Lecture 04

Storage Structure

Summary – last week

- Last week:
  - DWH Lifecycle
  - Basic Structure

- This week:
  - DW Architecture
    - Storage Architecture
    - Tier Architecture

Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

Storage Structure

- Storage structure
  - After extraction from the operational data, in DW information is stored in databases
  - The databases are operated by a DBMS
  - Different database structures can be used for a DW:
    - Relational model (RDB) operated by a RDBMS
    - MultiDimensional model (MDB) operated by a MDBMS

Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

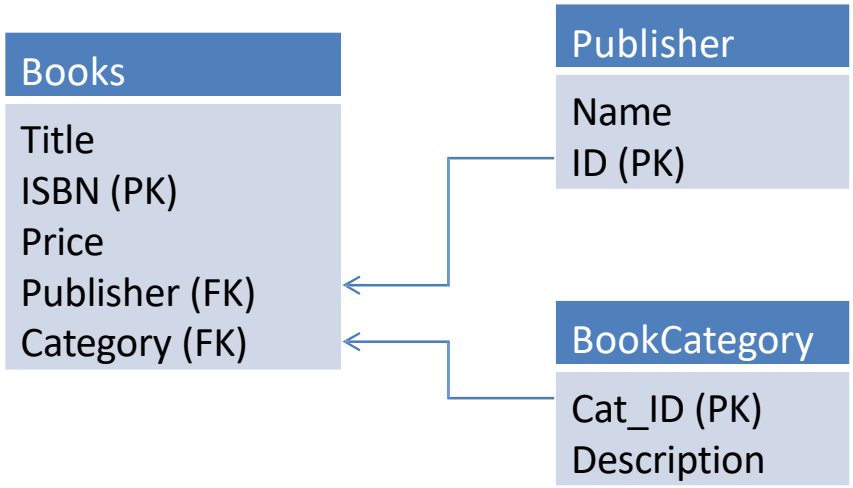
Storage Structure (cont'd.)

- RDB and MDB are complementary and do not have to exclude each other
  - In the staging area some RDBMS can be used, however it must be off-limits to user queries because of performance reasons
  - By default, normalized databases are excluded from the presentation area, which should be strictly multi-dimensionally (MDBMS)

Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

Relational DB

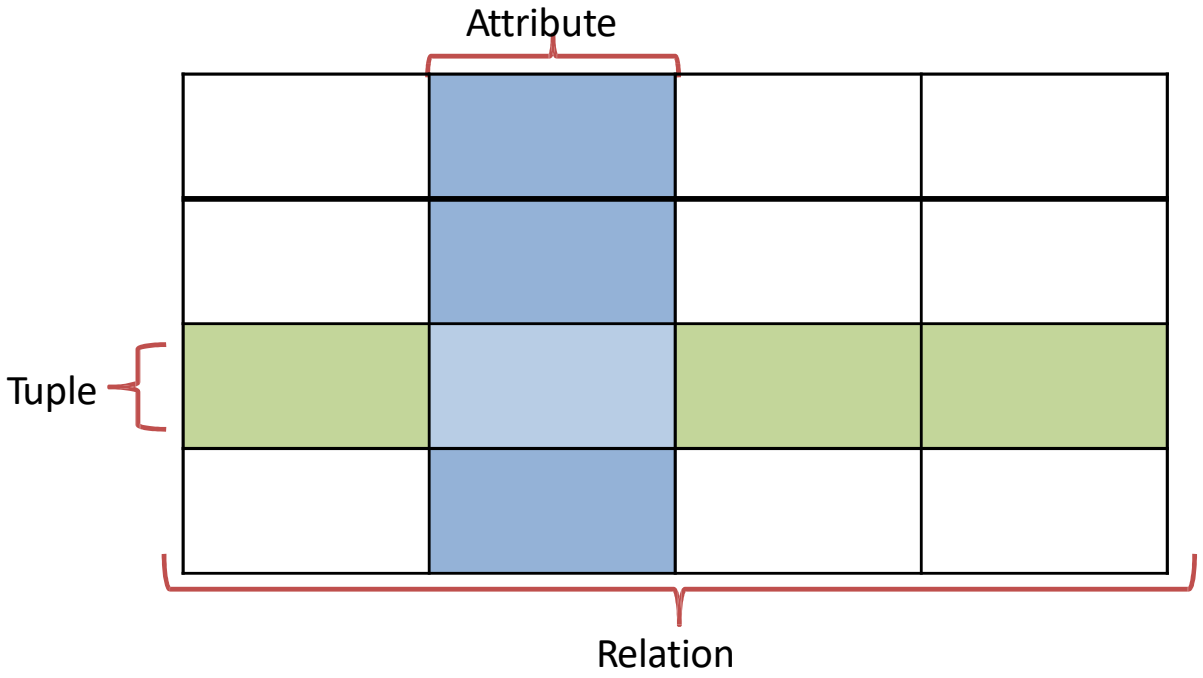
- DB in relational model
  - A database is seen as a collection of predicates over a finite set of variables
  - The content of the DB is modeled as a set of relations in which all predicates are satisfied



Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

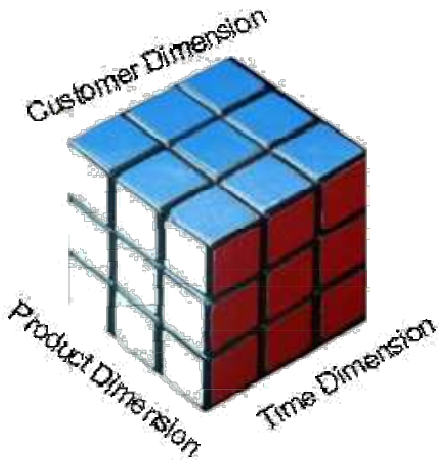
Relational DB (cont'd.)

- A relation is defined as a set of tuples that have the same attributes
  - It is usually described as a table



Multidimensional DB

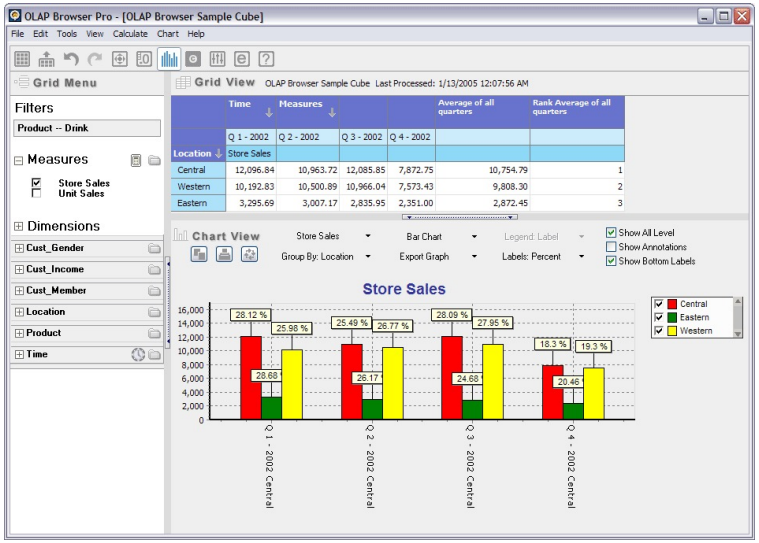
- Multidimensional DB (MDB) are optimized for DW and OLAP applications
  - They are created using input from the staging area
  - Designed for efficient and convenient storage and retrieval of large volumes of data
  - Stored, viewed and analyzed from different perspectives called dimensions



Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

Multidimensional DB (cont'd.)

- Example: an automobile manufacturer wants to increase sale volumes
  - Evaluation requires to view historical sale volume figures from multiple dimensions
  - Sales volume by model, by color, by dealer, over time



Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

Multidimensional DB (cont'd.)

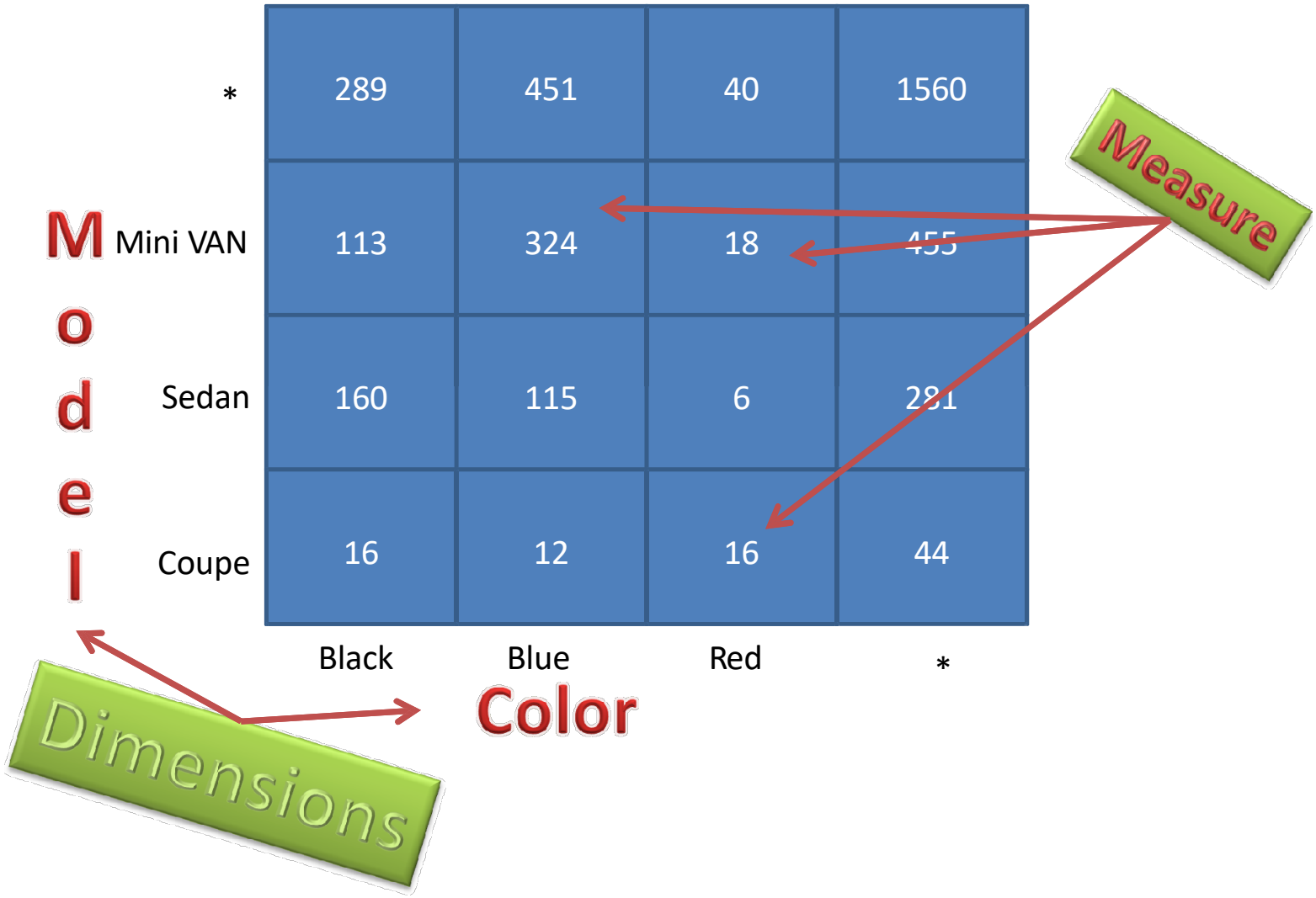
- A relational structure of the given evaluation would be

Model	Color	Sales volume
Mini VAN	Blue	324
Mini VAN	Black	113
Mini VAN	Red	18
Sedan	Black	160
Sedan	Blue	115
Sedan	Red	6
Sports coupe	Red	16
Sports coupe	Black	16
Sports coupe	Blue	12

Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

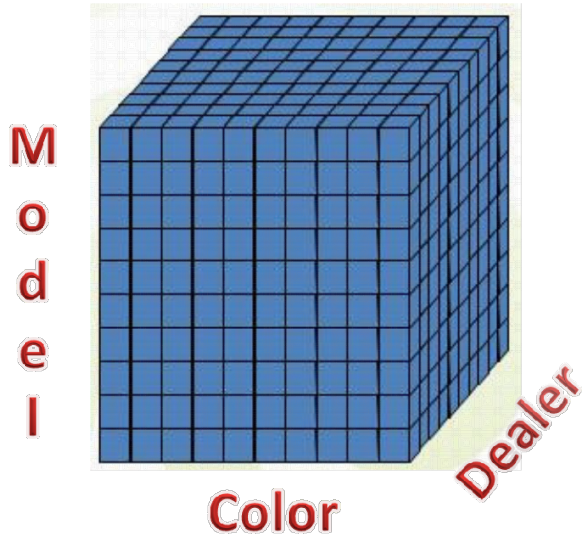
Multidimensional DB (cont'd.)

- Structure



Multidimensional DB (cont'd.)

- The complexity grows quickly with the number of dimensions and the number of positions
  - Example: 3 dimensions with 10 values each and no indexes
  - If we consider viewing information in a RDB it would result in a worst case of $10^3=1000$ records view



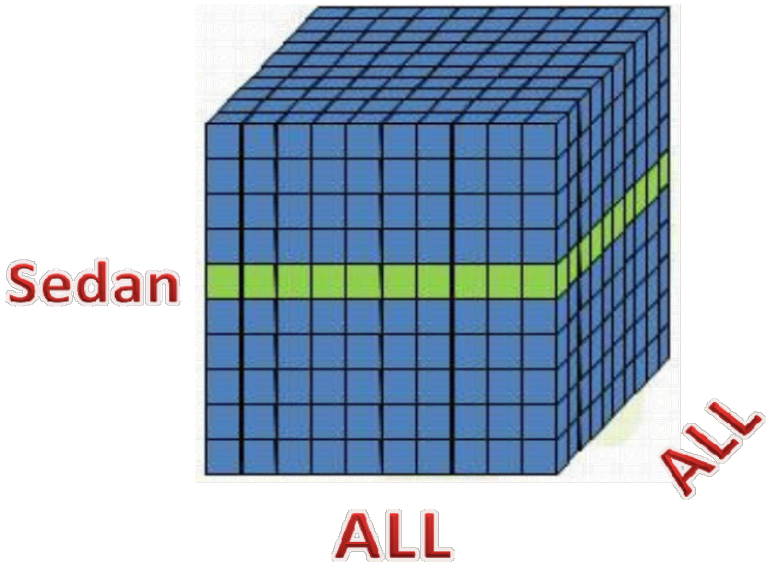
Multidimensional DB (cont'd.)

- Now, if we consider performance
  - For responding to a query when car type = Sedan, color = Blue, and dealer = Berg
    - RDBMS has to search through 1000 records to find the right record
    - MDB has more knowledge about where data lies
    - The maximum of searches in the case of MDB is of 30 positions

Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

Multidimensional DB (cont'd.)

- If the query is more relaxed
  - Total sales across all dealers for all colors when car type = sedan
    - RDBMS still has to go through the 1000 records
    - MDB, however, goes only through a slice of 10x10



Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

Multidimensional DB (cont'd.)

- Performance advantages
  - MDBs are an order of magnitude faster than RDBMSs
  - Performance benefits are more for queries that generate cross-tab views of data (the case of DW)
- Conclusion
  - The performance advantages offered by MDBs facilitates the development of interactive decision support applications like OLAP that can be impractical in a relational environment

Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

RDB vs. MDB

- Any database manipulation is possible with both technologies
- MDBs however offer some advantages in the context of DW:
  - Ease of data presentation
  - Ease of maintenance
  - Performance

RDB vs. MDB (cont'd.)

- Ease of data presentation
  - Data views are natural output of the MDBs
  - Obtaining the same views in RDB requires a complex query
    - Example with Walmart and Sybase:
      - select sum(sales.quantity_sold) from sales, products, product_categories, manufacturers, stores, cities where manufacturer_name = 'Colgate'

and product_category_name = 'toothpaste'
and cities.population < 40 000
and trunc(sales.date_time_of_sale) = trunc(sysdate-1)
and sales.product_id = products.product_id
and sales.store_id = stores.store_id
and products.product_category_id = product_categories.product_category_id
and products.manufacturer_id = manufacturers.manufacturer_id
and stores.city_id = cities.city_id

Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

RDB vs. MDB (cont'd.)

- Ease of data presentation
  - Top k queries cannot be expressed well in SQL
    - Find the five cheapest hotels in Islamabad
      - SELECT * FROM hotels h WHERE h.city = Islamabad AND 5 > (SELECT count(*) FROM hotels h1 WHERE h1.city = Islamabad AND h1.price < h.price);
    - Some RDBMS extended the functionality of SQL with STOP AFTER functionality
      - SELECT * FROM hotelsWHERE city = Islamabad Order By price STOPAFTER 5;

Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

RDB vs. MDB (cont'd.)

- Ease of maintenance
  - No additional overhead to translate user queries into requests for data
    - Data is stored as it is viewed
  - RDBs use indexes and sophisticated joins which require significant maintenance and storage to provide same intuitiveness

Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

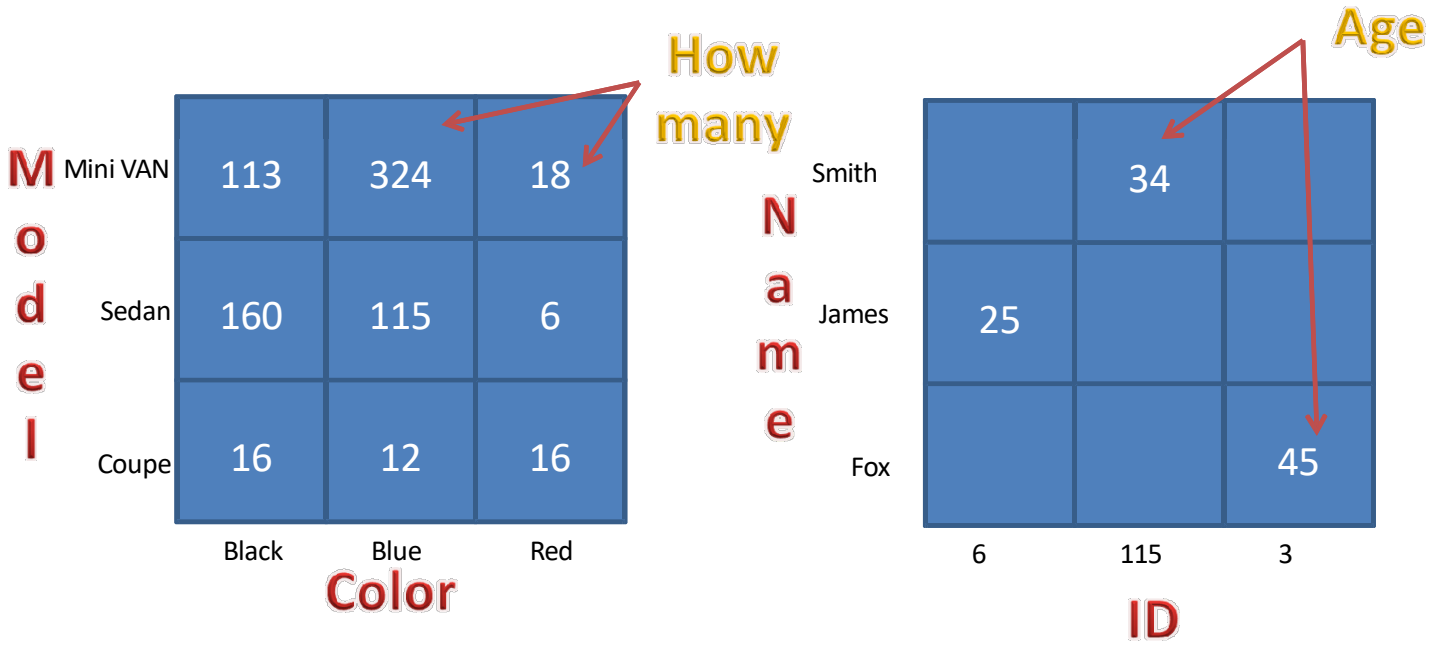
RDB vs. MDB (cont'd.)

- Performance
  - —Performance of MDBs can be matched by RDBs through database tuning
  - —Not possible to tune the database for all possible ad- hoc queries
  - —Aggregate navigators are helping RDBs to catch up with MDBs as far as aggregation queries are concerned

Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

When MDBs are In-appropriate?

- When MDBs are in-appropriate?
  - If the dataset types are not highly related, using a MDB results in a sparse representation



When MDBs are Appropriate?

- When MDBs are appropriate?
  - In the case of highly interrelated dataset types MDBs are recommended for greatest ease of access and analysis
  - Examples of applications
    - Financial Analysis and Reporting
    - Budgeting
    - Promotion Tracking
    - Quality Assurance and Quality Control
    - Product Profitability

Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

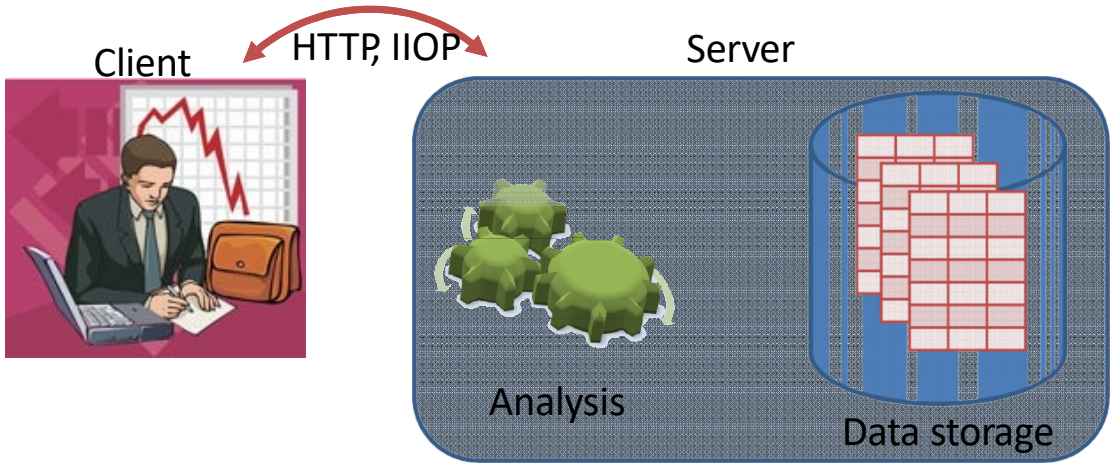
Tier Architecture

- Popular DW architectures
  - Generic Two-Tier Architecture
  - Independent Data Mart
  - Dependent Data Mart and Operational Data Store
  - Logical Data Mart and Active Warehouse
  - Three-Tier Architecture
- Other
  - One-Tier Architecture
  - N-Tier Architecture
  - Web-based Architecture

Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

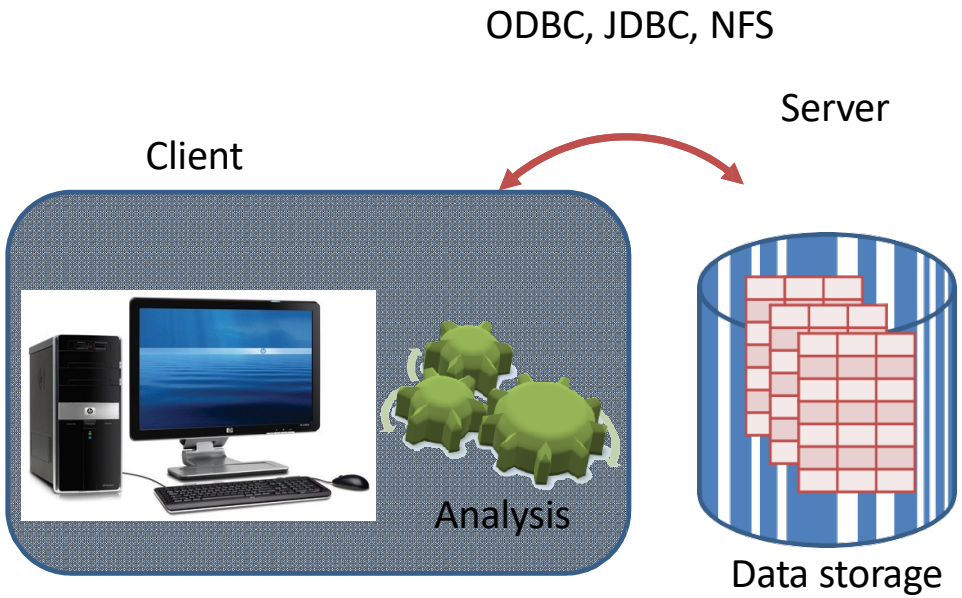
Layered Architecture

- Data **analysis** comes in two flavors
  - Depending on the execution place of the analysis
    - Thin Client
      - Analytics are executed on the server
      - Client just displays
      - This architecture fits well for Internet/Intranet DW access



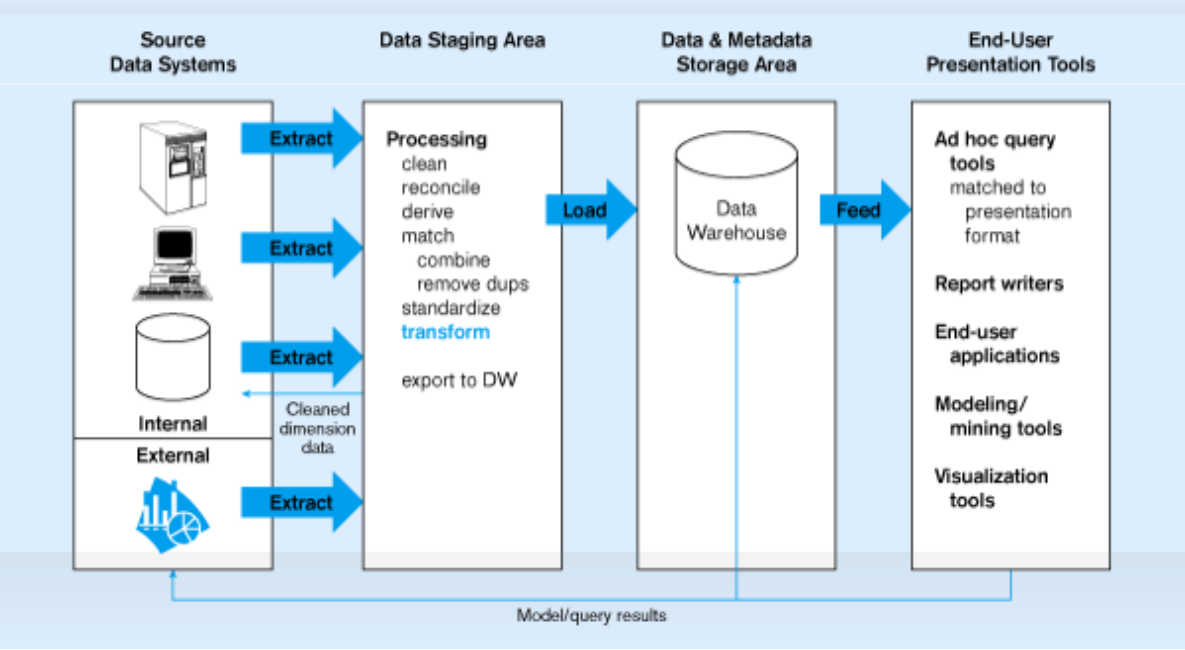
Layered Architecture (cont'd.)

- Fat Client
  - The server just delivers the data
  - Analytics are executed on the client
  - Communication between client and server must be able to sustain large data transfers



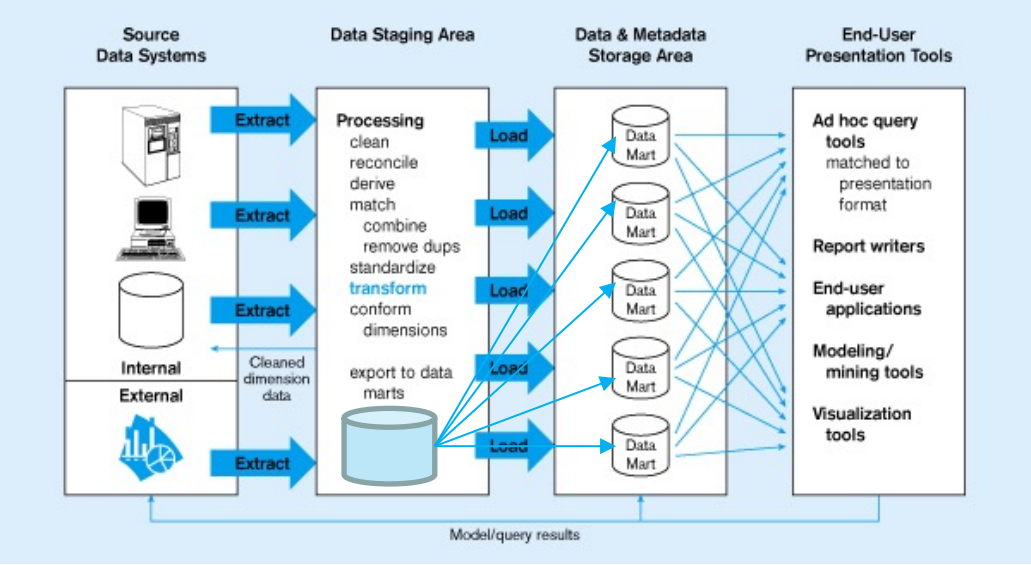
Layered Architecture (cont'd.)

- Generic Two-Tier Architecture
  - Data is not completely current in the DW
  - Periodic extraction



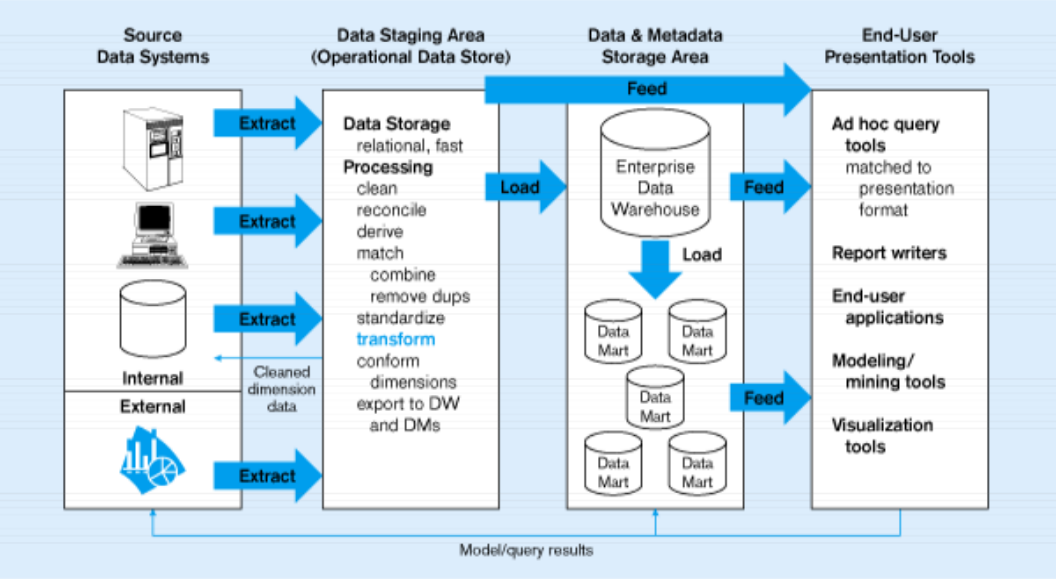
Layered Architecture (cont'd.)

- Independent Data Mart
  - Mini warehouses – limited in scope
  - Separate ETL for each independent Data Mart
  - High Data Marts access complexity



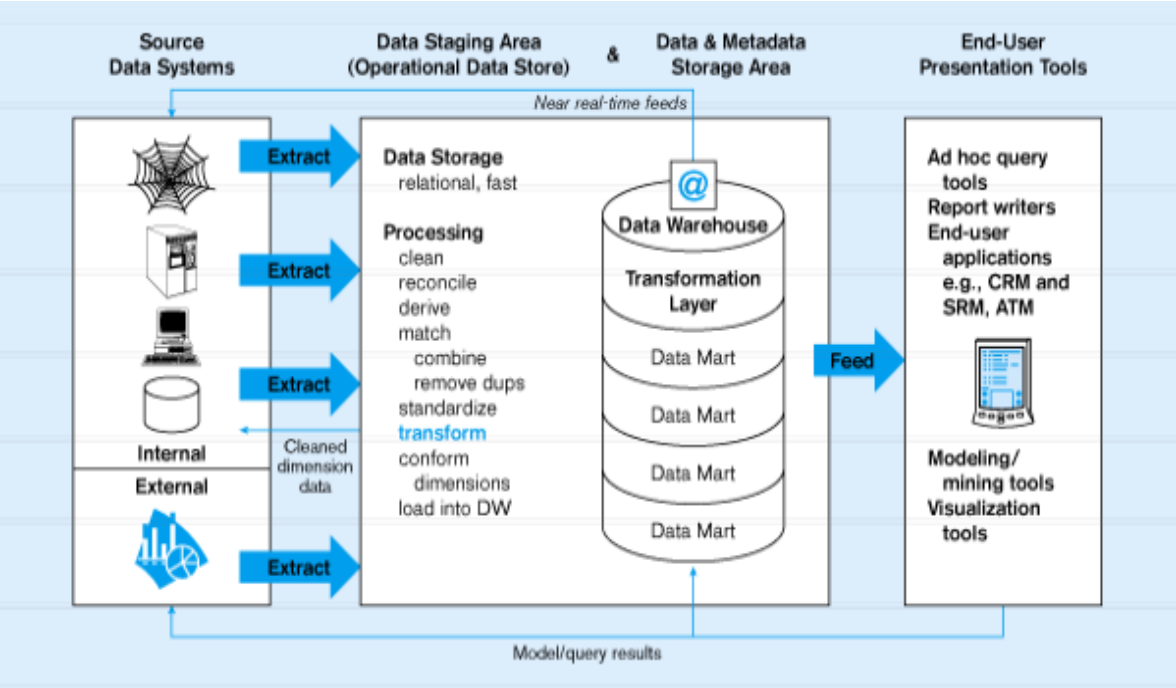
Layered Architecture (cont'd.)

- **Dependent Data Mart** and Operational Data Store
  - –Single ETL for the DW
  - – Data Marts are loaded from the DW
  - –More simple data access than in the previous case



Layered Architecture (cont'd.)

- **Logical Data Mart** and Active Warehouse
  - The ETL is near real-time
  - Data Marts are *not* separate databases, but logical views of the DW



DW vs. Data Marts

Scope	
DW	Data Marts
Application independent	Specific DSS application
Centralized,	Decentralized by user area
Planned	Organic, possibly not planned

Data	
DW	Data Marts
Historical, detailed, summarized	Some history, detailed, summarized
Lightly denormalized	Highly denormalized

Subjects	
DW	Data Marts
Multiple subjects	One central subject

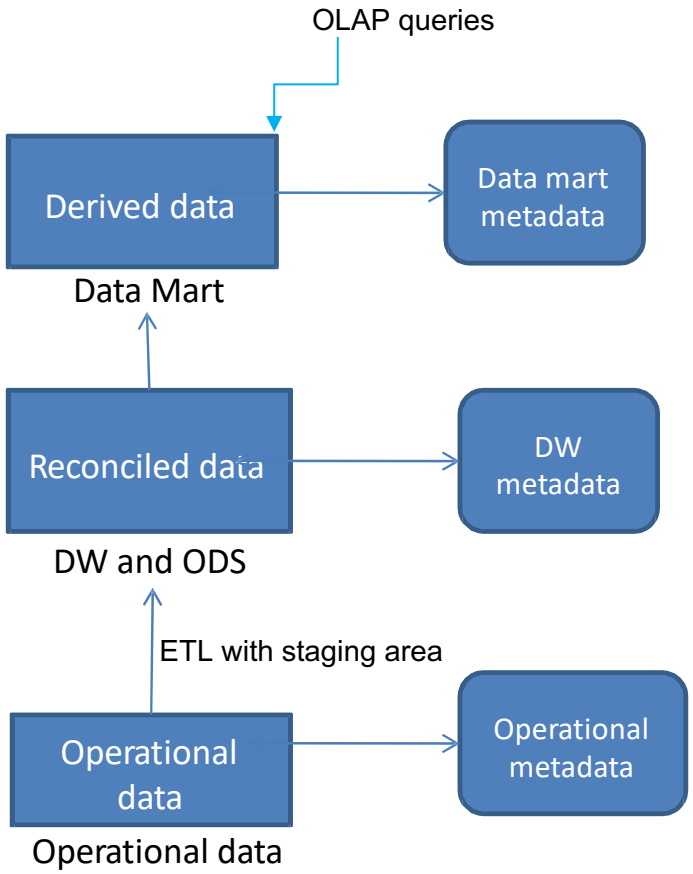
Sources	
DW	Data Marts
Many internal and external sources	Few internal and external sources

Other characteristics	
DW	Data Marts
Flexible	Restrictive
Data-oriented	Project oriented
Long life	Short life
Large	Start small, becomes large
Single complex structure	Multiple, semi-complex structure, together complex

Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

Layered Architecture (cont'd.)

- Generic Three-Tier Architecture
  - Derived data
    - Data that had been selected, formatted, and aggregated for DSS support
  - Reconciled data
    - Detailed, current data intended to be the single, authoritative source for all decision support



Layered Architecture (cont'd.)

- One-Tier Architecture
  - Theoretically possible
  - Might be interesting for mobile applications
- N-Tier Architecture
  - Higher tier architecture is also possible
    - But the complexity grows with the number of tier-interfaces
- Web-based Architecture
  - Advantages:
    - Usage of existing software, reduction of costs, platform independence
  - Disadvantages:
    - Security issues: data encryption/user access and identification

Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

Summary

- DW architectures:
  - Storage architecture
    - Relational DB
    - Multidimensional DB
  - Tier Architecture

Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides

Next lecture

- DW architecture (cont'd.)
  - Distributed DW
  - DW data modelling
    - Conceptual model

Acknowledgment - Thanks to Wolf-Tilo Balke and Silviu Homoceanu - TU Braunschweig for the slides