th>
<th>Vendor</th>
<th>Contract</th>
<th>Shipper</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retail Sales</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Retail Inventory</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Retail Deliveries</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Warehouse Inventory</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Warehouse Deliveries</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Purchase Orders</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
Creating the DW bus matrix is one of the most important up-front deliverables of a DW implementation.

Create a comprehensive list of dimensions before filling in the matrix.

The rows provide a concise overview about the dimensionality of the individual DMs.

The columns show the interaction between the DMs and the common/shared dimensions.
Conformed Dimensions

- **Conformed Dimensions** are either identical or strict mathematical subsets of the most granular, detailed dimension
- **Roll-up dimensions** conform to the base-level dimension
- **Example**: The sales process captures data at the product level, while the forecasting process does it at the brand level
  - Brand table conforms to the atomic product table as it is a strict subset of product table

<table>
<thead>
<tr>
<th>Product Dimension</th>
<th>Brand Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Key (PK)</td>
<td>Brand Key (PK)</td>
</tr>
<tr>
<td>Product Description</td>
<td>Brand Description</td>
</tr>
<tr>
<td>SKU Number</td>
<td>Subcategory Description</td>
</tr>
<tr>
<td>Brand Description</td>
<td>Category Description</td>
</tr>
<tr>
<td>Subcategory Description</td>
<td>Department Description</td>
</tr>
<tr>
<td>Category Description</td>
<td>Package Type Description</td>
</tr>
<tr>
<td>Department Description</td>
<td>Package Size</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>

`conforms`
Outline

1. The Grocery Store Example
2. More about Multidimensional Modeling
3. Inventory Management Example
4. Order Management
5. OncoNet
6. MEDAN
Order management consists of several critical business processes (order, shipment, invoice processing, etc.) and measures (sales volume, invoice revenue, etc.).

Data warehouse bus matrix

<table>
<thead>
<tr>
<th>Business Processes</th>
<th>Date</th>
<th>Product</th>
<th>Customer</th>
<th>Deal</th>
<th>Sales Rep</th>
<th>Ship From</th>
<th>Shipper</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quotes</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Orders</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shipment</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Invoicing</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
Order Management

Star Schema

Order Date Dimension
- OrderDateKey (PK)
- OrderDate
- Order Date Day of Week
  ...

Requested Ship Date Dimension
- Requested Ship Date Key (PK)
- Requested Ship Date
  ...

Order Transaction Facts
- Order Date Key (FK)
- Requested Ship Date Key (FK)
- Product Key (FK)
- Customer Ship To Key (FK)
- Sales Rep Key (FK)
- Deal key (FK)
- Order Number (DD)
- Order Quantity
- Gross Order Dollar Amount
- Order Deal Discount Dollar Amount
- Net Order Dollar Amount

Product Dimension
- ProductKey (PK)

Customer Ship To Dimension
- CustomerShipToKey (PK)
  ...

Sales Rep Dimension
- SalesRepKey (PK)
  ...

Deal Dimension
- DealKey (PK)
  ...

Issues
- Should there be a “Ship Date Key” in the fact table?
- Can/should Order Date Key and Requested Ship Date Key be foreign keys to the same dimension table?
Role-Playing

- **Role-playing** in DW occurs when a single dimension appears several times in the same fact table,
  - e.g., order date and requested ship date
- Should not be FK to the same dimension table
  - SQL would require the two dates to be the same
  - We might want to constrain the two dimensions differently
- The underlying dimension may exist as a single physical table, but each of the roles should be presented in a separate view with different labels

```
<table>
<thead>
<tr>
<th>Date Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>DateKey (PK)</td>
</tr>
<tr>
<td>Date</td>
</tr>
<tr>
<td>Date Day of Week</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>
```

```
create view
```

```
<table>
<thead>
<tr>
<th>Order Date Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order Date Key (PK)</td>
</tr>
<tr>
<td>Order Date</td>
</tr>
<tr>
<td>Order Date Day of Week</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>
```

```
create view
```

```
<table>
<thead>
<tr>
<th>Requested Ship Date Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requested Ship Date Key (PK)</td>
</tr>
<tr>
<td>Requested Ship Date</td>
</tr>
<tr>
<td>Requested Ship Date Day of Week</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>
```
Fact Normalization

- **Fact normalization**: Normalize the fact table and collapse all measures into a single measure along with a special fact dimension that identifies the type of the measure.

- Makes only sense if:
  - fact table is sparsely populated and no computations are made between different measures
  - e.g., medical tests where different parameters are measured and data is sparse

- **Example**: Normalize Order Transaction Fact Table
  - Fact Dimension table has entries for Order Quantity, Gross Order Dollar Amount, Order Deal Discount Dollar Amount, Net Order Dollar Amount

---

### Not normalized

**Order Transaction Facts**
- Order Date Key (FK)
- Requested Ship Date (FK)
- Order Quantity
- Gross Order Dollar Amount
- Order Deal Discount Dollar Amount
- Net Order Dollar Amount

---

### Normalized

**Order Transaction Facts**
- Order Date Key (FK)
- Requested Ship Date (FK)
- ... 

**Fact Dimension**
- Entry Type Key (PK)
- Entry Type
- Unit of Measurement
- Amount

...
Outline

1. The Grocery Store Example
2. More about Multidimensional Modeling
3. Inventory Management Example
4. Order Management
5. OncoNet
6. MEDAN
OncoNet is an application for the management of patients undergoing a cancer therapy.

Collaboration with Hospital Meran (BSc thesis of A. Heinisch)

Cancer therapy follows a treatment plan/protocol, which specifies certain events (medications, tests, etc.) at regular time points.

<table>
<thead>
<tr>
<th>Event type</th>
<th>Event description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data collection</td>
<td>Ct Thorax</td>
<td>Day 60, 100, 360, 720</td>
</tr>
<tr>
<td>Data collection</td>
<td>Ct Abdomen</td>
<td>Day 60, 100, 360, 720</td>
</tr>
<tr>
<td>Data collection</td>
<td>Hemogram</td>
<td>Day 110</td>
</tr>
<tr>
<td>Medication</td>
<td>Zofran</td>
<td>Day 1</td>
</tr>
<tr>
<td>Medication</td>
<td>Adriblastina</td>
<td>Day 1</td>
</tr>
</tbody>
</table>
Business process: Analysis of cancer therapies

Queries to answer:
- How many patients with normal blood pressure after medication X?
- Which dosages of drug A were successful to reduce parameter Y?
- etc.

Granularity: Individual events of the chemotherapy
- Includes measurements, examinations, questionnaires, etc.
Patient and Drug Dimension

### Patient Dimension
- Patient Key (SK)
- Patient ID
- Patient First Name
- Patient Last Name
- Patient Gender
- Patient Address
- Patient ZIP Code
- Patient Phone Number
- Patient Profession
- Patient First Language
- Patient Height
- Patient Weight
- Patient Body Surface Area
- Patient Place of Birth
- Patient Birthday Date Key
- Patient Death Date Key
- Patient First Admission Doctor
- Patient First Admission Area
- Patient Smoking Indicator
- Patient Cigarettes per Day
- Patient Alcohol Indicator
- Patient Alcohol Amount per Day

### Drug Dimension
- Drug Key (SK)
- Drug ID
- Drug Name
- Drug Category
- Drug Active Substance
- Drug Manufacturer
- Drug Quantity Unit
- Drug Quantity Unit Description
- Drug Administration Type
- Drug Administration Location
- Drug Packaging
- Drug Packaging AIC Code
...
Normalized fact table

- Only one measure is used in the fact table
- Type of measure is described in Event Dimension Table and Investigation Dimension Table

### Chemotherapy Event Facts

- Date Key (FK)
- Prescribing Date Key (FK)
- Relative Date Key (FK)
- Patient Key (FK)
- Therapy Key (FK)
- Drug Key (FK)
- **Event Key (FK)**
- Investigated Key (FK)
- Numerical Value
- Textual Value
<table>
<thead>
<tr>
<th>Date Key</th>
<th>Date Type</th>
<th>Full Date</th>
<th>Day of Week</th>
<th>Weekday Ind</th>
<th>Month</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Normal</td>
<td>05.06.2012</td>
<td>Tuesday</td>
<td>Weekday</td>
<td>June</td>
<td>2012</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Patient Key</th>
<th>Patient ID</th>
<th>First Name</th>
<th>Last Name</th>
<th>Gender</th>
<th>Language</th>
<th>Weight</th>
<th>Height</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>345</td>
<td>Hans</td>
<td>Maier</td>
<td>male</td>
<td>German</td>
<td>68</td>
<td>185</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Therapy Key</th>
<th>Therapy ID</th>
<th>Therapy Name</th>
<th>Therapy Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>25436</td>
<td>NHL Chop 14</td>
<td>Profile</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Investigation Key</th>
<th>Group Level</th>
<th>Label</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Haemogram</td>
<td>MCH</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Haemogram</td>
<td>HB</td>
<td>g/dl</td>
</tr>
<tr>
<td>3</td>
<td>Haemogram</td>
<td>MCV</td>
<td>pg</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Event Key</th>
<th>Event ID</th>
<th>Event Name</th>
<th>Event Type</th>
<th>Responsible</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9372</td>
<td>Urgent Laboratory</td>
<td>Laboratory Test Nurse</td>
<td></td>
</tr>
</tbody>
</table>
Outline

1. The Grocery Store Example
2. More about Multidimensional Modeling
3. Inventory Management Example
4. Order Management
5. OncoNet
6. MEDAN
MEDAN (MEdical Data Warehousing and ANalysis)

- Collaboration between the Hospital Meran and the FUB
- Objectives
  - Conduct research and create competences in the field of medical data warehousing and analysis
  - Build a BI/DW solution for the hospital
    - Administrative DW
    - Medical DW
  - Develop and apply data analysis/mining techniques
MEDAN Data Sources

- Data sources in a health care environment
  - Internal production systems (SQL, Excel, Text files, ...)
  - External information systems

![Diagram showing Medawi health care data structure]
Budgeting Using a Spreadsheet

Example: Budgeting in the controlling department

- Complex and error-prone Excel spreadsheets have been used in the past
- Monthly reports have to be sent to the Province

<table>
<thead>
<tr>
<th>Costo Efficienza</th>
<th>2019</th>
<th>PIANO</th>
<th>PESO</th>
<th>Limite Inferiore</th>
<th>Limite Superiore</th>
<th>PRO%</th>
<th>SCOST.</th>
<th>MAT. RATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consumi beni sanitari</td>
<td>30.06</td>
<td>4.865</td>
<td>4.11</td>
<td>50%</td>
<td>0</td>
<td>437.38</td>
<td>0</td>
<td>501.299</td>
</tr>
<tr>
<td>P.I. + HOSEP2</td>
<td>30.06</td>
<td>3.646</td>
<td>3.728</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3.264</td>
</tr>
<tr>
<td>Consumi beni non sanitari</td>
<td>30.06</td>
<td>3.856</td>
<td>3.651</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6.524</td>
</tr>
<tr>
<td>altri costi</td>
<td>30.06</td>
<td>4.536</td>
<td>4.536</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>511.107</td>
</tr>
<tr>
<td>Totale costi</td>
<td>30.06</td>
<td>994.724</td>
<td>994.724</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1,020.199</td>
</tr>
<tr>
<td>costi personali (non da classificare)</td>
<td>30.06</td>
<td>1,000</td>
<td>7.00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6.29</td>
</tr>
<tr>
<td>presenza media</td>
<td>31.06</td>
<td>9.16</td>
<td>9.16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6.44</td>
</tr>
<tr>
<td>Tasso utilizzo letti</td>
<td>31.06</td>
<td>54.19%</td>
<td>51.31%</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>54.26%</td>
</tr>
<tr>
<td>degenza media</td>
<td>31.06</td>
<td>3.33</td>
<td>3.16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3.00</td>
</tr>
<tr>
<td>tasso op</td>
<td>30.06</td>
<td>70.73%</td>
<td>87.23%</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>70.73%</td>
</tr>
<tr>
<td>% ricoveri di 1 giorno</td>
<td>31.06</td>
<td>7.81%</td>
<td>2.00%</td>
<td>50%</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>54.26%</td>
</tr>
<tr>
<td>peso medio ORG</td>
<td>30.06</td>
<td>6.66</td>
<td>6.66</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6.44</td>
</tr>
<tr>
<td>mobilità provinciale passeggeri</td>
<td>30.06</td>
<td>121.248</td>
<td>121.248</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>959.214</td>
</tr>
<tr>
<td>mobilità provinciale attiva</td>
<td>30.06</td>
<td>593.010</td>
<td>593.010</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>24.44</td>
</tr>
<tr>
<td>mobilità instituzioni</td>
<td>30.06</td>
<td>99.99</td>
<td>99.99</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6.524</td>
</tr>
<tr>
<td>servizio trasporti n° pazienti</td>
<td>30.06</td>
<td>32</td>
<td>32</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>21</td>
</tr>
<tr>
<td>servizio trasporti €</td>
<td>30.06</td>
<td>1.566</td>
<td>1.566</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>999</td>
</tr>
</tbody>
</table>

Sommario: 100%
A *QlickView* application to replace Excel
- *QlickView* is a state-of-the-art data analysis tool
- All data are kept in main memory → fast
- Easy-to-use for small/medium sized applications
- No solution for ETL/data staging

- Direct access to the data sources
- Data integration in *QlickView*
- No DW in place
**Budgeting Using QlickView/2**

- QlickView application: Actual budget
- Big improvement over Excel solution
Budgeting Using QlickView/2

- QlickView application: Budget over time
It was difficult to convince decision makers to do proper modeling and build a DW as the core of a BI solution.

- They were too much technology-driven (QlikView, ...)
- QlickView is mainly an analysis tool and cannot replace a DW
  - Good for quick and small ad-hoc solutions
  - Difficult to do data cleaning and hence to control data quality
  - Not scalable for many applications, changing sources, etc.

Since Oracle technology was already in place, we finally were able to convince them to use this technology.
MEDAN Architecture

- Oracle ODI for ETL part and Data Storage
- Oracle DB for the DW
- QlikView and OBI for data analysis
MEDAN Bottom Up Approach

- We developed a prototype of DM CONT for the Hospital of Meran, consisting of 3 cubes:
  - Hospital Stays
  - Services
  - Transfers
- Each cube corresponds to a business process
- Goal: Deploy DM CONT in the Hospital of Meran, then in the other hospitals in South Tyrol
- Repeat the same cycle for the other DMs: DM Personnel, DM Pharmacy, DM Laboratory, etc.
Each event stores a hospital stay of a patient

Similar model for the other data cubes:
- Services
- Transfer

Shared dimensions are used
Conceptual Model of Services and Transfers
Logical Model of Hospital Stays

- **Snowflake schema**
Multi-valued Dimensions

- Diagnoses are an example of multi-valued attributes/dimensions, i.e., many-to-many relationships between attributes (multiple arcs in DFM)
  - a patient typically has multiple diagnoses (up to > 10)
  - books and authors is another example

- Solutions
  - Reserve multiple columns (one for each diagnosis)
    - Results in many empty cells, i.e., sparse fact table
  - Use several facts for a single hospital stay (one for each diagnosis)
    - Similar to fact normalization for measures
    - Increases the number of tuples in fact table

- Using bridge tables (next slide)
Bridge Tables

- **Bridge tables** help to deal with multi-valued dimensions
- Bridge table implements two one-to-many relationships

### Hospital Stay Facts
- Data Hospital Stay Admission (FK)
- Data Hospital Stay Discharge (FK)
- Health Record Number (FK)
- Health Record Year (FK)
- Patient Key (FK)
- Diagnosis Group (FK)
- ...

### Diagnosis Group Dimension
- Diagnosis Group Key (PK)
- Diagnosis Group Name
- ...

### Diagnosis Group Bridge
- Diagnosis Group Key (FK)
- Diagnosis Key (FK)
- Diagnosis Level
- Weighting Factor
- ...

### Diagnosis Dimension
- Diagnosis Key (FK)
- Diagnosis Name
- Diagnosis Code
- Diagnosis Type
- Diagnosis Category
- ...
MEDAN Lessons Learned

- Developing a BI platform is a **process that takes years**
- A well-designed and **consistent DW is the foundation for BI**
  - QlikView is a tool for quick analyses, but cannot replace a DW
- Do not put **anything** in a single data mart
  - Use one DM for one business process (set of closely related business queries)
- Different opinions on bottom-up vs. top-down, but bottom-up seems to have more acceptance
- Data modeling is **difficult but unavoidable**
  - Helps to get a conformed view on the business
    - e.g., what is an admission/hospital stay?
  - Different granularity by different users, e.g., Province, hospital
- A **Business Intelligence Competence Center (BICC)** was missing, but would have been very helpful to
  - coordinate the whole project
  - take important decisions about the data
Simplified **DW design** process by Kimball and Ross consists of 4 steps:
- Choose business processes, granularity, dimensions, and measures
- **Surrogate key** should be used instead of operational codes
- **Degenerate dimensions** are stored in the fact table
  - or stored all together in a **junk dimension**
- **DW sizing**
  - dimensions are a small portion of the **DW**, hence can/should contain as much information as possible
  - fact table determines the size of the **DW**
- **Dimensional model** is easy to use, e.g., drag and drop for report generation
Different Inventory models: periodic snapshot model, transactions model, accumulating

**Shared and conformed dimensions** are crucial to integrate several DMs across a value chain

**DW bus architecture** is a standard interface to support incremental DW design

**DW bus matrix** is a way to document the DW bus architecture

**Role-playing** allows to physically store a dimension only once, but use it several times in different roles and with different names

**Fact normalization** collapses all measures into a single measure together with a special fact dimension to determine the type of the measure

- Only useful when the fact table is sparse

**Multi-valued dimensions** if a dimension occurs more than once in a single fact, e.g., a patient has typically several diagnoses

- Corresponds to multiple arcs in DFM (many-to-many relationships)

**Bridge tables** can be used to represent such dimensions