Modelling a Data Warehouse

Overview

- Steps for modelling a DWH
- Data granularity
- Data storage
- Attribute hierarchies
- Querying a DWH / OLAP
- Frequent mistakes when building a DWH
- Example: grocery store (R. Kimball [KIM96])

Modelling a Data Warehouse

Basics

- Data from source systems: OLTP, legacy systems, syndicated data
- Cleaned - within itself consistent data

Data available -> building a Data Warehouse:

- Which business processes to model
- Defining granularity of data to be fed into the DWH
- Modelling the DWH data structure for storing the data
- Transforming data according to DWH structure
- Calculating aggregations and derived attributes

Example: Grocery Store

- Grocery chain with 500 stores spread over 3 states in the US
- Stores: supermarkets with departments like grocery, dairy, meat, frozen food, bakery, liquor, drugs etc.
- About 60,000 products in each store
(R. Kimball, [KIM96])

Which Business Processes to Model

DWH represent a business process view of the underlying data as opposed to the transaction-oriented OLTP systems

The decision which business processes to model has serious effects on the resulting data warehouse
- Problems to be addressed
- Questions to be asked
- Information needed and available
- Central DWH or data marts?

Example: Business Process

OLTP - data:

- Point-of-Sales data (POS)
- Vendor delivery data
- Accounting data
- Customer data
- Promotions
- ...

Goal: build a daily item movement database

Granularity

- Data is fed into the DWH at a certain level of granularity
- Based on this level of granularity aggregations can be defined
- Higher granularity - more data, lower granularity - less data

Questions:

- Which levels of granularity are available?
- Which levels of granularity are reasonable and useful in the DWH
  (temperature sensor data: per millisecond, second, minute, hour?)

Tendancy to store highest-granularity data where possible - once the granularity has been reduced, detailed information is no longer recoverable
**Modelling a Data Warehouse**

**Example: Granularity**

Which granularity for POS data?
Possibilities:
- single transactions per customer per product per store
- group transactions per customer per product per store
- group transactions per store per product per day
- group transactions per store per product per week
- group transactions per day per product group per region

Goal: daily item movement database
transactions per day per product per store

**Ways of Storing the Data**

- Data used for OLAP analysis must be stored in some kind of database to be accessed by the OLAP engine
- MOLAP?
- ROLAP?
- HOLAP?
- Data Marts?

**Facts and Dimensions**

**Facts:**
- Represent primary business process areas
- Unlikely to change once they are generated
- Stored at a certain level of granularity

**Dimensions:**
- Reference information by which facts can be structured for analysis
- Define aggregation hierarchies

**Example: Storage for Grocery DWH**

- Relational databases widely available
- Relational databases used for OLTP systems at companies
- Experienced IT personnel at companies used to relational databases
- ROLAP approach currently most common

Relational database used for storing Grocery DWH data

**Example: Facts and Dimensions**

- Grocery Store, POS-Data
- Facts:
  - POS: sales facts
- Dimensions:
  - Time
  - Store
  - Promotion
  - Product
Modelling a Data Warehouse

ROLAP Storage Schema - Schema

Star Schema:
- Partitioning the data
- Denormalizing tables
- One central fact table is surrounded by several dimension tables
- Queries address the fact table and are structured using the dimension tables
- No need for joins spanning multiple tables
- Most prominent model for DWH

Snowflake Schema:
- Based on star - schema
- Fact table structure identical to star - schema
- Dimension tables normalized (3. NF)
- Cleaner structuring of dimensions
- Database people used to 3. NF
- But: necessity to hide the now more complex structure from the user
- Usually not fully normalized dimension tables

But: necessity to hide the now more complex structure from the user

Example: Choosing a Schema for Grocery DWH

- Snowflake schema higher normalized
- Uses less disk space
- Browsing by direct access to tables more complicated because of references spanning multiple tables
- Dimension tables rather small -> little disk space benefit compared to size of DWH
- Star schema simpler to administer

Choosing a star schema for the grocery store DWH

Example: Star Schema for Grocery Store DWH

Attributes
- Decision which fields to add to the various dimension tables as well as to the fact table
- Attribute hierarchies
- Aggregation levels
- Considering possible queries and constraints on the tables
- Effects on OLAP operations like drill-down, roll-up
- Separately for each table

Example: Fact Table

- Stores data relevant for chosen business process area
- Includes key to the attached dimension tables
- Data taken from OLTP system: POS data
- Product sales per store per day
- Defining the place where the aggregation takes place: POS systems calculate the sales for each product and upload to the central DWH

Example: Fact Table (2)

Fact table attributes for sales data

<table>
<thead>
<tr>
<th>keys</th>
<th>facts</th>
</tr>
</thead>
<tbody>
<tr>
<td>time_key</td>
<td>dollar_sales</td>
</tr>
<tr>
<td>product_key</td>
<td>units_sales</td>
</tr>
<tr>
<td>store_key</td>
<td>dollar_cost</td>
</tr>
<tr>
<td>promotion_key</td>
<td>customer_count</td>
</tr>
<tr>
<td></td>
<td>... (Ibid)</td>
</tr>
</tbody>
</table>
Modelling a Data Warehouse

Example: Fact Table (3)

- Key of fact table is made up of four foreign keys of dimension tables
- Basic facts obtained from the POS system
- Additional derived attributes for analysis purposes to be defined
- Size considerations (estimations): Gross revenue of grocery chain: $4 billion, average price of product $2 -> about 2 billion items sold (ticket lines)
  3 years history -> 6 billion records -> storing single transactions not easily feasible
  2 billion ticket lines divided by 365 days divided by 500 stores -> ~11,000 items data per day per store to be transferred if aggregation is performed at central DWH
  average store 30,000 different products, about 10% sold per day -> transferring ~3000 records per day per store to central DWH if aggregation is performed at POS

Modelling a Data Warehouse

Example: Dimension Time

- Most general dimension
- Present in almost any DWH
- ‘Date’ attribute enough if only consecutive order of days relevant
- Separate dimension for evaluations concerning days of week, fiscal periods, seasons, holidays, special events, festivals etc.
- Can be built in advance

Modelling a Data Warehouse

Example: Dimension Time (2)

Time dimension for daily data

<table>
<thead>
<tr>
<th>day_key</th>
<th>quarter</th>
</tr>
</thead>
<tbody>
<tr>
<td>day_of_week</td>
<td></td>
</tr>
<tr>
<td>day_number_in_month</td>
<td></td>
</tr>
<tr>
<td>day_number_overall</td>
<td></td>
</tr>
<tr>
<td>week_number_in_year</td>
<td></td>
</tr>
<tr>
<td>week_number_overall</td>
<td></td>
</tr>
<tr>
<td>month</td>
<td></td>
</tr>
<tr>
<td>month_number_overall</td>
<td></td>
</tr>
<tr>
<td>year</td>
<td></td>
</tr>
<tr>
<td>year_number_overall</td>
<td></td>
</tr>
</tbody>
</table>

Modelling a Data Warehouse

Example: Dimension Time (3)

- Preloaded with data for 5 - 10 years -> ~3650 records
- day_of_week allows fast comparison of e.g. Monday to Saturday business
- day_number_in_month and last_day_in_month_flag used for daywise comparison, pay-day analysis
- day_number_overall consecutive numbering of days allowing fast arithmetics across month and year boundaries
- Similar flags for weekday - weekend, business, holiday - non holiday, quarter comparisons etc.
- event for special events like festivals, strike, catastrophies
- MIND: promotion periods could be handled as time attribute, but since they vary from store to store and from product to product this is not feasible - promotions form separate dimension

Modelling a Data Warehouse

Example: Dimension Time (4)

Star-Schema vs. Snowflake-Schema

- Identifies each product by its stock keeping units ID (SKU)
- Based on universal product code (UPC) imprinted as barcode
- Includes special codes for in-store products like fresh meat, groceries, bakery goods etc.
- Stores description of products
- Package size, product groups, brand names, subcategories and categories, department, etc.

Modelling a Data Warehouse

Example: Dimension Product

- Most general dimension
- Present in almost any DWH
- ‘Date’ attribute enough if only consecutive order of days relevant
- Separate dimension for evaluations concerning days of week, fiscal periods, seasons, holidays, special events, festivals etc.
- Can be built in advance
Modelling a Data Warehouse

Example: Dimension Product (2)

Product dimension

- **product_key (SKU)**
- SKU_description
- package_size
- brand
- subcategory
- category
- department
- package_type
- diet_type
- weight
- weight_unit_of_measure
- units_per_retail_case
- units_per_shipping_case
- cases_per_pallet
- shelf_width
- shelf_height
- shelf_depth
- ...

- Usually many more attributes stored in product dimension table
- Causal conditions highly correlated: price reduction or coupons combined with ads

- So-called 'normalization': only about 30 attributes present in the fact table storing only products sold (POS - data)
- Normalization: could be normalized to save disk space, but not necessarily
- Usually many more attributes stored in product dimension table

Example: Dimension Product (3)

- Managed by headquarters and distributed to stores
- Defines a kind of merchandise hierarchy, e.g. SKUs roll up to package_size -> brand
- Sales hierarchy: geography and sales region hierarchy
- Usually many more attributes stored in product dimension table

Example: Dimension Store

- Describes each store of the grocery chain
- Geographic dimension
- Created at headquarters by collecting information from stores (contrary to product data which is available at headquarters and distributed to stores)
- Two types of hierarchies in store dimension: geographic hierarchy and sales region hierarchy
- Attributes describing stores with respect to relevant analysis queries like store size, location, available departments etc.

Example: Dimension Store (2)

Product dimension

- store_key
- store_name
- store_number
- store_street_address
- store_city
- store_county
- store_state
- store_zip
- sales_district
- sales_region
- sales_manager
- store_phone
- store_tax
- floor_plan_type
- photo_processing_type
- finance_services_type
- first_opened_date
- last_remodel_date
- store_sftp
- grocery_sftp
- frozen_sftp
- meat_sftp
- ...

Example: Dimension Store (3)

- Geographic hierarchy: store -> store_zip -> store_county -> store_state
- Sales hierarchy: store -> sales_district -> sales_region
- floor_plan_type, finance_services_type are text fields filled with standardized descriptors that can be read and interpreted directly -> can be used for generating OLAP queries by interacting with the table
- first_opened_date, last_remodel_date are date-type fields either directly filled with data values or linked to copies of the time dimension

Example: Dimension Promotion

- Describes each promotion condition under which a product is sold, e.g. temporary price reduction, newspaper ads, coupons, etc.
- So-called 'causal dimension': factors are thought to change product sales
- Causal conditions highly correlated: price reduction or coupons combined with ads -> one record for each combination of promotions
- Can be used to analyze which products experienced an increased sale during the promotion period
- Cannot be used to analyze which products which did not sell because they are not present in the fact table storing only products sold (POS - data)
Modelling a Data Warehouse

Example: Dimension Promotion (2)

Product dimension
- promotion_key
- promotion_name
- price_reduction_type
- ad_type
- display_type
- coupon_type
- ad_media_name
- display_provider
- promo_cost
- promo_begin_date
- promo_end_date
- ...
Most queries address detailed data on one dimension and use summarized data on other dimensions?

- Additivity in time dimension?
- Additivity in stores dimension?
- Additivity in promotion dimension?

Goal: Accelerating the most frequent queries

Steps:
- Identify the most frequent queries
- Identify dimensions and aggregates that are most relevant to the respective business areas
- Define aggregate hierarchy
- Create selected pre-calculated aggregate fact tables
- Create corresponding aggregate dimension tables

The use of prestored summaries (aggregates) is the single most effective tool for the data warehouse designer to control performance

(R. Kimball, [KDM96])
Modelling a Data Warehouse

Example: Identifying Queries

- Sales of bakery goods during holiday periods compared to non-holiday periods
- Sales in the western sales district compared to the eastern sales district
- Sales of low-fat food products in the last 24 months
- Profitability of newspaper ads compared to radio commercials, effects of combinations of both
- ...

Example: Identifying Dimensions

Identify Dimensions

- Select dimensions most frequently involved in list of relevant queries
- Mind: select only the most relevant dimensions
- Consider size of aggregate tables: sparsity failure!

Sparcity: size explosion:
even if only 10% of a stores’ different products are being sold per day:
- not only 10% of all brands being sold in that store on that day
- more than 10% of all different products being sold over all stores

Example: Identifying Dimensions

Identify Hierarchies

Dimensions:
- Product?
- Stores?
- Time?
- Promotion?
- ...

Consider additivity of fact table attributes
(e.g. customer count per product group?)

Example: Hierarchy for Product

Attributes - Dimension Hierarchy:

- product_key (SKU)?
- SKU_description?
- package_size?
- brand?
- subcategory?
- category?
- department?
- package_type?
- diet_type?
- weight?
- weight_unit_of_measure?
- units_per_retail_case?
- units_per_shipping_case?
- cases_per_pallet?
- shelf_width?
- shelf_height?
- shelf_depth?
- ...

Example: Hierarchy for Product 2)
Modelling a Data Warehouse

Example: Aggregates for Stores

Attributes - Dimension Hierarchy:

- store_key
- store_name
- store_number
- store_street_address
- store_city
- store_state
- store_zip
- sales_district
- sales_region
- store_manager
- store_phone

- store_tax
- floor_plan_type
- photo_processing_type
- finance_services_type
- first_opened_date
- last_remodel_date
- store_sqft
- grocery_sqft
- frozen_sqft
- meat_sqft

- time_key
- day_of_week
- day_number_in_month
- day_number_overall
- week_number_in_year
- week_number_overall
- month
- month_number_overall

- quarter
- fiscal_period
- holiday_flag
- weekend_flag
- last_day_in_month_flag
- season
- event

- store_key
- store_state
- store_county
- store_city
- store_street_address
- store_manager
- floor_plan_type
- store_tax
- floor_plan_type
- photo_processing_type
- finance_services_type
- first_opened_date
- last_remodel_date
- store_sqft
- grocery_sqft
- frozen_sqft
- meat_sqft

- time_key
- day_of_week
- day_number_in_month
- day_number_overall
- week_number_in_year
- week_number_overall
- month
- month_number_overall

- quarter
- fiscal_period
- holiday_flag
- weekend_flag
- last_day_in_month_flag
- season
- event

Modelling a Data Warehouse

Example: Hierarchy for Stores (2)

Modelling a Data Warehouse

Example: Hierarchy for Time

Attributes - Dimension Hierarchy:

- time_key
- day_of_week
- day_number_in_month
- day_number_overall
- week_number_in_year
- week_number_overall
- month
- month_number_overall

- quarter
- fiscal_period
- holiday_flag
- weekend_flag
- last_day_in_month_flag
- season
- event

Modelling a Data Warehouse

Example: Hierarchy for Time (2)

Modelling a Data Warehouse

Aggregated Fact Tables

- Identify required/desired fact tables
- Higher-order aggregates can be calculated using lower-order aggregates e.g. sales per department can be based on sales per category
- Estimate their number and size
- Check availability of data
- Check needed dimension aggregate tables
Dimension tables tend to get very small as we move up the hierarchy e.g. 1 entry in dimension table for all_stores, 1 for all_products, 1-10 for all years, ...

Check which aggregate fact tables are needed and which can be calculated

Number and size of aggregate fact tables explodes!

- Three-way aggregate: category totals by district totals by month totals
- Two-way aggregate: district totals by month totals by product
- Two-way aggregate: category totals by month totals by store
- Two-way aggregate: category totals by district totals by day
- One-way aggregate: district totals by store by day
- One-way aggregate: category totals by store by day
- One-way aggregate: monthly totals by product by store
- Two-way aggregate: category totals by district totals by day
- Two-way aggregate: category totals by month totals by store
- Two-way aggregate: district totals by month totals by product
- Three-way aggregate: category totals by district totals by month totals

We end up with 7 aggregate dimension tables (small) and 35 aggregate fact tables

- 7 aggregated fact tables
- Aggregated fact tables are derivatives of basic fact table
- Checking additivity of fact table attributes:
  - dollar_sales ?
  - unit_sales?
  - dollar_cost ?
  - customer_count ?
- Aggregates of customer_count must be created at POS (point of sales)

- Keys for linking fact to dimension tables in starschema
- Keys for linking aggregated fact tables to aggregated dimension tables
- Defining attributes in aggregated dimension tables

- Keys for linking fact to dimension tables in starschema
- Keys for linking aggregated fact tables to aggregated dimension tables
- Defining attributes in aggregated dimension tables

- Keys for linking fact to dimension tables in starschema
- Keys for linking aggregated fact tables to aggregated dimension tables
- Defining attributes in aggregated dimension tables

- Keys for linking fact to dimension tables in starschema
- Keys for linking aggregated fact tables to aggregated dimension tables
- Defining attributes in aggregated dimension tables

- Keys for linking fact to dimension tables in starschema
- Keys for linking aggregated fact tables to aggregated dimension tables
- Defining attributes in aggregated dimension tables

Addtional fact tables can be added as necessity arises

Dimension tables tend to get very small as we move up the hierarchy e.g. 1 entry in dimension table for all_stores, 1 for all_products, 1-10 for all years, ...

Number and size of aggregate fact tables explodes!

Check which aggregate fact tables are needed and which can be calculated

- Product: per category, all merchandise total
- Store: per district, per region, all stores total
- Time: month, year

We end up with 7 aggregate dimension tables (small) and 35 aggregate fact tables
Modelling a Data Warehouse

Example: Possible Aggregated Fact Tables

Three way aggregates:

- Category by district by month
- Category by all stores by month
- Category by year
- Category by all stores by year
- All merchandise by all stores by year

Two way aggregates:

- Category by district by month
- Category by all stores by month
- Category by year
- Category by all stores by year
- All merchandise by district by month
- All merchandise by all stores by month
- All merchandise by year
- All merchandise by all stores by year

One way aggregates:

- Category by store by day
- Region by product by day
- Month by product by store
- All merchandise by store by day

District totals by month totals by product
Category totals by district totals by day
Category totals by month totals by store
Category totals by district totals by month

A suggestion / estimation:

Table | Sparsity | Factor
--- | --- | ---
Category totals by store by day | 0.2 | 0.2
District totals by store by day | 0.1 | 0.1
Monthly totals by product by store | 0.3 | 0.3
Category totals by district totals by day | 0.2\% | 0.02
Category totals by month totals by store | 0.2\% | 0.06
District totals by month totals by product | 0.1\% | 0.03
Category totals by district totals by month totals | 0.2\% | 0.06
TOTAL | 0.716

Baseline: 10,000 products * 1,000 stores * 100 time periods * 10% = 100 mil.
Plus aggregated fact tables: ~72% of baseline = 0.72 * 100 mil. = 72 mil.
Modelling a Data Warehouse

Sparsity Failure (4)

e.g. brands total by store by day:

- Assumption: only 10% of all products sold in given store on a given day ->
  1% of 10,000 products = 100 products being sold ->
  1,000 products in daily store data

- If every product sold belongs to a different brand ->
  1,000 different brands = 50%

- There may be 50% of all brands in the daily store data as opposed to 10% of the
  individual products ->
  1,000 brands in daily store data

- Aggregated fact table may have same size as basic fact table

Modelling a Data Warehouse

Sparsity Failure (5)

Table

<table>
<thead>
<tr>
<th>Product</th>
<th>Store</th>
<th>Time</th>
<th>Sparsity</th>
<th># Records</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base: Products by store by week</td>
<td>10,000</td>
<td>1,000</td>
<td>100%</td>
<td>100 mil.</td>
</tr>
<tr>
<td>Brand by store by week</td>
<td>2,000</td>
<td>1,000</td>
<td>50%</td>
<td>100 mil.</td>
</tr>
<tr>
<td>Product by district by week</td>
<td>10,000</td>
<td>1,000</td>
<td>50%</td>
<td>50 mil.</td>
</tr>
<tr>
<td>Brand by district by week</td>
<td>2,000</td>
<td>1,000</td>
<td>80%</td>
<td>16 mil.</td>
</tr>
<tr>
<td>Brand by store by period</td>
<td>2,000</td>
<td>1,000</td>
<td>30%</td>
<td>48 mil.</td>
</tr>
<tr>
<td>Product by district by period</td>
<td>10,000</td>
<td>1,000</td>
<td>80%</td>
<td>24 mil.</td>
</tr>
<tr>
<td>Brand by district by period</td>
<td>2,000</td>
<td>1,000</td>
<td>100%</td>
<td>6 mil.</td>
</tr>
</tbody>
</table>

TOTAL : 494 mill.

Database may grow up 394% !

---

Modelling a Data Warehouse

Sparsity Failure (6)

- Take care when designing aggregate tables

- Make sure that each aggregate summarizes at least about 10 - 20 records on the average

Example:

- Product dimension summarized only 5 lower level products on the average
- Time dimension summarized only about 3 periods on the average
- One way aggregates of time and product contributed 250 mil. records
- If each had summarized about 20 lower level items on average ->
  only about 70 mil. new records even with sparsity growing to 70%

Modelling a Data Warehouse

Querying Aggregate Tables

Queries transformed into SQL statements

e.g.: Show total sales by category in Cincinnati stores on New Year's Day 1998 for base fact table:

```sql
select category_description, sum(sales_dolars)
from base_sales_fact, product, store, time
where base_sales_fact.product_key = product.product_key
and base_sales_fact.store_key = store.store_key
and base_sales_fact.time_key = time.time_key
and store.city = 'Cincinnati'
and time.day = 'January 1, 1998'
and time.year = 1998
and category_sales_fact.category_key = category_product.product_key
and category_sales_fact.store_key = store.store_key
and category_sales_fact.time_key = time.time_key
and category_sales_fact.category_key = category_description.category_key

group by category_description
```

Modelling a Data Warehouse

Querying Aggregate Tables (2)

Same query if category totals aggregate table exists:

```sql
select category_description, sum(sales_dolars)
from category_sales_fact, category_product, store, time
where category_sales_fact.product_key = category_product.product_key
and category_sales_fact.store_key = store.store_key
and category_sales_fact.time_key = time.time_key
and category_sales_fact.category_key = category_description.category_key
and time.day = 'January 1, 1998'

group by category_description
```

Category_sales_fact and corresponding category_product dimension table replace base sales fact and basic product dimension table

Modelling a Data Warehouse

Querying Aggregate Tables (3)

- Navigator
- Reads users query and transforms it into best available aggregate query
- Metadata descriptions provide information about existing aggregate tables
- Existence of aggregate tables is transparent to the user
- Can be used to build statistics on user queries, aggregate table usage and the need for additional aggregates
- Allows incremental rollout of new aggregate tables
Modelling a Data Warehouse

Querying Aggregate Tables (4)

- Navigator: Replacing base-level fact and dimension tables with aggregated fact and dimension tables
  1.) Rank order all aggregate tables
  2.) Starting from the smallest, verify whether all of the dimensional attributes in the query can be found. If so, replace base tables in query with corresponding aggregate tables. If not continue with the next bigger aggregate table until finally reaching the base fact table

---

Modelling a Data Warehouse

Metadata

- Describes the data in the DWH
- Technical metadata
- Business metadata
- Operational metadata

---

Modelling a Data Warehouse

Example: Navigator

---

Modelling a Data Warehouse

Example: Metadata Fact Table

---

Modelling a Data Warehouse

Example: Metadata - Attributes

---

Modelling a Data Warehouse

Example: Metadata - Derived Attributes
Modelling a Data Warehouse

Example: Metadata - Dimensions

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Description</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>&quot;Product&quot;</td>
<td>Product</td>
</tr>
<tr>
<td>Key</td>
<td>&quot;product_key&quot;</td>
<td>key</td>
</tr>
<tr>
<td>Brand</td>
<td>&quot;brand&quot;</td>
<td>brand</td>
</tr>
<tr>
<td>Category</td>
<td>&quot;category&quot;</td>
<td>category</td>
</tr>
<tr>
<td>Subcategory</td>
<td>&quot;subcategory&quot;</td>
<td>subcategory</td>
</tr>
</tbody>
</table>

Modelling a Data Warehouse

Example: Metadata - Dimension Hierarchies

<table>
<thead>
<tr>
<th>Hierarchies of Dimension</th>
<th>previous</th>
<th>next</th>
</tr>
</thead>
<tbody>
<tr>
<td>brand</td>
<td></td>
<td></td>
</tr>
<tr>
<td>category</td>
<td></td>
<td></td>
</tr>
<tr>
<td>subcategory</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Modelling a Data Warehouse

Example: Metadata - Time Dimension

<table>
<thead>
<tr>
<th>Time Dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Year</td>
<td>&quot;Time&quot;</td>
</tr>
<tr>
<td>Month</td>
<td>&quot;Month&quot;</td>
</tr>
<tr>
<td>Week</td>
<td>&quot;Week&quot;</td>
</tr>
</tbody>
</table>

Modelling a Data Warehouse

Example: EIS Page

Executive Information Systems

- Briefing books
- Tables
- Charts

Modelling a Data Warehouse

Example: Briefing Book

BRIEFING BOOK

- Revenue of 97
- Costs and financial data
- Sales in detail-organized chart
- Overview of 97
- Revenue and cost items for year 97
- Information on sample blog
Modelling a Data Warehouse

Example: Table

- Not knowing what you really want
- Thinking that DWH-design is the same as transactional DB design - a DWH is not simply a big database!
- Loading the warehouse with data simply because it is available
- Underestimating the complexity of a DWH project
- Getting caught by technological gadgets
- Focusing on internal data and ignoring the use of external data
- Using data with overlapping or confusing definitions / semantics
- Believing performance and scalability promises
- Believing that once a DWH is running the work is done