Data warehouse Introduction

Elena Baralis
Politecnico di Torino

Decision support systems

- Huge *operational databases* are available in most companies
  - these databases may provide a large wealth of useful information
- Decision support systems provide means for
  - in depth analysis of a company's business
  - *faster* and *better* decisions

Elena Baralis
Politecnico di Torino

Strategic decision support

- Demand evolution analysis and forecast
- Critical business areas identification
- Budgeting and management transparency
  - reporting, practices against frauds and money laundering
- Identification and implementation of winning strategies
  - cost reduction and profit increase

Elena Baralis
Politecnico di Torino

Business Intelligence

- BI provides support to strategic decision support in companies
- Objective: transforming company data into actionable information
  - at different detail levels
  - for analysis applications
- Users may have heterogeneous needs
- BI requires an appropriate hardware and software infrastructure

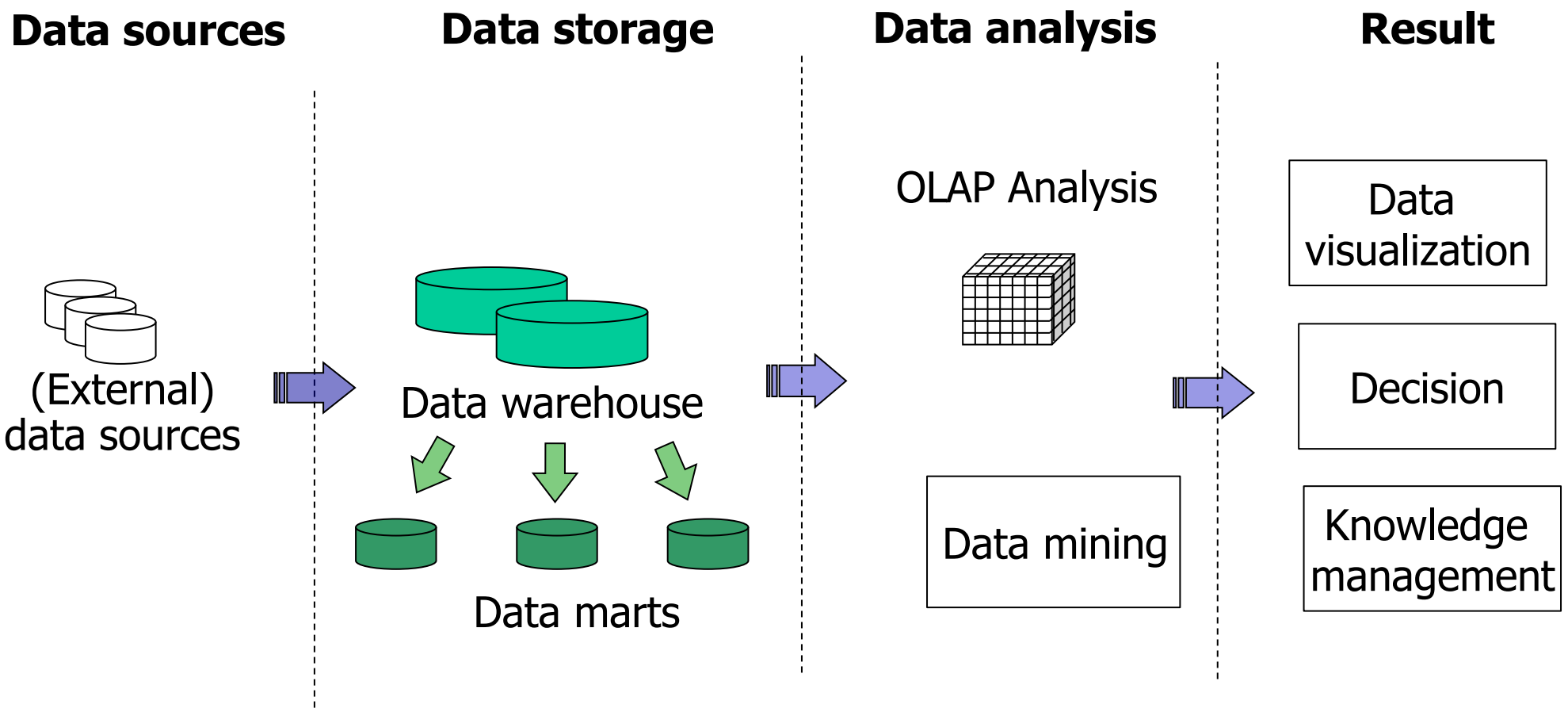
Elena Baralis
Politecnico di Torino

Applications

- Manifacturing companies: order management, client support
- Distribution: user profile, stock management
- Financial services: buyer behavior (credit cards)
- Insurance: claim analysis, fraud detection
- Telecommunication: call analysis, churning, fraud detection
- Public service: usage analysis
- Health: service analysis and evaluation

Elena Baralis
Politecnico di Torino

Business intelligence at a glance



Elena Baralis
Politecnico di Torino

Type of Information processing

- Transaction processing
- Analytical processing

Elena Baralis
Politecnico di Torino

Transaction processing

- On Line Transaction Processing (OLTP)
  - Traditional DBMS usage
- Characterized by
  - snapshot of current data values
  - detailed data, relational representation
  - structured, repetitive operations
  - read/write access to few records
  - short transactions
  - isolation, reliability, and integrity are critical (ACID)
  - database size ≈ 100MB-GB

Elena Baralis
Politecnico di Torino

Analytical processing

- On Line Analytical Processing (OLAP)
  - Decision support applications
- Characterized by
  - “historical” data
  - consolidated, integrated data
  - ad hoc applications
  - read access to millions of records
  - complex queries
  - consistency before and after periodical loads
  - database size ≈ 100GB-TB

Elena Baralis
Politecnico di Torino

Data warehouse

- Database devoted to decision support, which is kept *separate* from company operational databases
- Data which is
  - devoted to a specific subject
  - Integrated and consistent
  - time dependent, non volatile

  used for decision support in a company

W. H. Inmon, Building the data warehouse, 1992

Elena Baralis
Politecnico di Torino

Why separate data?

- Performance
  - complex queries reduce performance of operational transaction management
  - different access methods at the physical level
- Data management
  - missing information (e.g., history)
  - data consolidation
  - data quality (inconsistency problems)

Elena Baralis
Politecnico di Torino

Data model

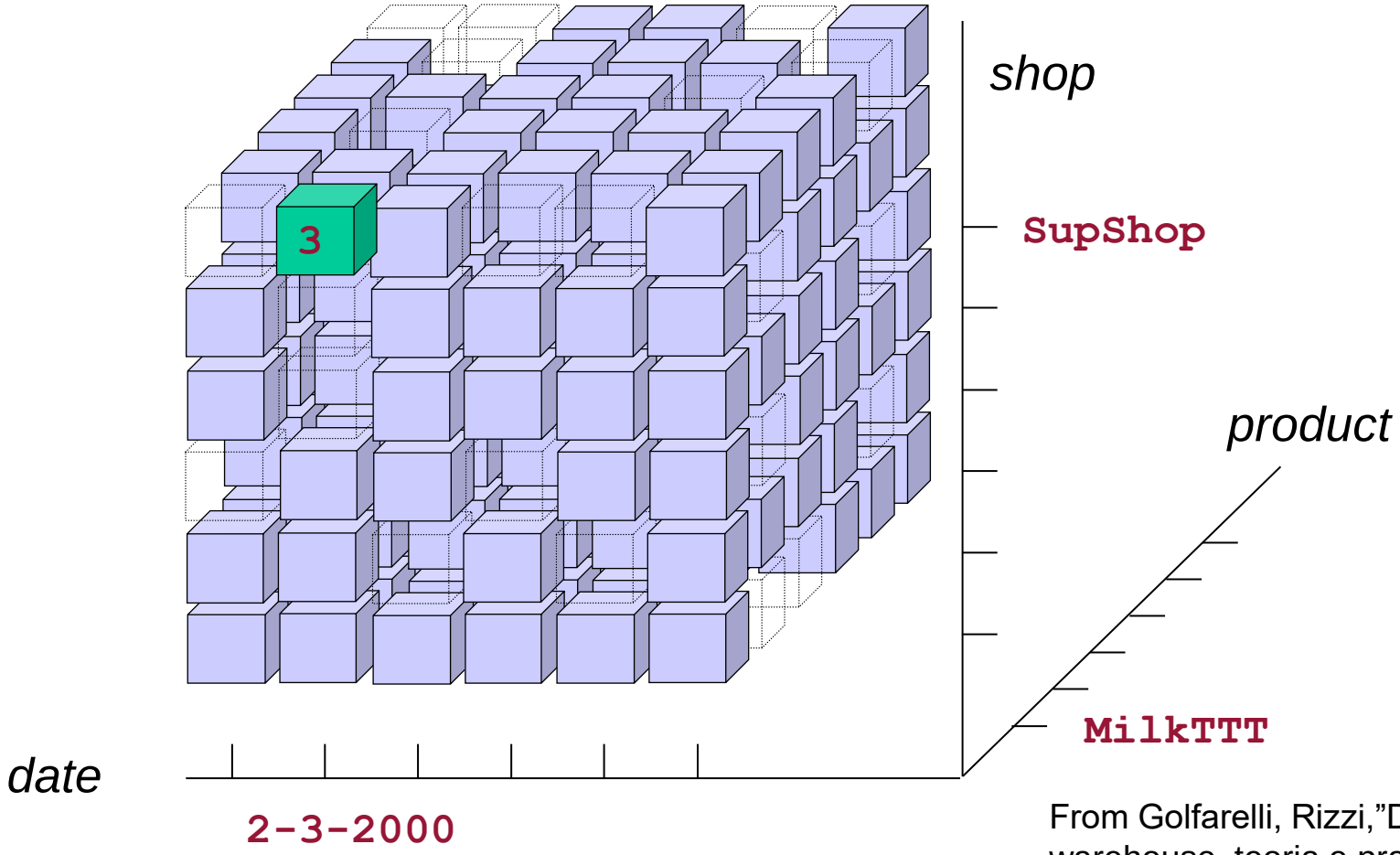
Elena Baralis
Politecnico di Torino

Multidimensional representation

- Data are represented as an (hyper)cube with three or more dimensions
- Measures on which analysis is performed: cells at dimension intersection
- Data warehouse for tracking sales in a supermarket chain:
  - dimensions: product, shop, time
  - measures: sold quantity, sold amount, ...

Elena Baralis
Politecnico di Torino

Multidimensional representation



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

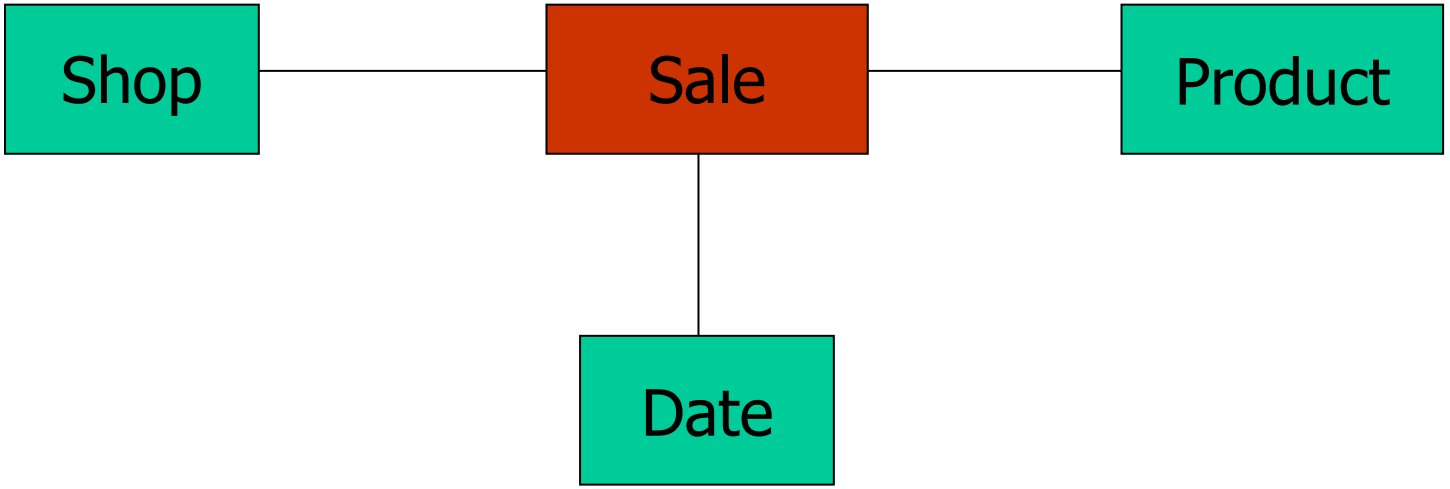
Relational representation: star model

- Numerical measures stored in the *fact table*
  - attribute domain is numeric
- *Dimensions* describe the context of each measure in the fact table
  - characterized by many descriptive attributes

Elena Baralis
Politecnico di Torino

Example

Data warehouse for tracking sales in a supermarket chain



Elena Baralis
Politecnico di Torino

Data warehouse size

- Time dimension: 2 years x 365 days
- Shop dimension: 300 shops
- Product dimension: 30.000 products, of which 3.000 sold every day in every shop
- Number of rows in the fact table:

  730 x 300 x 3000 = 657 millions

⇨ Size of the fact table ≈ 21GB

Elena Baralis
Politecnico di Torino

NOSQL data representation

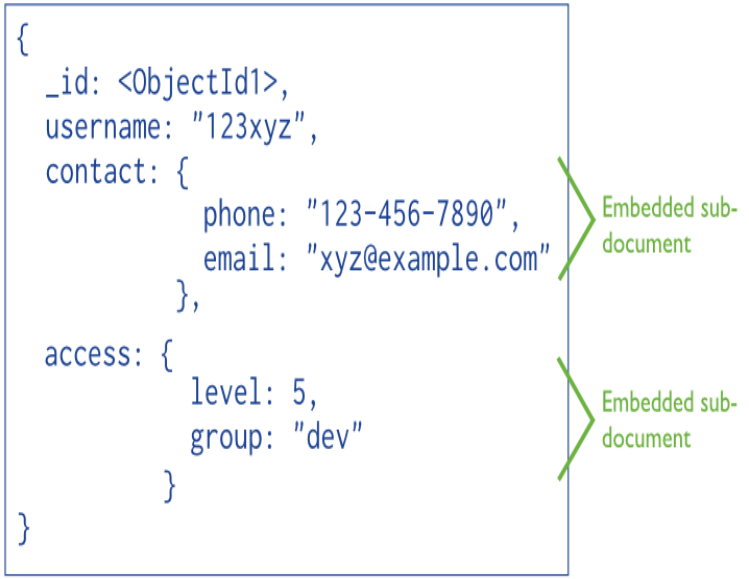
- A database is a set of collections
- Each collection contains a set of documents
- Each document is described by a list of key-value fields and each field can hold any data type
- Documents from the same collection can be heterogeneous
- Since the data representation is schema-less it not required to define the schema of the documents a-priori and objects of the same collections can be characterized by different fields

Relational database	NOSQL database
Table	Collection
Row	Document
Column	Field

Elena Baralis
Politecnico di Torino

Example of Document Data

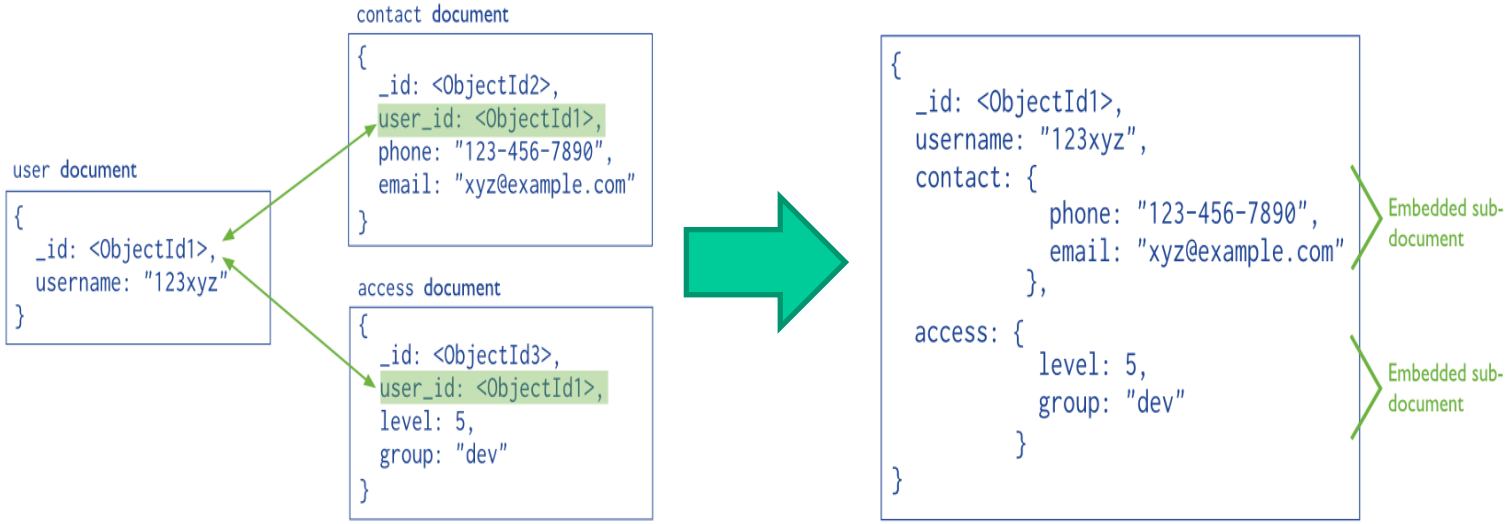
- Records are stored into Documents
  - field-value pairs
  - similar to JSON objects
  - may be nested



Elena Baralis
Politecnico di Torino

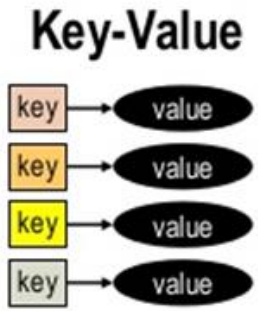
Example of Document Data

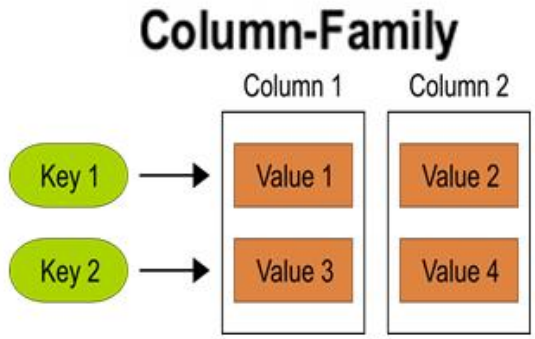
- Relations among documents are inefficient, and leads to de-normalization
  - Object(ID) reference, with **no native join**

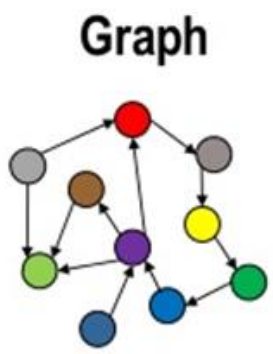


Elena Baralis
Politecnico di Torino

Types of NoSQL databases







Elena Baralis
Politecnico di Torino

Data analysis

Elena Baralis
Politecnico di Torino

Data analysis tools

- OLAP analysis: complex aggregate function computation
  - support to different types of aggregate functions (e.g., moving average, top ten)
- Data analysis by means of data mining techniques
  - various analysis types
  - significant algorithmic contribution

Elena Baralis
Politecnico di Torino

Key Performance Indicator (KPI)

- KPIs are measurable values that demonstrates how effectively a company is achieving key business objectives.
- They are used to periodically assess at multiple levels the performance of organizations and their success at reaching targets
  - high-level KPIs may focus on the overall performance of the business
  - low-level KPIs may focus on processes in specifics areas/departments (e.g, sales, marketing, HR).
- One of the most important aspects of KPIs is that they are a form of communication.
- Example KPIs: Days to deliver an order, number of new customers acquired, employee satisfaction, …

Elena Baralis
Politecnico di Torino

Data analysis tools

- Presentation
  - separate activity: data returned by a query may be rendered by means of different presentation tools
- Motivation search
  - Data exploration by means of progressive, “incremental” refinements (e.g., drill down)

Elena Baralis
Politecnico di Torino

Data visualization

Elena Baralis
Politecnico di Torino

Informative Dashboard

- A dashboard is a user interface that organizes and presents information in a way that is easy to read.
- It is a visual display of the most important information needed to achieve one or more objectives
- Dashboard are small and concise to allow monitoring relevant phenomena at a glance
- Visual Business Intelligence for enlightening analysis and communication

Elena Baralis
Politecnico di Torino

Data warehouse architectures

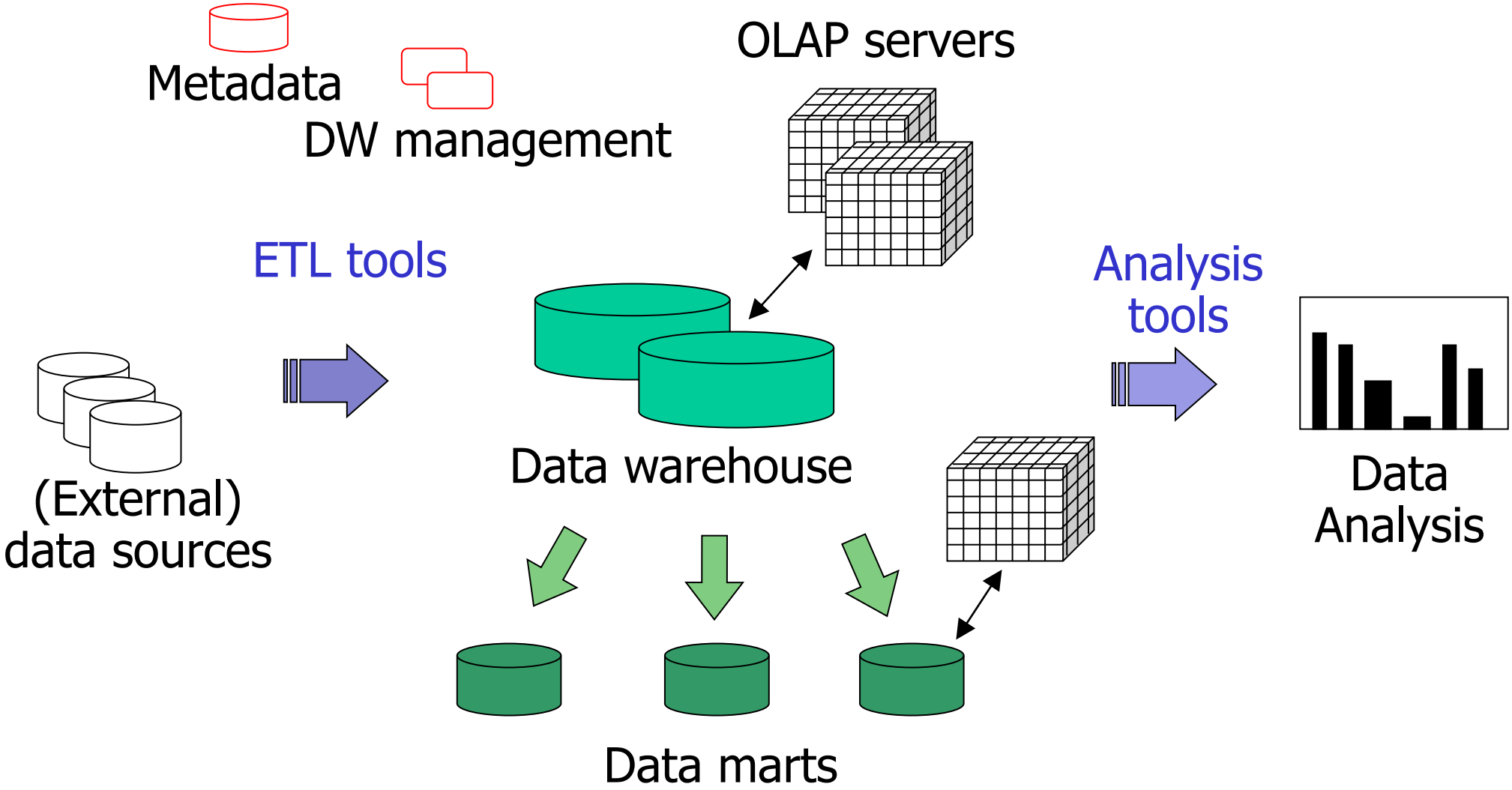
Elena Baralis
Politecnico di Torino

Data warehouse architectures

- Separation between transactional computing and data analysis
  - avoid one level architectures
- Architectures characterized by two or more levels
  - separate to a different extent data incoming into the data warehouse from analyzed data
  - more scalable

Elena Baralis
Politecnico di Torino

Data warehouse: architecture



Elena Baralis
Politecnico di Torino

Data warehouse and data mart

Company data warehouse: it contains *all* the information on the company business

- extensive functional modelling process
- design and implementation require a long time

Data mart: departimental information subset focused on a given subject

- two architectures
  - dependent, fed by the company data warehouse
  - independent, fed directly by the sources
- faster implementation
- requires careful design, to avoid subsequent data mart integration problems

Elena Baralis
Politecnico di Torino

Servers for Data Warehouses

- ROLAP (Relational OLAP) server
  - extended relational DBMS
    - compact representation for sparse data
  - SQL extensions for aggregate computation
  - specialized access methods which implement efficient OLAP data access
- MOLAP (Multidimensional OLAP) server
  - data represented in proprietary (multidimensional) matrix format
    - sparse data require compression
  - special OLAP primitives
- HOLAP (Hybrid OLAP) server
- NOSQL architectures

Elena Baralis
Politecnico di Torino

Extraction, Transformation and Loading (ETL)

- Prepares data to be loaded into the data warehouse
  - data extraction from (OLTP and external) sources
  - data cleaning
  - data transformation
  - data loading
- Performed
  - when the DW is first loaded
  - during periodical DW refresh

Elena Baralis
Politecnico di Torino

ETL process

- *Data extraction*: data acquisition from sources
- *Data cleaning*: techniques for improving data quality (correctness and consistency)
- *Data transformation*: data conversion from operational format to data warehouse format
- *Data loading:* update propagation to the data warehouse

Elena Baralis
Politecnico di Torino

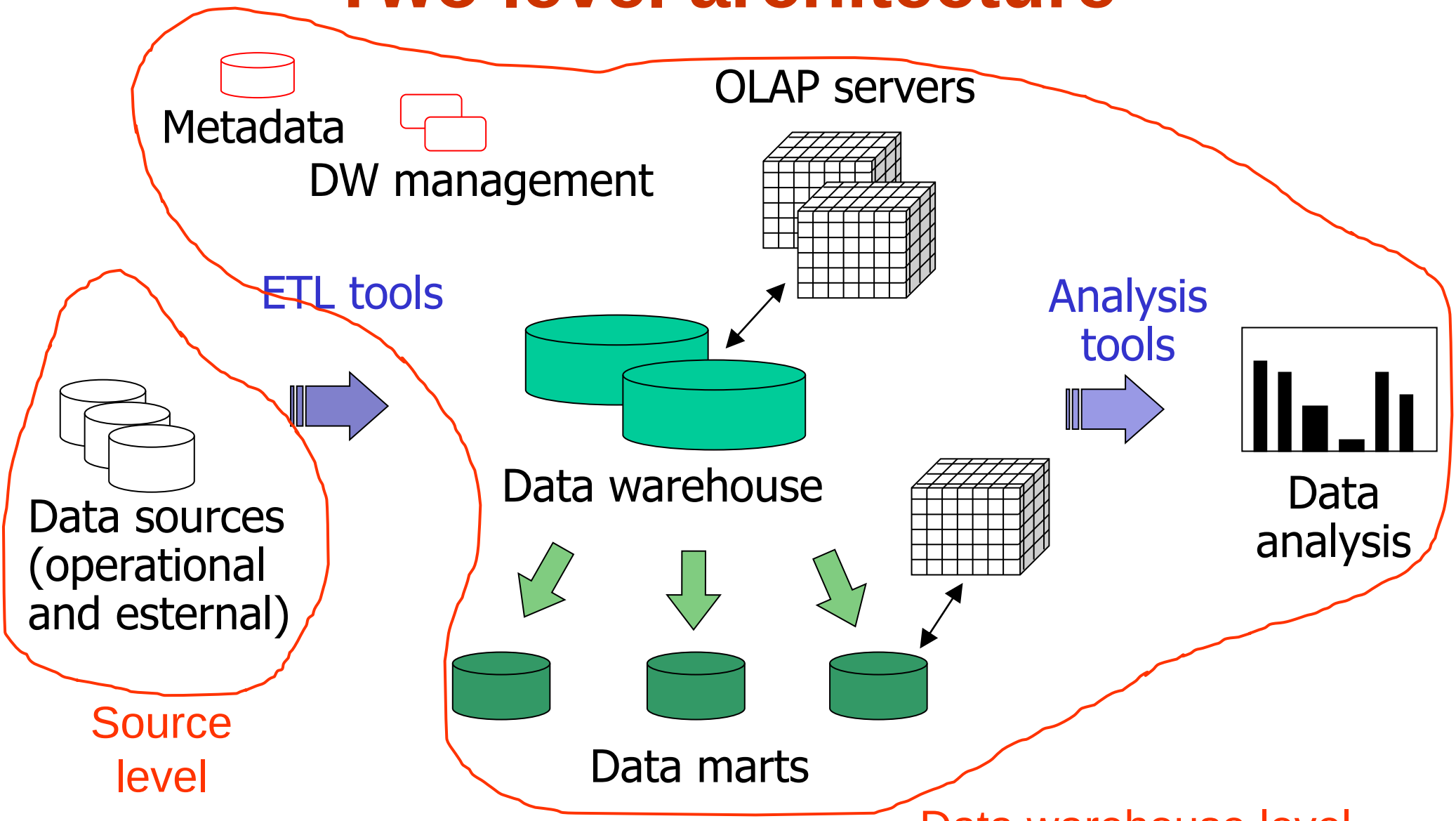
Metadata

metadata = data about data

- Different types of metadata:
  - for data transformation and loading: describe data sources and needed transformation operations
    - Useful using a common notation to represent data sources and data after transformation
    - CWMI (Common Warehouse Metadata Initiative): standard proposed by OMG to exchange data between DW tools and repository of metadata in heterogenous and distributed environments
  - for data management: describe the structure of the data in the data warehouse
    - also for materialized view
  - for query management: data on query structure and to monitor query execution
      - SQL code for the query
      - execution plan
      - memory and CPU usage

Elena Baralis
Politecnico di Torino

Two level architecture



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

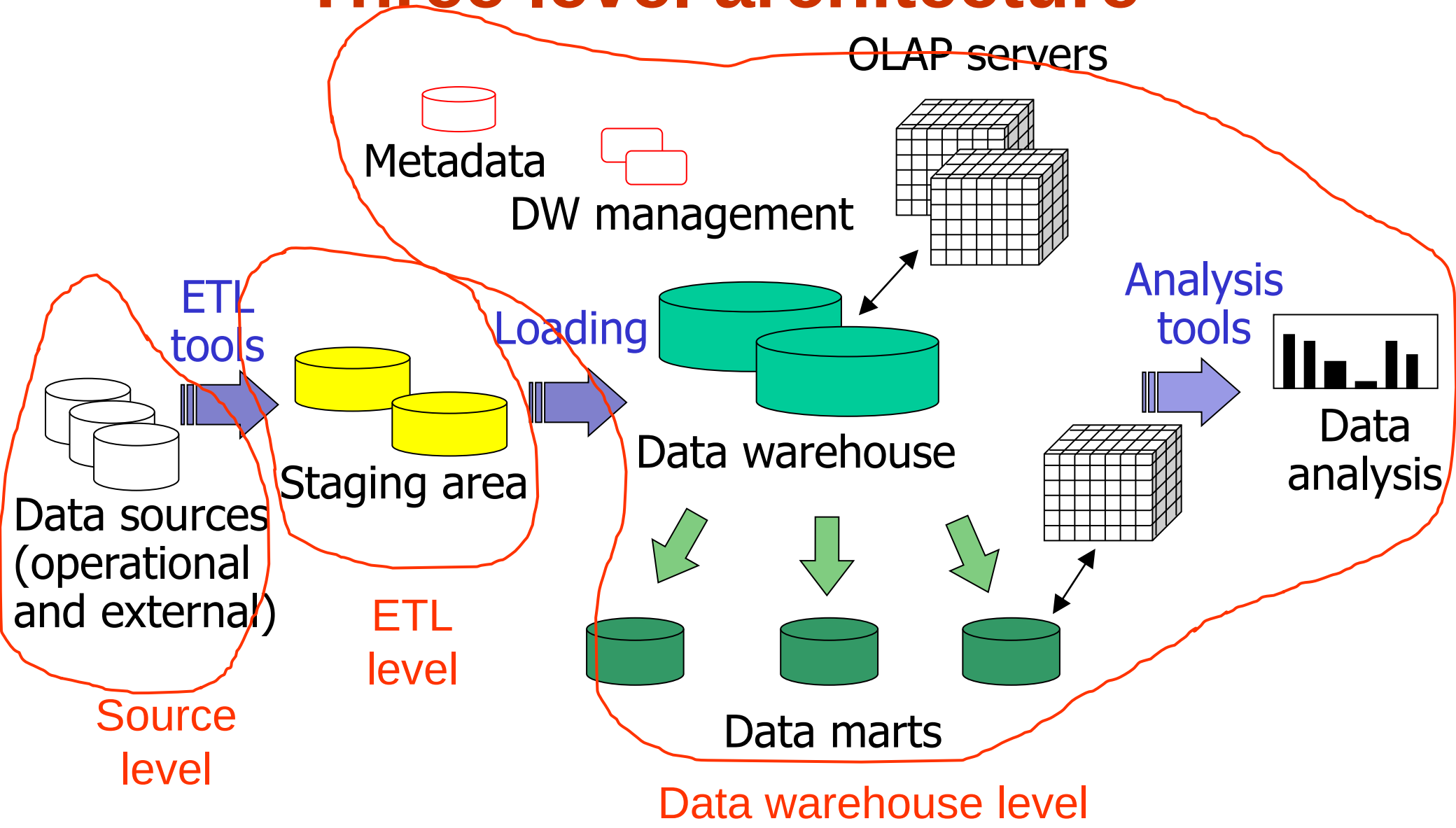
Elena Baralis
Politecnico di Torino

Two level architecture features

- Decoupling between source and DW data
  - management of external (not OLTP) data sources (e.g., text files)
  - data modelling suited for OLAP analysis
  - physical design tailored for OLAP load
- Easy management of different temporal granularity of operational and analytical data
- Partitioning between transactional and analytical load
- “On the fly” data transformation and cleaning (ETL)

Elena Baralis
Politecnico di Torino

Three level architecture



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Three level architecture features

- *Staging area:* buffer area allowing the separation between ET management and data warehouse loading
  - complex transformation and cleaning operations are eased
  - provides an integrated model of business data, still close to OLTP representation
  - sometime denoted as Operational Data Store (ODS)
- Introduces further redundancy
  - more disk space is required for data storage

Elena Baralis
Politecnico di Torino

Data warehouse design

Elena Baralis
Politecnico di Torino

Risk factors

- High user expectation
  - the data warehouse is *the* solution of the company's problems
- Data and OLTP process quality
  - incomplete or unreliable data
  - non integrated or non optimized business processes
- "Political" management of the project
  - cooperation with "information owners"
  - system acceptance by end users
  - deployment
    - appropriate training

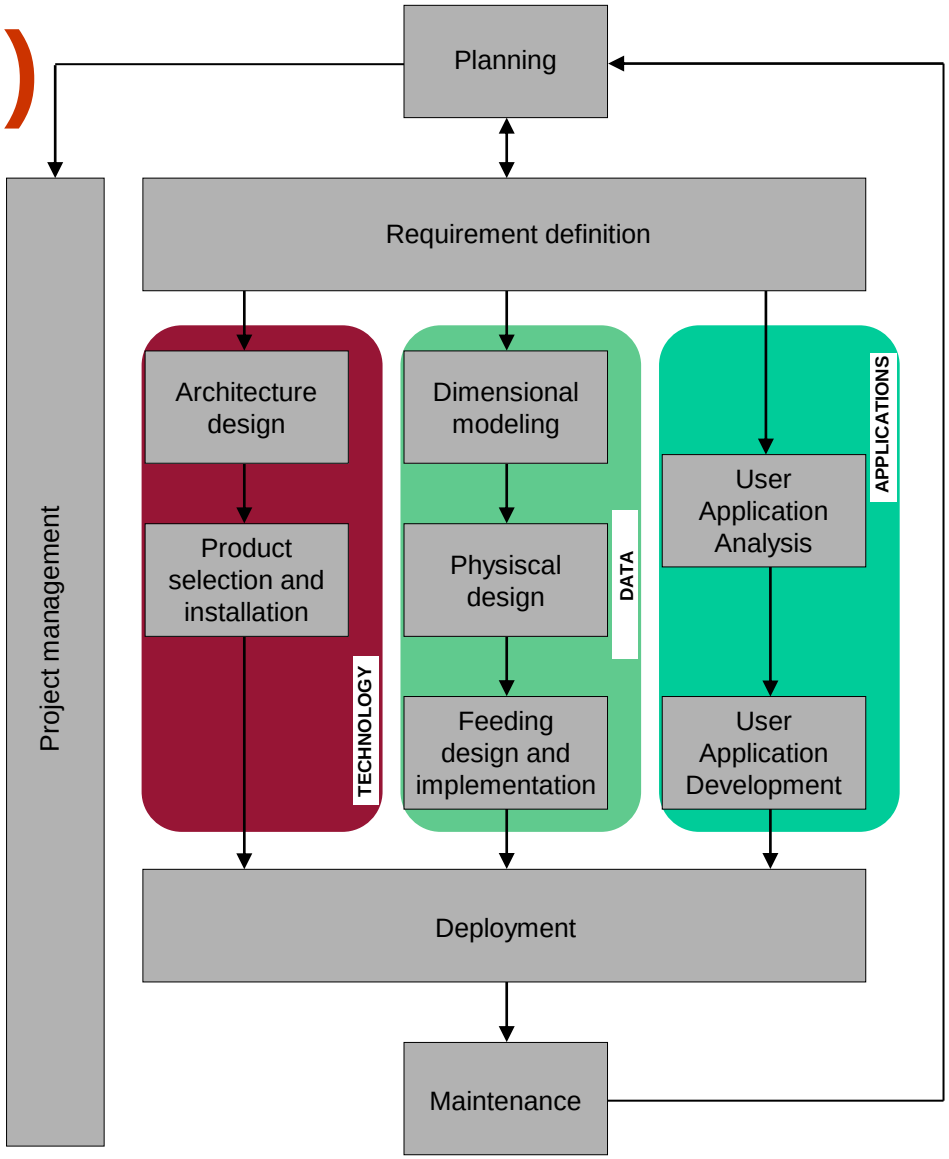
Elena Baralis
Politecnico di Torino

Data warehouse design

- Top-down approach
  - the data warehouse provides a global and complete representation of business data
  - significant cost and time consuming implementation
  - complex analysis and design tasks
- Bottom-up approach
  - incremental growth of the data warehouse, by adding data marts on specific business areas
  - separately focused on specific business areas
  - limited cost and delivery time
  - easy to perform intermediate checks

Elena Baralis
Politecnico di Torino

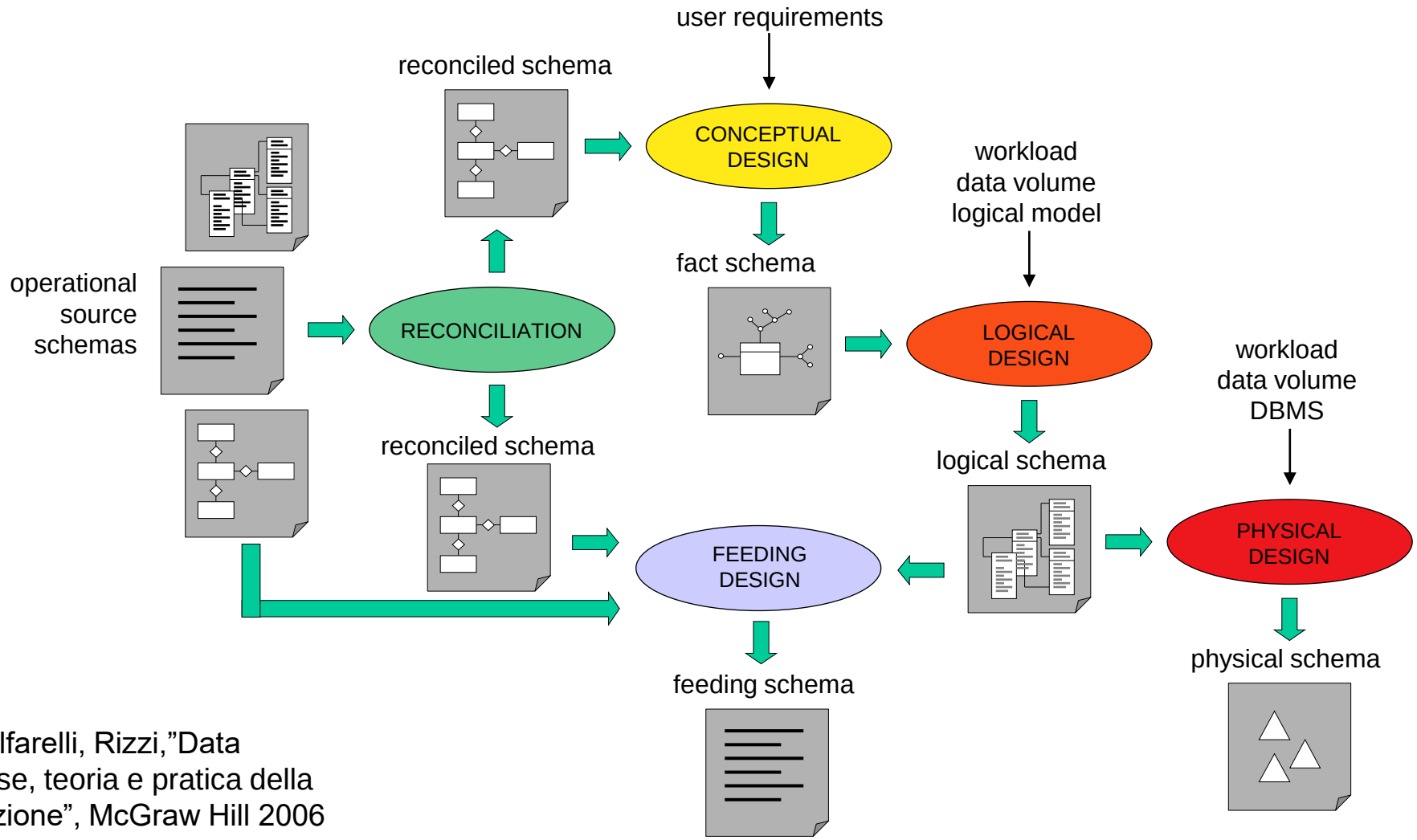
Business Dimensional Lifecycle (Kimball)



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Data mart design



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Requirement analysis

Elena Baralis
Politecnico di Torino

Requirement analysis

- It collects
  - data analysis requirements to be supported by the data mart
  - implementation constraints due to existing information systems
- Requirement sources
  - business users
  - operational system administrators
- The first selected data mart is
  - crucial for the company
  - feeded by (few) reliable sources

Elena Baralis
Politecnico di Torino

Application requirements

- Description of relevant events (facts)
  - each fact represents a category of events which are relevant for the company
    - examples: (in the CRM domain) complaints, services
  - characterized by descriptive dimensions (setting the granularity), history span, relevant measures
  - informations are gathered in a glossary
- Workload description
  - periodical business reports
  - queries expressed in natural language
    - example: number of complaints for each product in the last month

Elena Baralis
Politecnico di Torino

Structural requirements

- Feeding periodicity
- Available space for
  - data
  - derived data (indices, materialized views)
- System architecture
  - level number
  - dependent or independent data marts
- Deployment planning
  - start up
  - training

Elena Baralis
Politecnico di Torino

Conceptual design

Elena Baralis
Politecnico di Torino

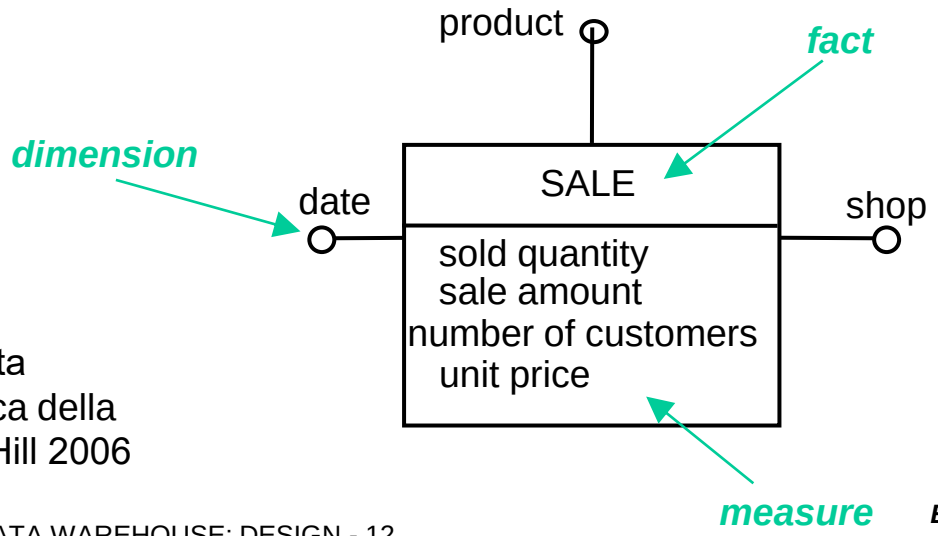
Conceptual design

- No currently adopted modeling formalism
  - ER model not adequate
- *Dimensional Fact Model* (Golfarelli, Rizzi)
  - graphical model supporting conceptual design
  - for a given fact, it defines a *fact schema* modelling
    - dimensions
    - hierarchies
    - measures
  - it provides design documentation both for requirement review with users, and after deployment

Elena Baralis
Politecnico di Torino

Dimensional Fact Model

- Fact
  - it models a set of relevant events (sales, shippings, complaints)
  - it evolves with time
- Dimension
  - it describes the analysis coordinates of a fact (e.g., each sale is described by the sale date, the shop and the sold product)
  - it is characterized by many, typically categorical, attributes
- Measure
  - it describes a numerical property of a fact (e.g., each sale is characterized by a sold quantity)
  - aggregates are frequently performed on measures

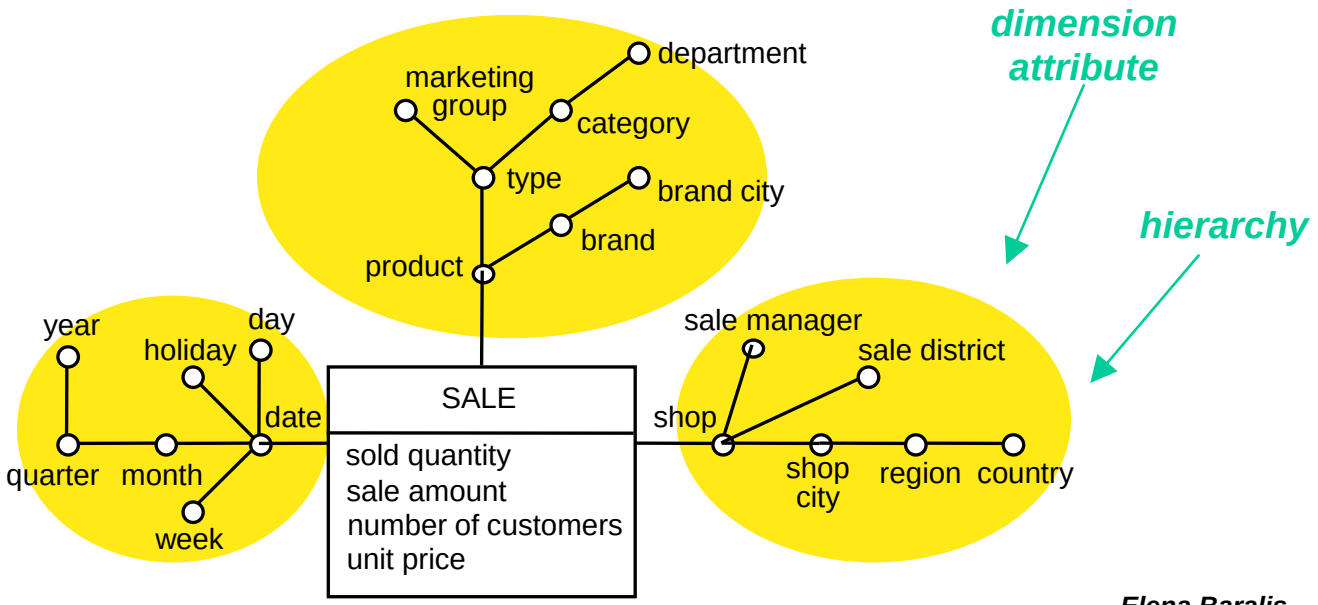


From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

DFM: Hierarchy

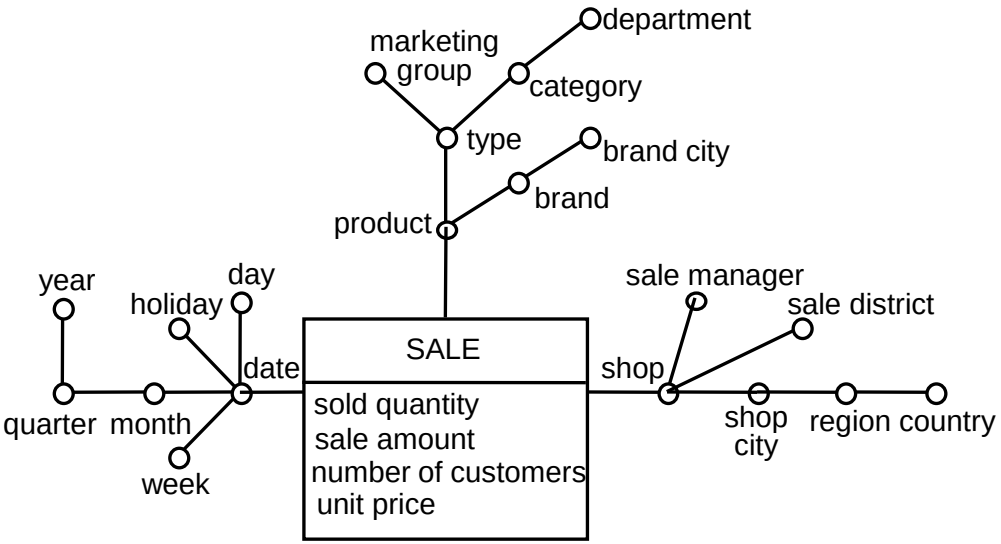
- Each dimension can have a set of associated attributes
- The attributes describe the dimension at different abstraction levels and can be structured as a hierarchy
- The hierarchy represents a generalization relationship among a subset of attributes in a dimension (e.g., geografic hierarchy for the shop dimension)
- The hierarchy represents a functional dependency (1:n relationship)

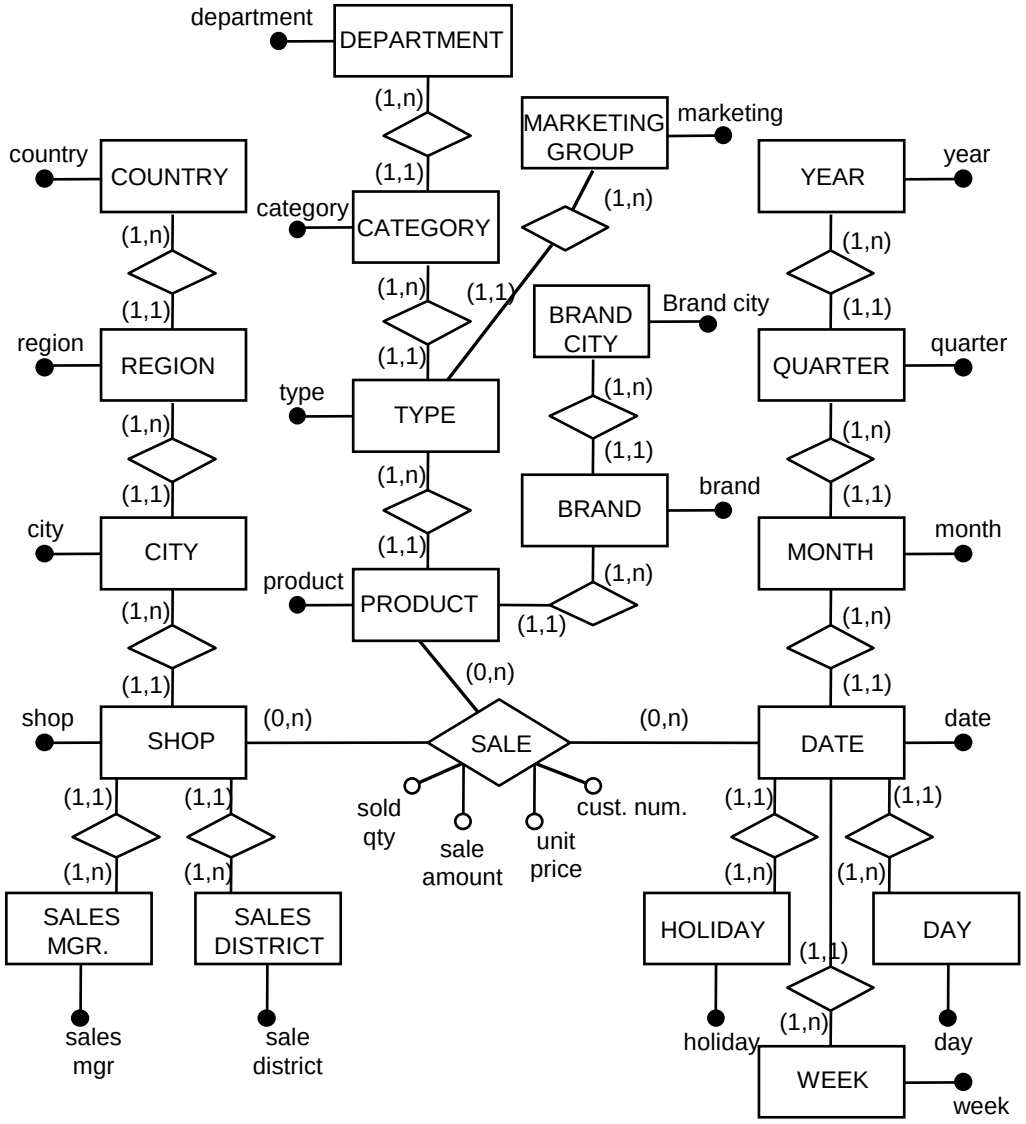


From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Comparison with ER

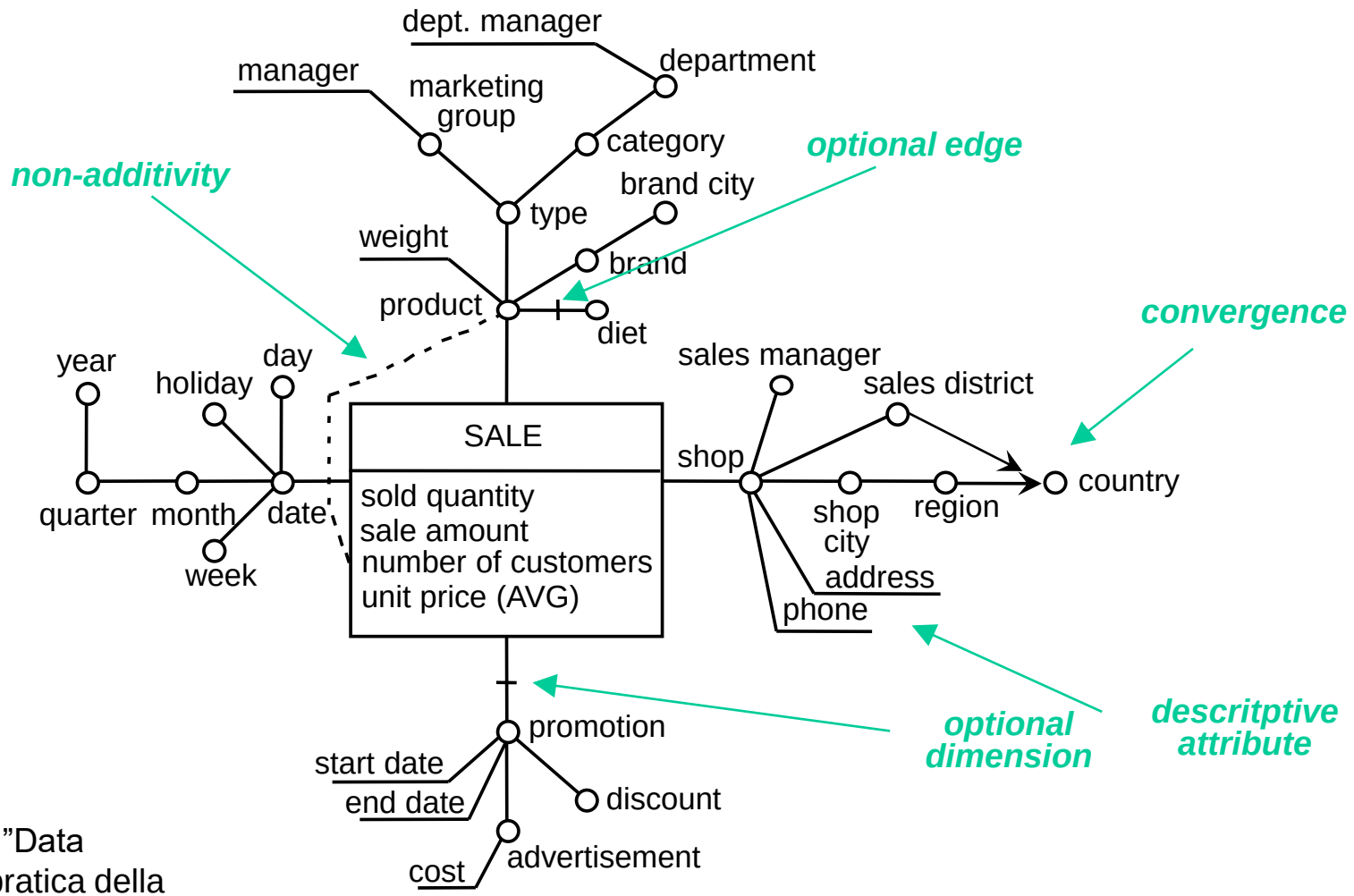




From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Advanced DFM



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Aggregation

- Aggregation computes measures with a coarser granularity than those in the original fact schema
  - detail reduction is usually obtained by climbing a hierarchy
  - standard aggregate operators: SUM, MIN, MAX, AVG, COUNT
- Measure characteristics
  - additive
  - not additive: cannot be aggregated along a given hierarchy by means of the SUM operator
  - not aggregable

Elena Baralis
Politecnico di Torino

Measure classification

- Stream measures
  - can be evaluated cumulatively at the end of a time period
  - can be aggregated by means of all standard operators
  - examples: sold quantity, sale amount
- Level measures
  - evaluated at a given time (snapshot)
  - not additive along the time dimension
  - examples: inventory level, account balance
- Unit measures
  - evaluated at a given time and expressed in relative terms
  - not additive along any dimension
  - examples: unit price of a product

Elena Baralis
Politecnico di Torino

Aggregate operators

category	type	product
home cleaning	washing powder	Brillo
		Sbianco
		Lucido
	soap	Manipulite
		Scent
food	milk	Latte F Slurp
		Latte U Slurp
		Yogurt Slurp
	soda	Bevimi
		Colissima

year	1999				2000			
quart.	I '99	II '99	III '99	IV '99	I '00	II '00	III '00	IV '00
	100	90	95	90	80	70	90	85
	20	30	20	10	25	30	35	20
	60	50	60	45	40	40	50	40
	15	20	25	30	15	15	20	10
	30	35	20	25	30	30	20	15
	90	90	85	75	60	80	85	60
	60	80	85	60	70	70	75	65
	20	30	40	35	30	35	35	20
	20	10	25	30	35	30	20	10
	50	60	45	40	50	60	45	40

year	1999				2000			
quart.	I'99	II'99	III'99	IV'99	I'00	II'00	III'00	IV'00
category								
home clean.	225	225	220	200	190	185	215	170
food	240	270	280	240	245	275	260	195

year	1999	2000
category		
home clean.	870	760
food	1030	975

year	1999	2000
category	type	
home cleaning	washing p.	670	605
	soap	200	155
food	milk	750	685
	soda	280	290

From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

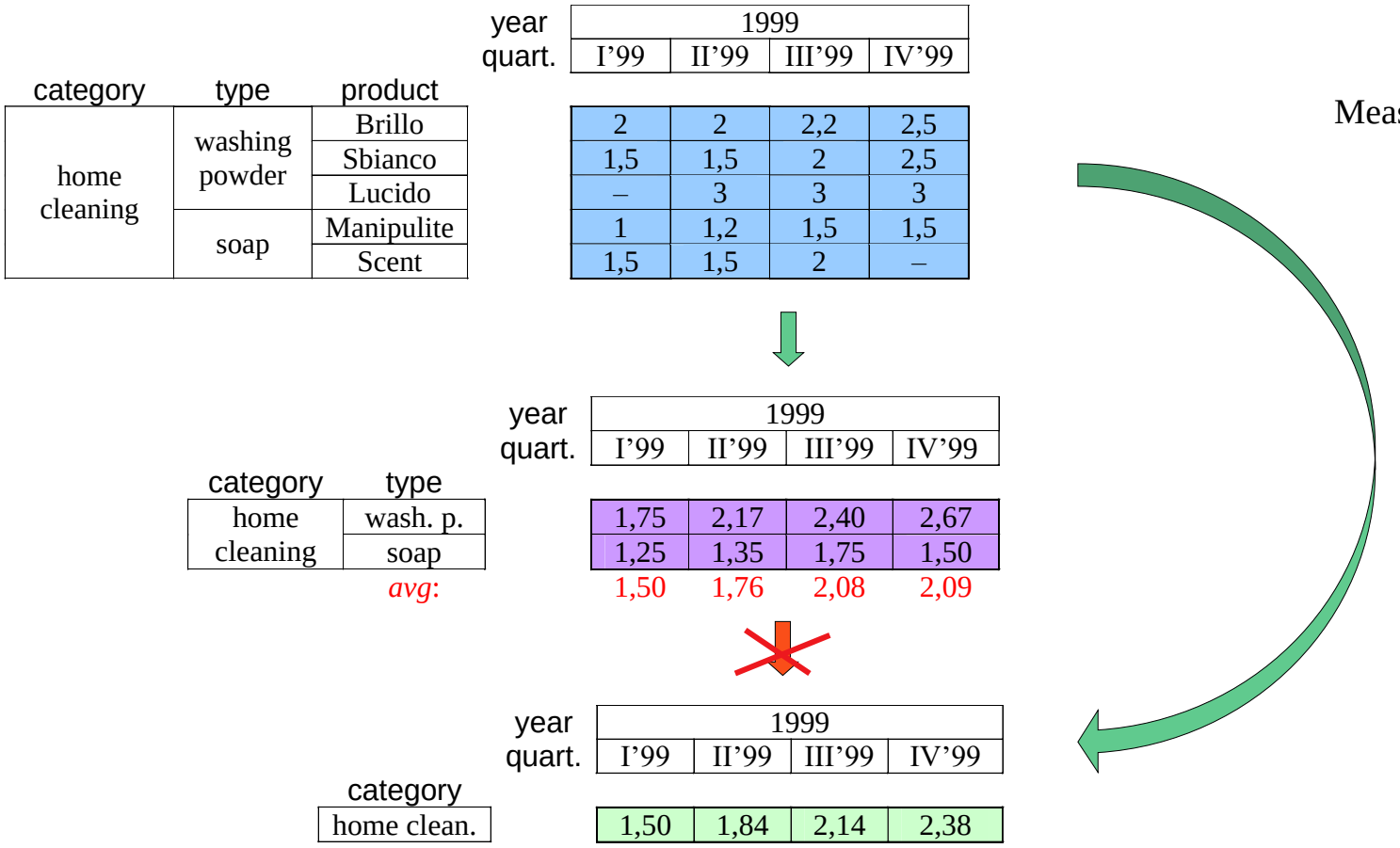
Elena Baralis
Politecnico di Torino

Aggregate operators

- Distributive
  - can always compute higher level aggregations from more detailed data
  - examples: sum, min, max

Elena Baralis
Politecnico di Torino

Non distributive operators



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

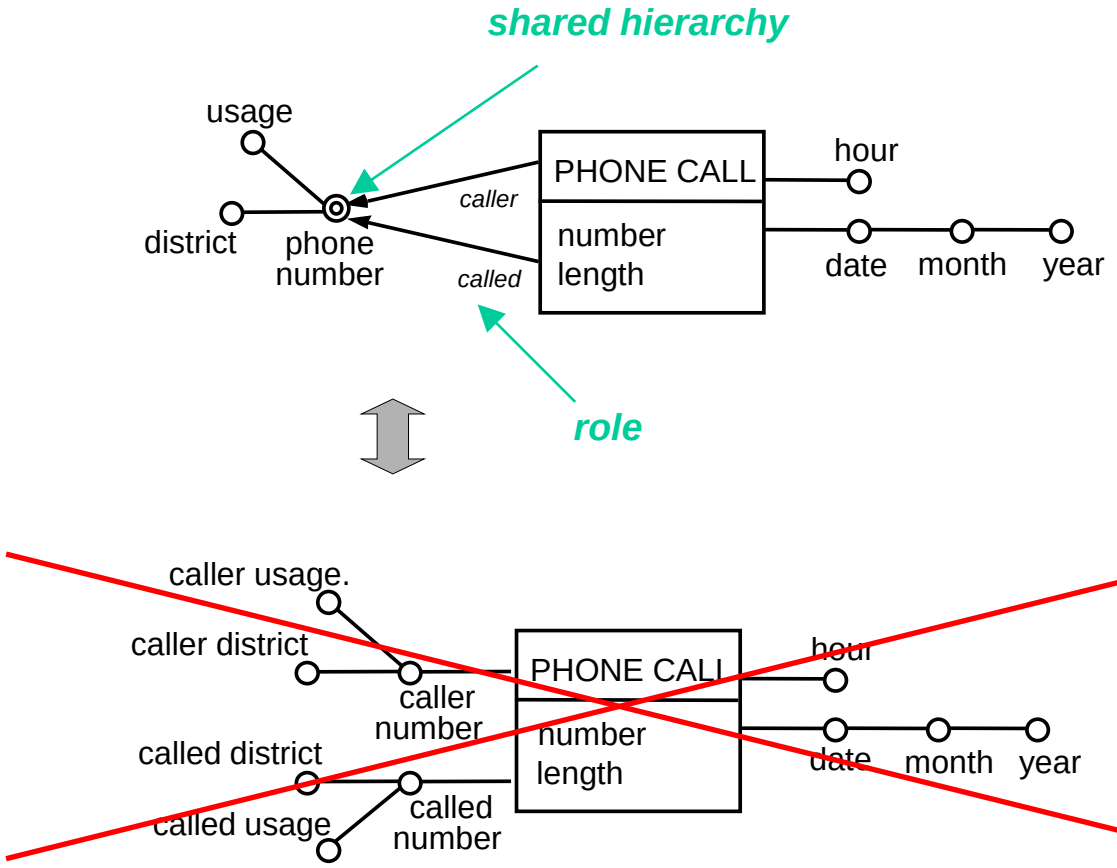
Elena Baralis
Politecnico di Torino

Aggregate operators

- Distributive
  - can always compute higher level aggregations from more detailed data
  - examples: sum, min, max
- Algebraic
  - can compute higher level aggregations from more detailed data *only* when supplementary support measures are available
  - examples: avg (it requires count)
- Olistic
  - *can not* compute higher level aggregations from more detailed data
  - examples: mode, median

Elena Baralis
Politecnico di Torino

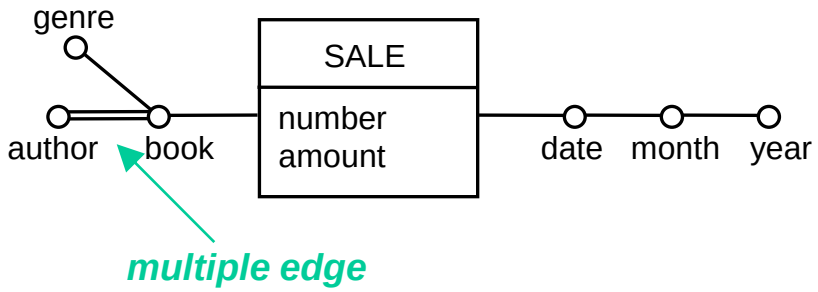
Advanced DFM

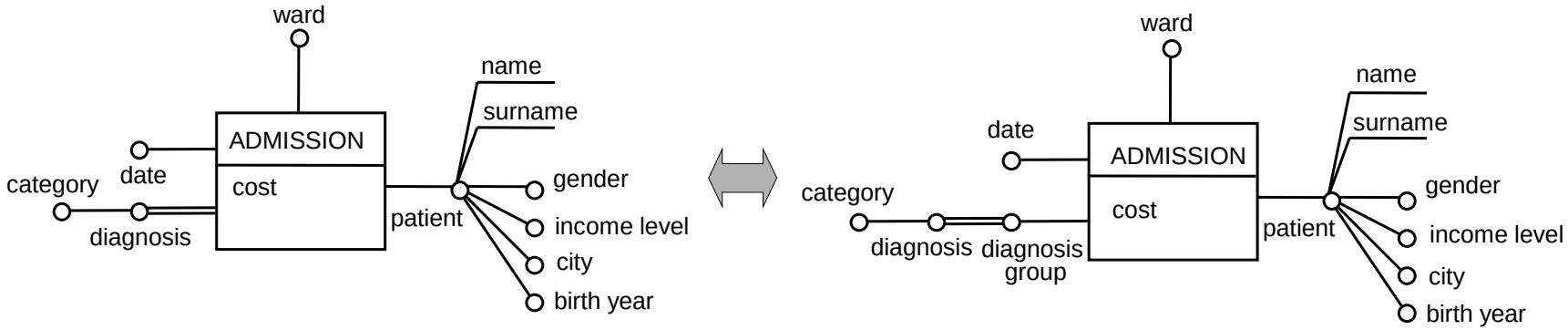


From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Advanced DFM



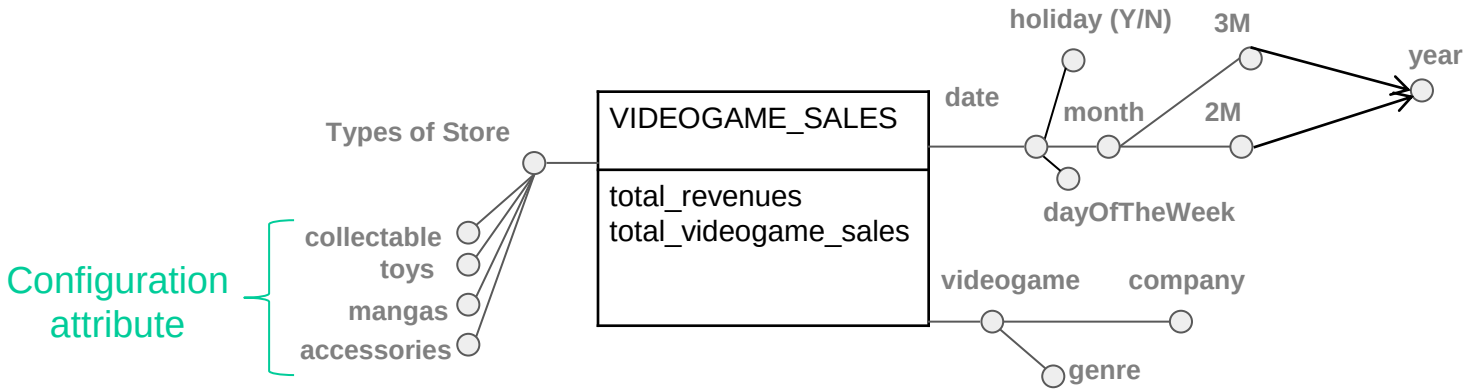


From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Configuration Attribute

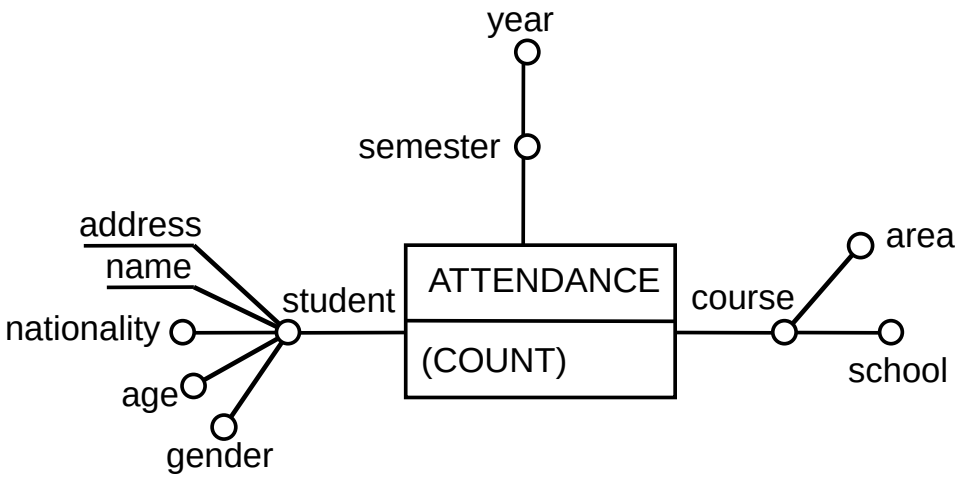
- Multi-valued categorical attribute
  - it can assume several values at the same time
  - characterized by a few distinct values (<=10)
- Representation by enumerating possible values
  - each attribute takes a boolean value (Y/N)
  - easier writing of complex queries



Elena Baralis
Politecnico di Torino

Factless fact schema

- Some events are not characterized by measures
  - empty (i.e., factless) fact schema
  - it records occurrence of an event
- Used for
  - counting occurred events (e.g., course attendance)
  - representing events not occurred (coverage set)



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Representing time

- Data modification over time is explicitly represented by event occurrences
  - time dimension
  - events stored as facts
- Also dimensions may change over time
  - modifications are typically slower
    - slowly changing dimension [Kimball]
  - examples: client demographic data, product description
  - if required, dimension evolution should be explicitly modeled

Elena Baralis
Politecnico di Torino

How to represent time (type I)

- Snapshot of the current value
  - data is overwritten with the current value
  - it overrides the past with the current situation
  - used when an explicit representation of the data change is not needed
  - example
    - customer Mario Rossi changes marital status after marriage
    - all his purchases correspond to the “married” customer

Elena Baralis
Politecnico di Torino

How to represent time (type II)

- Events are related to the temporally corresponding dimension value
  - after each state change in a dimension
    - a new dimension instance is created
    - new events are related to the new dimension instance
  - events are partitioned after the changes in dimensional attributes
  - example
    - customer Mario Rossi changes marital status after marriage
    - his purchases are partitioned in purchases performed by “unmarried” Mario Rossi and purchases performed by “married” Mario Rossi (a new instance of Mario Rossi)

Elena Baralis
Politecnico di Torino

How to represent time (type III)

- All events are mapped to a dimension value sampled at a given time
  - it requires the explicit management of dimension changes during time
    - the dimension schema is modified by introducing
      - two timestamps: validity start and validity end
      - a new attribute which allows identifying the sequence of modifications on a given instance (e.g., a “master” attribute pointing to the root instance)
    - each state change in the dimension requires the creation of a new instance

Elena Baralis
Politecnico di Torino

How to represent time (type III)

- Example
  - customer Mario Rossi changes marital status after marriage
  - validity end timestamp of first Mario Rossi instance is given by the marriage date
  - validity start timestamp of the new instance is the same day
  - purchases are partitioned as in type II
  - a new attribute allows tracking all changes of Mario Rossi instance

Elena Baralis
Politecnico di Torino

Workload

- Workload defined by
  - standard reports
  - approximate estimates discussed with users
- Actual workload difficult to evaluate at design time
  - if the data warehouse succeeds, user and query number may grow
  - query type may vary over time
- Data warehouse tuning
  - performed after system deployment
  - requires monitoring the actual system workload

Elena Baralis
Politecnico di Torino

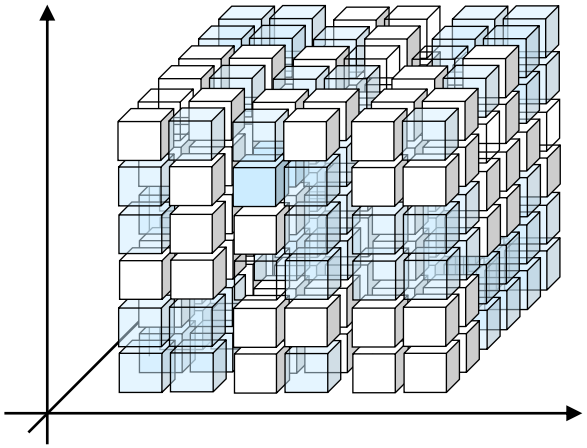
Data volume

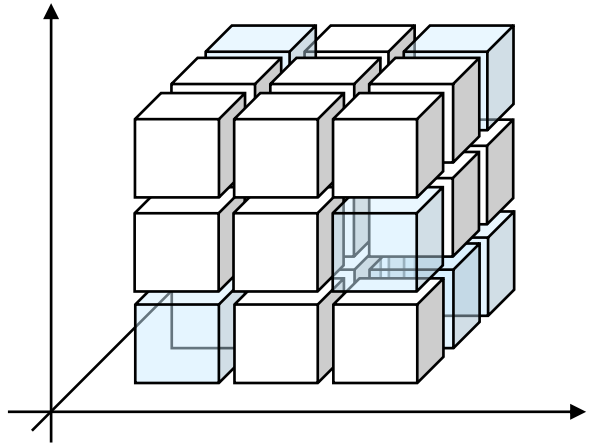
- Estimation of the space required by the data mart
  - for data
  - for derived data (indices, materialized views)
- To be considered
  - event cardinality for each fact
  - domain cardinality (number of distinct values) for hierarchy attributes
  - attribute length
- It depends on the temporal span of data storage
- Sparsity
  - occurred events are not all combinations of the dimension elements
  - example: the percentage of products actually sold in each shop and day is roughly 10% of all combinations

Elena Baralis
Politecnico di Torino

Sparsity

- It decreases with increasing data aggregation level
- May significantly affect the accuracy in estimating aggregated data cardinality





From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Logical design

Elena Baralis
Politecnico di Torino

Logical design

- We address the relational model (ROLAP)
  - inputs
    - conceptual fact schema
    - workload
    - data volume
    - system constraints
  - output
    - relational logical schema
- Based on different principles with respect to traditional logical design
  - data redundancy
  - table denormalization

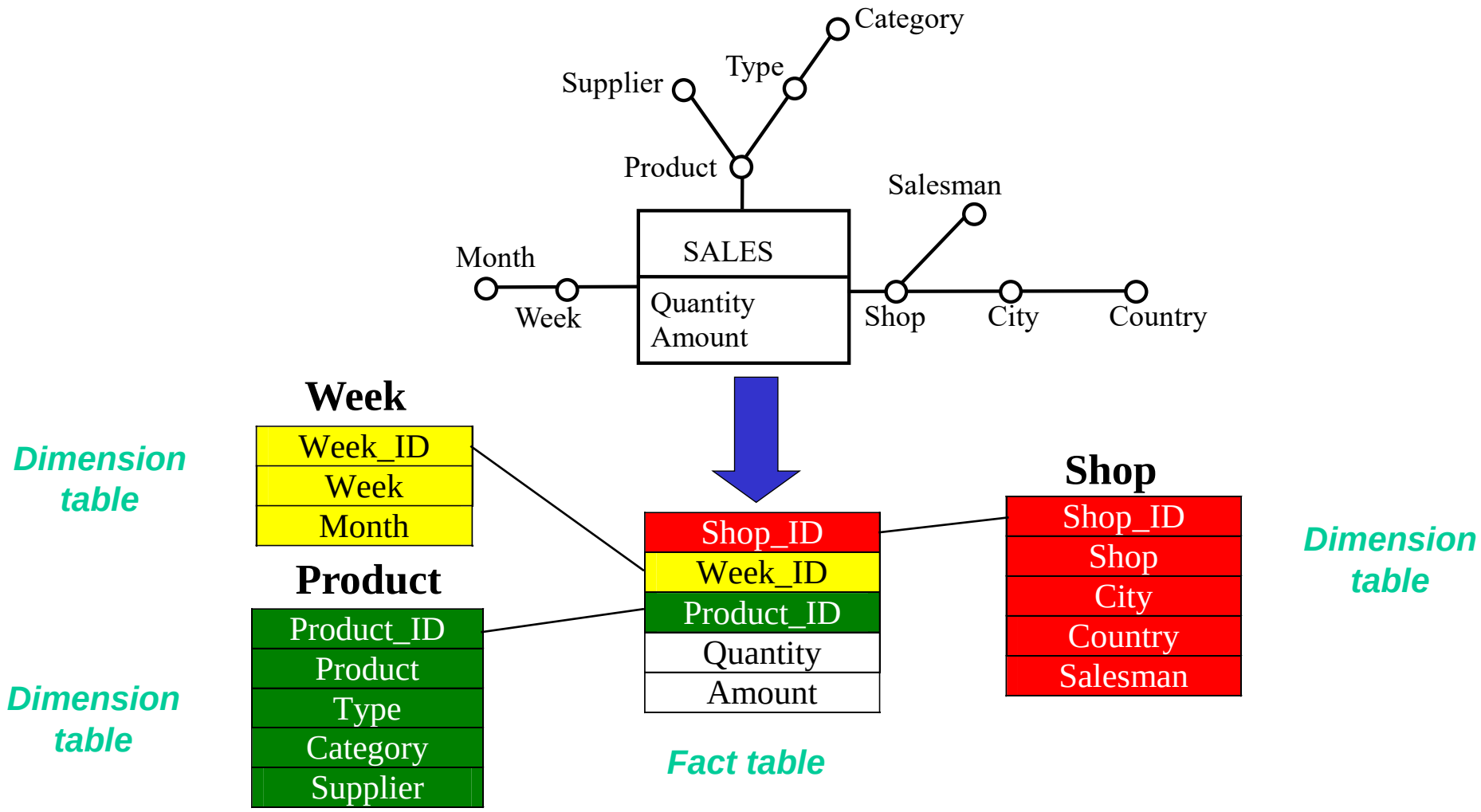
Elena Baralis
Politecnico di Torino

Star schema

- Dimensions
  - one table for each dimension
  - surrogate (generated) primary key
  - it contains all dimension attributes
  - hierarchies are not explicitly represented
    - all attributes in a table are at the same level
  - totally denormalized representation
    - it causes data redundancy
- Facts
  - one fact table for each fact schema
  - primary key composed by foreign keys of all dimensions
  - measures are attributes of the fact table

Elena Baralis
Politecnico di Torino

Star schema



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

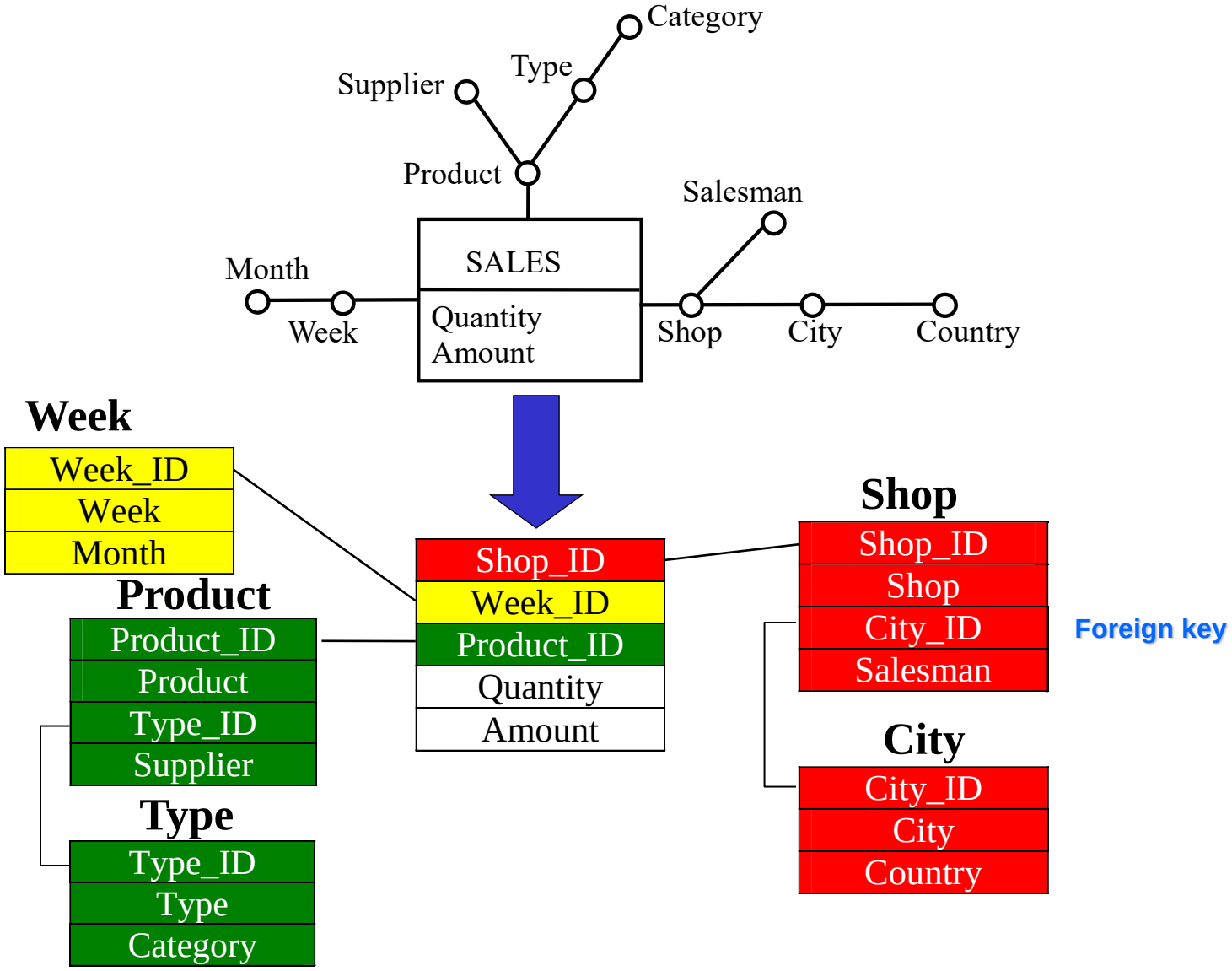
Elena Baralis
Politecnico di Torino

Snowflake schema

- Some functional dependencies are separated, by partitioning dimension data in several tables
  - a new table separates two branches of a dimensional hierarchy (hierarchy is cut on a given attribute)
  - a new foreign key correlates the dimension with the new table
- Decrease in space required for storing the dimension
  - decrease is frequently not significant
- Increase in cost for reading entire dimension
  - one or more joins are needed

Elena Baralis
Politecnico di Torino

Snowflake schema



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

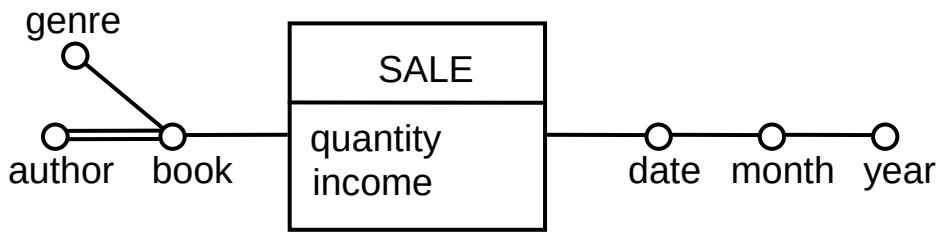
Elena Baralis
Politecnico di Torino

Star or snowflake?

- The snowflake schema is usually not recommended
  - storage space decrease is rarely beneficial
    - most storage space is consumed by the fact table (difference with dimensions is several orders of magnitude)
  - cost of join execution may be significant
- The snowflake schema is rarely used in the data mart design

Elena Baralis
Politecnico di Torino

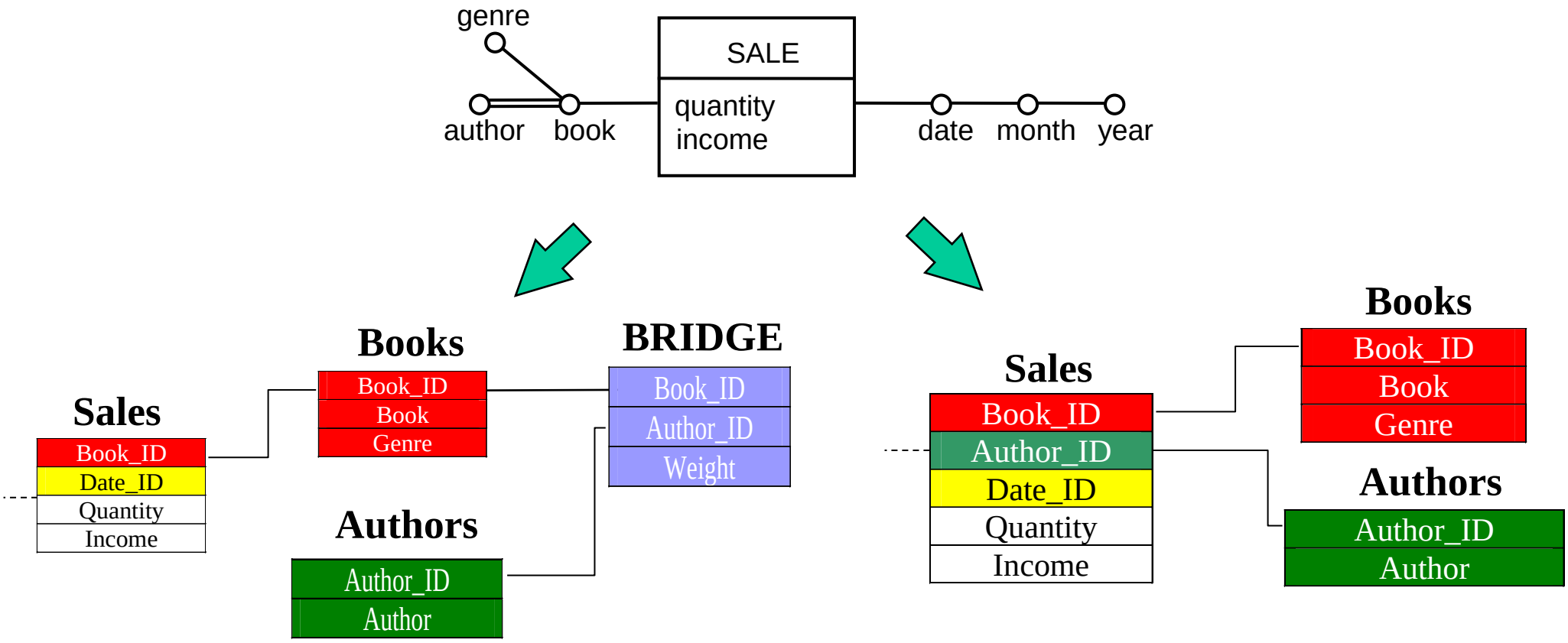
Multiple edges



- Implementation techniques
  - bridge table
    - new table which models many to many relationship
    - new attribute weighting the contribution of tuples in the relationship
  - push down
    - multiple edge integrated in the fact table
    - new corresponding dimension in the fact table

Elena Baralis
Politecnico di Torino

Multiple edges



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Multiple edges

- Queries
  - Weighted query: consider the weight of the multiple edge
    - example: author income
    - by using bridge table:

```
SELECT Author_ID, SUM(Income*Weight)
...
group by Author_ID
```

  - Impact query: do not consider the weight of the multiple edge
    - example: book copies sold for each author
    - by using bridge table:

```
SELECT Author_ID, SUM(Quantity)
...
group by Author_ID
```


Elena Baralis
Politecnico di Torino

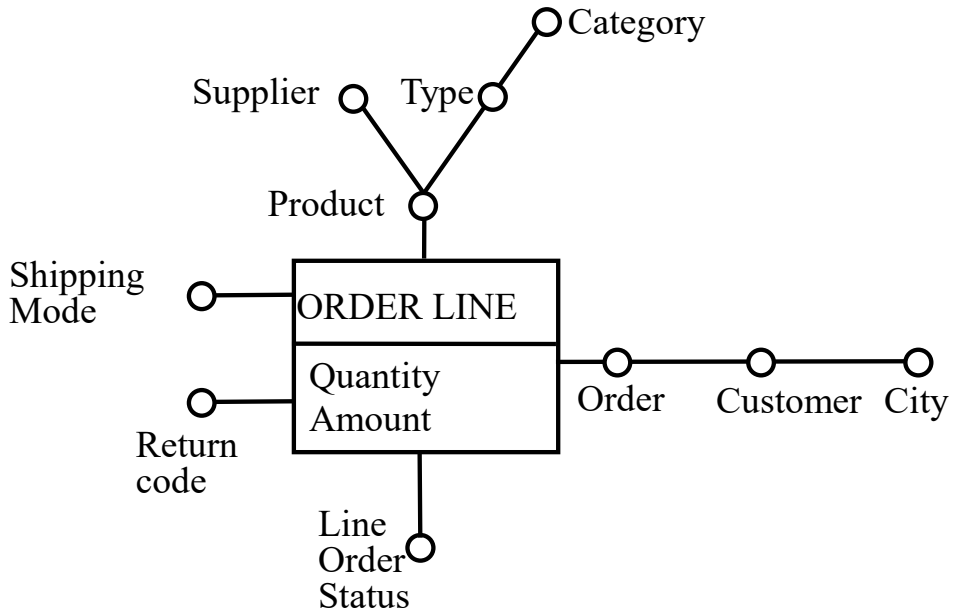
Multiple edges

- Comparison
  - weight is explicited in the bridge table, but wired in the fact table for push down
    - (push down) hard to perform impact queries
    - (push down) weight is computed when feeding the DW
    - (push down) weight modifications are hard
  - push down causes significant redundancy in the fact table
  - query execution cost is lower for push down
    - less joins

Elena Baralis
Politecnico di Torino

Degenerate dimensions

- Dimensions with a single attribute



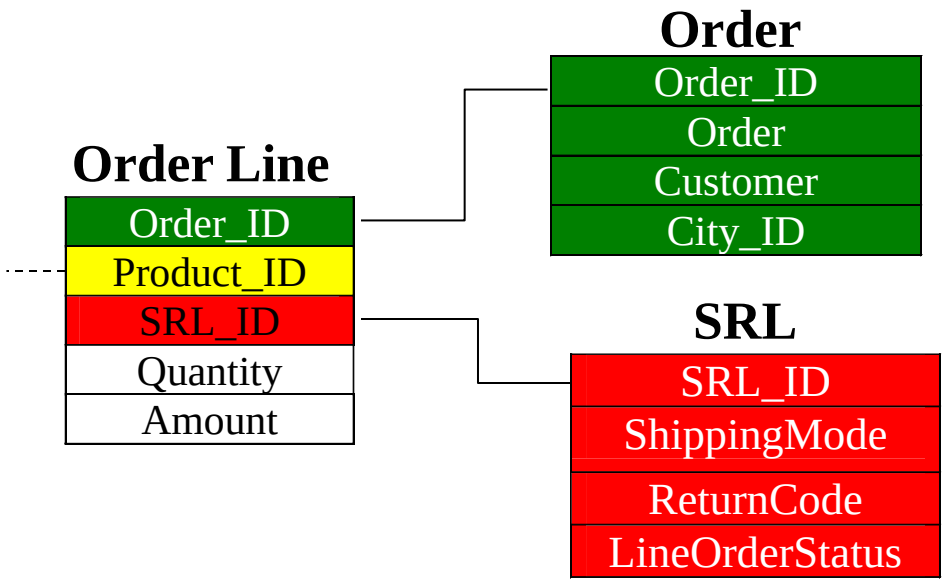
Elena Baralis
Politecnico di Torino

Degenerate dimensions

- Implementations
  - Integration into the fact table
    - for attributes with a (very) small size
  - junk dimension
    - single dimension containing several degenerate dimensions
    - no functional dependencies among attributes in the junk dimension
      - all attribute value combinations are allowed
      - feasible only for attribute domains with small cardinality

Elena Baralis
Politecnico di Torino

Junk dimension



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Data warehouse
Data analysis

Elena Baralis
Politecnico di Torino

Data analysis

- OLAP analysis: complex aggregate function computation
  - support to different types of aggregate functions (e.g., moving average, top ten)
- Comparison operations, exploited to compare business trends (example: sale figure comparison for different time periods)
  - difficult by exploiting plain SQL
- Data analysis by means of data mining techniques

Elena Baralis
Politecnico di Torino

User interface

Users may query the data warehouse by means of various tools:

- controlled query environments
- query and report generation tools
- data mining tools

Elena Baralis
Politecnico di Torino

Controlled query environment

- It encompasses
  - complex queries with predefined structure (usually parametric)
  - ad hoc analysis procedures
  - predefined reports
- Techniques and knowledge of a specific economic area may be exploited
- It requires ad hoc code development
  - stored procedures, application packages, predefined joins and aggregations
  - flexible tools for report management are available, which allow defining
    - report layout
    - publication periodicity
    - distribution list

Elena Baralis
Politecnico di Torino

Ad hoc query environment

- Arbitrary OLAP queries may be defined
- Queries are designed on demand by users
  - query is defined by point and click techniques, which automatically generate SQL instructions
  - (typically) complex queries may be defined
  - spreadsheet is the user interface paradigm
- An OLAP session allows successive refinements of the same query
- Used when predefined reports are not enough

Elena Baralis
Politecnico di Torino

OLAP analysis

Elena Baralis
Politecnico di Torino

OLAP analysis

- Available query operations
  - roll up, drill down
  - slice and dice
  - (table) pivot
  - sorting
- Operations may be
  - used together in the same query
  - exploited in sequence to refine the same query which builds up the OLAP session

Elena Baralis
Politecnico di Torino

Roll up

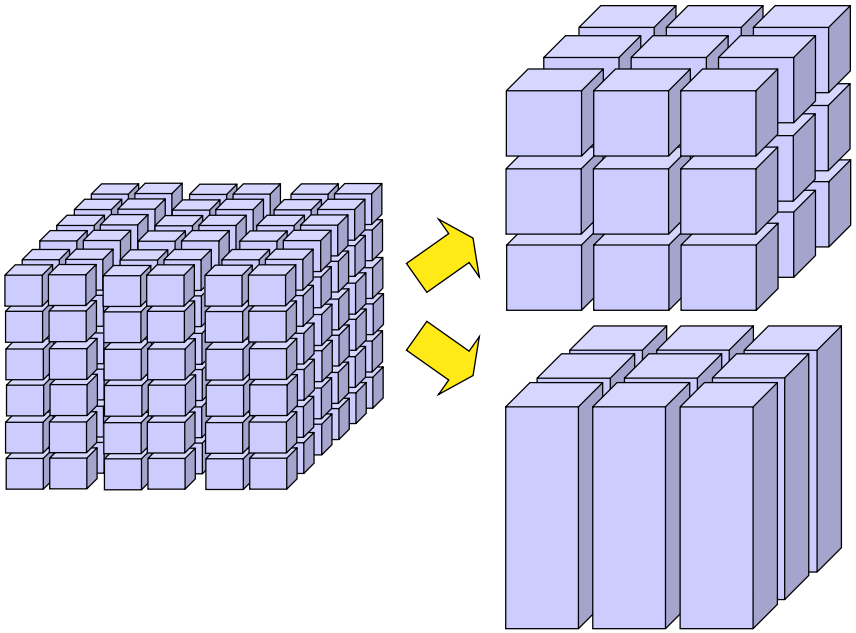
- Data detail reduction by
  - decreasing detail in a dimension, by climbing up a hierarchy
    - example

      group by store, month $\rightarrow$ group by city, month
  - dropping a whole dimension
    - example

      group by product, city $\rightarrow$ group by product

Elena Baralis
Politecnico di Torino

Roll up



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Roll up

Metrics / Customer Region / Month	Dollar Sales										
	North-East	Mid-Atlantic	South-East	Central	South	North-West	South-West	England	France	Germany	Canada
Jan 97	$ 620	$ 753	$ 30	$ 660	$ 2.405	$ 1.312	$ 440	$ 1.002	$ 1.002	$ 383	$ 210
Feb 97	$ 258	$ 252	$ 800	$ 975	$ 160	$ 582	$ 744	$ 310	$ 799	$ 118	$ 357
Mar 97	$ 648	$ 244	$ 148	$ 250	$ 1.085	$ 2.961	$ 650	$ 1.240	$ 119	$ 142	$ 96
Apr 97	$ 787	$ 588	$ 447	$ 486	$ 226	$ 506	$ 601	$ 119	$ 550	$ 85	
May 97	$ 1.350	$ 245	$ 936	$ 159	$ 664	$ 626	$ 107	$ 135	$ 200	$ 177	$ 230
Jun 97	$ 842	$ 582	$ 1.281	$ 937	$ 240	$ 774	$ 176	$ 1.139	$ 652	$ 254	$ 745
Jul 97	$ 652	$ 690	$ 486	$ 1.293	$ 605	$ 303	$ 818	$ 103	$ 124	$ 173	$ 66
Aug 97	$ 1.783	$ 304	$ 1.032	$ 170	$ 398	$ 356	$ 432	$ 190	$ 241	$ 407	$ 259
Sep 97	$ 581	$ 778	$ 3.558	$ 587	$ 440	$ 1.652	$ 1.071	$ 315	$ 210	$ 202	
Oct 97	$ 2.291	$ 1.840	$ 600	$ 656	$ 1.300	$ 718	$ 1.210	$ 427	$ 220	$ 520	$ 65
Nov 97	$ 39	$ 1.602	$ 1.082	$ 1.187	$ 842	$ 759	$ 745	$ 232	$ 101	$ 1.037	$ 37
Dec 97	$ 381	$ 1.588	$ 343	$ 118	$ 1.459	$ 635	$ 2.021	$ 259	$ 210	$ 119	$ 189
Jan 98	$ 311	$ 1.174	$ 2.634	$ 3.130	$ 954	$ 2.083	$ 1.351	$ 747	$ 426	$ 447	$ 1.141
Feb 98	$ 2.518	$ 702	$ 1.123	$ 1.336	$ 1.227	$ 3.887	$ 545	$ 268	$ 277	$ 282	
Mar 98	$ 2.459	$ 1.523	$ 1.178	$ 4.708	$ 1.420	$ 3.514	$ 1.948	$ 1.705	$ 276	$ 1.168	$ 63
Apr 98	$ 407	$ 841	$ 524	$ 712	$ 133	$ 2.486	$ 49	$ 390	$ 1.298	$ 221	$ 46
May 98	$ 667	$ 1.721	$ 440	$ 148	$ 80	$ 1.310	$ 303	$ 104	$ 657	$ 65	
Jun 98	$ 699	$ 1.096	$ 898	$ 353	$ 902	$ 839		$ 230	$ 155	$ 105	$ 75
Jul 98	$ 586	$ 1.897	$ 412	$ 226	$ 406	$ 361	$ 1.628	$ 267	$ 1.011	$ 41	$ 184
Aug 98	$ 894	$ 326	$ 792	$ 1.832	$ 1.199	$ 295	$ 1.816	$ 277	$ 102	$ 118	$ 115
Sep 98	$ 338	$ 3.179	$ 505	$ 427	$ 99	$ 2.976	$ 885	$ 135	$ 85	$ 1.110	$ 510
Oct 98	$ 544	$ 413	$ 1.467	$ 209	$ 679	$ 706	$ 556	$ 480	$ 485	$ 99	$ 160
Nov 98	$ 671	$ 459	$ 1.471	$ 2.066	$ 701	$ 716	$ 986	$ 1.127	$ 154	$ 440	$ 361
Dec 98	$ 836	$ 2.096	$ 1.726	$ 3.642	$ 395	$ 1.740	$ 1.943	$ 1.143	$ 366	$ 307	$ 118

Metrics / Customer Region / Quarter	Dollar Sales										
	North-East	Mid-Atlantic	South-East	Central	South	North-West	South-West	England	France	Germany	Canada
Q1 1997	$ 1.526	$ 1.249	$ 978	$ 1.885	$ 3.650	$ 4.855	$ 1.834	$ 2.552	$ 1.920	$ 643	$ 663
Q2 1997	$ 2.979	$ 1.415	$ 2.664	$ 1.582	$ 1.130	$ 1.906	$ 884	$ 1.393	$ 1.402	$ 516	$ 975
Q3 1997	$ 3.016	$ 1.772	$ 5.076	$ 2.050	$ 1.443	$ 2.311	$ 2.321	$ 608	$ 575	$ 782	$ 325
Q4 1997	$ 2.711	$ 5.030	$ 2.025	$ 1.961	$ 3.601	$ 2.112	$ 3.976	$ 918	$ 531	$ 1.676	$ 291
Q1 1998	$ 5.288	$ 3.399	$ 4.935	$ 9.174	$ 3.601	$ 9.484	$ 3.844	$ 2.720	$ 979	$ 1.897	$ 1.204
Q2 1998	$ 1.773	$ 3.658	$ 1.862	$ 1.213	$ 1.115	$ 4.635	$ 352	$ 724	$ 2.110	$ 391	$ 121
Q3 1998	$ 1.818	$ 5.402	$ 1.709	$ 2.485	$ 1.704	$ 3.632	$ 4.329	$ 679	$ 1.198	$ 1.269	$ 809
Q4 1998	$ 2.051	$ 2.968	$ 4.664	$ 5.917	$ 1.775	$ 3.162	$ 3.485	$ 2.750	$ 1.005	$ 846	$ 639

From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Roll up

Category	Year	Metrics / Customer Region	Dollar Sales: North-East	Mid-Atlantic	South-East	Central	South	North-West	South-West	England	France	Germa
Electronics	1997		$ 138	$ 1.774	$ 384	$ 138	$ 2.346	$ 2.554	$ 2.184	$ 566	$ 199	$
	1998		$ 1.184	$ 4.529	$ 1.892	$ 7.232	$ 651	$ 9.488	$ 476	$ 2.683	$ 462	$ 7
Food	1997		$ 759	$ 682	$ 729	$ 262	$ 588	$ 469	$ 807	$ 156	$ 615	$ 1
	1998		$ 538	$ 925	$ 959	$ 677	$ 213	$ 1.503	$ 261	$ 165	$ 175	$ 1
Gifts	1997		$ 2.532	$ 1.355	$ 1.854	$ 1.413	$ 2.535	$ 2.132	$ 1.904	$ 908	$ 375	$ 1.0
	1998		$ 1.955	$ 2.785	$ 2.800	$ 2.695	$ 1.813	$ 2.844	$ 1.778	$ 1.158	$ 717	$ 6
Health & Beauty	1997		$ 624	$ 640	$ 1.317	$ 647	$ 588	$ 754	$ 654	$ 143	$ 292	$ 3
	1998		$ 611	$ 887	$ 566	$ 382	$ 499	$ 1.162	$ 1.044	$ 273	$ 72	
Household	1997		$ 5.354	$ 4.112	$ 5.410	$ 4.446	$ 3.058	$ 3.974	$ 2.654	$ 3.545	$ 2.875	$ 1.9
	1998		$ 5.787	$ 5.320	$ 5.416	$ 6.812	$ 4.334	$ 5.008	$ 7.588	$ 2.139	$ 3.649	$ 2.7
Kid's Korner	1997		$ 201	$ 398	$ 485	$ 186	$ 409	$ 323	$ 396	$ 105	$ 34	$
	1998		$ 247	$ 422	$ 441	$ 380	$ 221	$ 592	$ 290	$ 198	$ 19	$
Travel	1997		$ 624	$ 505	$ 564	$ 386	$ 300	$ 978	$ 416	$ 48	$ 38	
	1998		$ 608	$ 559	$ 1.096	$ 611	$ 464	$ 316	$ 573	$ 257	$ 198	$

Category	Year	Metrics / Dollar Sales
Electronics	1997	$ 10.616
	1998	$ 29.299
Food	1997	$ 5.300
	1998	$ 5.638
Gifts	1997	$ 16.315
	1998	$ 20.047
Health & Beauty	1997	$ 6.042
	1998	$ 5.665
Household	1997	$ 38.383
	1998	$ 50.391
Kid's Korner	1997	$ 2.559
	1998	$ 2.943
Travel	1997	$ 4.497
	1998	$ 4.792

From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Drill down

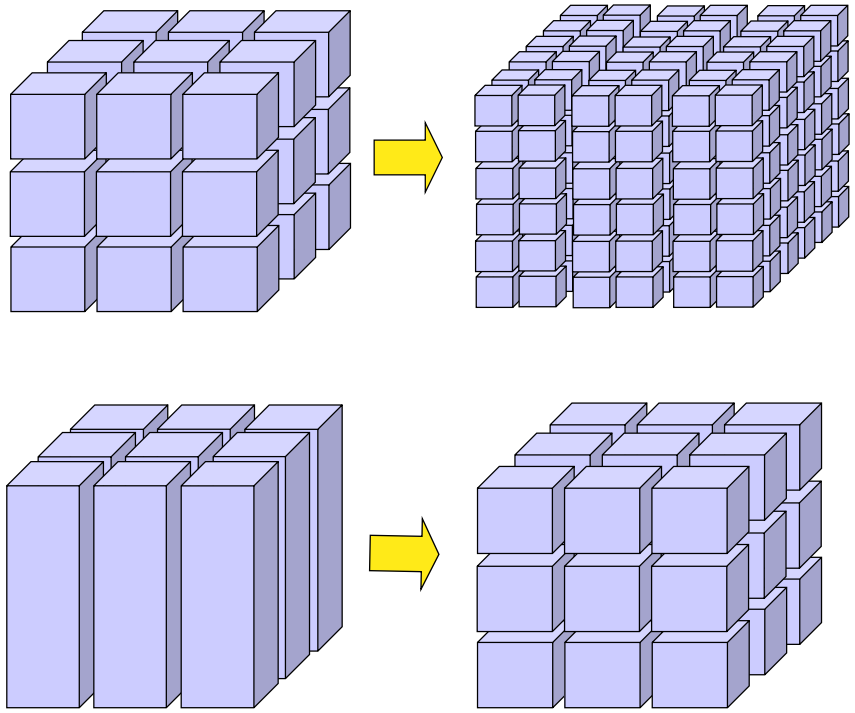
- Data detail increase by
  - increasing detail in a dimension, by walking down a hierarchy
    - example

      group by city, month → group by store, month
  - adding a whole dimension
    - example

      group by product → group by product, city
- Frequently drill down operates on a subset of data produced by the initial query

Elena Baralis
Politecnico di Torino

Drill down



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Drill down

Metrics	Dollar Sales										
Customer Region / Quarter	North-East	Mid-Atlantic	South-East	Central	South	North-West	South-West	England	France	Germany	Canada
Q1 1997	$ 1.526	$ 1.249	$ 978	$ 1.885	$ 3.650	$ 4.855	$ 1.834	$ 2.552	$ 1.920	$ 643	$ 663
Q2 1997	$ 2.979	$ 1.415	$ 2.664	$ 1.582	$ 1.130	$ 1.906	$ 884	$ 1.393	$ 1.402	$ 516	$ 975
Q3 1997	$ 3.016	$ 1.772	$ 5.076	$ 2.050	$ 1.443	$ 2.311	$ 2.321	$ 608	$ 575	$ 782	$ 325
Q4 1997	$ 2.711	$ 5.030	$ 2.025	$ 1.961	$ 3.601	$ 2.112	$ 3.976	$ 918	$ 531	$ 1.676	$ 291
Q1 1998	$ 5.288	$ 3.399	$ 4.935	$ 9.174	$ 3.601	$ 9.484	$ 3.844	$ 2.720	$ 979	$ 1.897	$ 1.204
Q2 1998	$ 1.773	$ 3.658	$ 1.862	$ 1.213	$ 1.115	$ 4.635	$ 352	$ 724	$ 2.110	$ 391	$ 121
Q3 1998	$ 1.818	$ 5.402	$ 1.709	$ 2.485	$ 1.704	$ 3.632	$ 4.329	$ 679	$ 1.198	$ 1.269	$ 809
Q4 1998	$ 2.051	$ 2.968	$ 4.664	$ 5.917	$ 1.775	$ 3.162	$ 3.485	$ 2.750	$ 1.005	$ 846	$ 639

Metrics	Dollar Sales												
Customer City / Quarter	Arlin	San Pedro	Springfield	Chappel Hill	Scranburg	Pebble Beach	Martinsville	Maddon	Peoria	Pecos	Lake Barkley	Alcameda	Fingers Lake
Q1 1997	$ 675										$ 39		
Q2 1997				$ 203					$ 53				$ 135
Q3 1997				$ 276								$ 252	$ 63
Q4 1997	$ 215	$ 124			$ 113	$ 45	$ 192	$ 348				$ 79	$ 98
Q1 1998			$ 140	$ 174			$ 85				$ 237	$ 30	$ 119
Q2 1998								$ 12	$ 17				
Q3 1998	$ 734					$ 25	$ 1.535						
Q4 1998						$ 219	$ 119	$ 142		$ 85	$ 1.533		

From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Drill down

Metrics	Dollar Sales	
Year	1997	1998
Category		
Electronics	$ 10.616	$ 29.299
Food	$ 5.300	$ 5.638
Gifts	$ 16.315	$ 20.047
Health & Beauty	$ 6.042	$ 5.665
Household	$ 38.383	$ 50.391
Kid's Korner	$ 2.559	$ 2.943
Travel	$ 4.497	$ 4.792

Metrics	Dollar Sales											
Customer	North-East		Mid-Atlantic		South-East		Central		South		North-West	
Region												
Year	1997	1998	1997	1998	1997	1998	1997	1998	1997	1998	1997	1998
Category												
Electronics	$ 138	$ 1.184	$ 1.774	$ 4.529	$ 384	$ 1.892	$ 138	$ 7.232	$ 2.346	$ 651	$ 2.554	$ 9.488
Food	$ 759	$ 538	$ 682	$ 925	$ 729	$ 959	$ 262	$ 677	$ 588	$ 213	$ 469	$ 1.503
Gifts	$ 2.532	$ 1.955	$ 1.355	$ 2.785	$ 1.854	$ 2.800	$ 1.413	$ 2.695	$ 2.535	$ 1.813	$ 2.132	$ 2.844
Health & Beauty	$ 624	$ 611	$ 640	$ 887	$ 1.317	$ 566	$ 647	$ 382	$ 588	$ 499	$ 754	$ 1.162
Household	$ 5.354	$ 5.787	$ 4.112	$ 5.320	$ 5.410	$ 5.416	$ 4.446	$ 6.812	$ 3.058	$ 4.334	$ 3.974	$ 5.008
Kid's Korner	$ 201	$ 247	$ 398	$ 422	$ 485	$ 441	$ 186	$ 380	$ 409	$ 221	$ 323	$ 592
Travel	$ 624	$ 608	$ 505	$ 559	$ 564	$ 1.096	$ 386	$ 611	$ 300	$ 464	$ 978	$ 316

From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

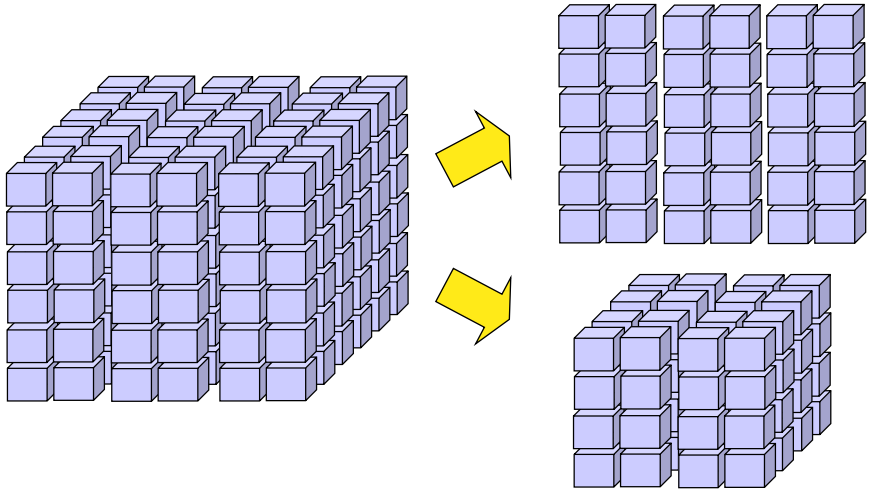
Elena Baralis
Politecnico di Torino

Slice and dice

- Selection of a data subset by means of selection predicates
  - slice: equality predicate selecting a “slice”
    - example: Year=2005
  - dice: predicate expression selecting a “dice”
    - example: Category=‘Food’ and City=‘Torino’

Elena Baralis
Politecnico di Torino

Slice and dice



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Slice and dice

Category	Year	Metrics / Customer Region	Dollar Sales North-East	Mid-Atlantic	South-East	Central	South	North-West	South-West	England	France	Germa
Electronics	1997		$ 138	$ 1.774	$ 384	$ 138	$ 2.346	$ 2.554	$ 2.184	$ 566	$ 199	$
	1998		$ 1.184	$ 4.529	$ 1.892	$ 7.232	$ 651	$ 9.488	$ 476	$ 2.683	$ 462	$ 7
Food	1997		$ 759	$ 682	$ 729	$ 262	$ 588	$ 469	$ 807	$ 156	$ 615	$ 1
	1998		$ 538	$ 925	$ 959	$ 677	$ 213	$ 1.503	$ 261	$ 165	$ 175	$ 1
Gifts	1997		$ 2.532	$ 1.355	$ 1.854	$ 1.413	$ 2.535	$ 2.132	$ 1.904	$ 908	$ 375	$ 1.0
	1998		$ 1.955	$ 2.785	$ 2.800	$ 2.695	$ 1.813	$ 2.844	$ 1.778	$ 1.158	$ 717	$ 6
Health & Beauty	1997		$ 624	$ 640	$ 1.317	$ 647	$ 588	$ 754	$ 654	$ 143	$ 292	$ 3
	1998		$ 611	$ 887	$ 566	$ 382	$ 499	$ 1.162	$ 1.044	$ 273	$ 72	
Household	1997		$ 5.354	$ 4.112	$ 5.410	$ 4.446	$ 3.058	$ 3.974	$ 2.654	$ 3.545	$ 2.875	$ 1.9
	1998		$ 5.787	$ 5.320	$ 5.416	$ 6.812	$ 4.334	$ 5.008	$ 7.588	$ 2.139	$ 3.649	$ 2.7
Kid's Korner	1997		$ 201	$ 398	$ 485	$ 186	$ 409	$ 323	$ 396	$ 105	$ 34	$
	1998		$ 247	$ 422	$ 441	$ 380	$ 221	$ 592	$ 290	$ 198	$ 19	$
Travel	1997		$ 624	$ 505	$ 564	$ 386	$ 300	$ 978	$ 416	$ 48	$ 38	
	1998		$ 608	$ 559	$ 1.096	$ 611	$ 464	$ 316	$ 573	$ 257	$ 198	$

Filter Details:
Year = 1998

Category	Metrics / Customer Region	Dollar Sales North-East	Mid-Atlantic	South-East	Central	South	North-West	South-West	England	France	Germany	Ca
Electronics		$ 1.184	$ 4.529	$ 1.892	$ 7.232	$ 651	$ 9.488	$ 476	$ 2.683	$ 462	$ 702	
Food		$ 538	$ 925	$ 959	$ 677	$ 213	$ 1.503	$ 261	$ 165	$ 175	$ 100	
Gifts		$ 1.955	$ 2.785	$ 2.800	$ 2.695	$ 1.813	$ 2.844	$ 1.778	$ 1.158	$ 717	$ 686	
Health & Beauty		$ 611	$ 887	$ 566	$ 382	$ 499	$ 1.162	$ 1.044	$ 273	$ 72		
Household		$ 5.787	$ 5.320	$ 5.416	$ 6.812	$ 4.334	$ 5.008	$ 7.588	$ 2.139	$ 3.649	$ 2.791	$
Kid's Korner		$ 247	$ 422	$ 441	$ 380	$ 221	$ 592	$ 290	$ 198	$ 19	$ 69	
Travel		$ 608	$ 559	$ 1.096	$ 611	$ 464	$ 316	$ 573	$ 257	$ 198	$ 55	

From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Slice and dice

Metrics / Customer City / Subcategory	Dollar Sales: Afton	Akron	Albon	Alcameda	Alka	Allagash	Alta	Altoola	Amestra	Amsterdam	Andersonville	Annap
Audio						$ 85						
Automotive								$ 30				
Chocolate	$ 42	$ 42		$ 50		$ 20		$ 22	$ 44			$
Christmas	$ 30					$ 25	$ 30	$ 15				
Classic Toys						$ 7	$ 26				$ 38	
Coffee			$ 9									
Comfort				$ 59		$ 59						
Furniture							$ 485					
Gadgets							$ 199	$ 79	$ 79			
Games & Puzzles							$ 17		$ 45		$ 45	
Gift Baskets			$ 55	$ 43								$
Golf	$ 25							$ 25	$ 14		$ 25	
Hearth									$ 15			
Jewelry	$ 75			$ 189		$ 24	$ 77	$ 189	$ 24			
Kitchen						$ 55	$ 21		$ 76			$
Lawn & Garden	$ 75		$ 100		$ 15	$ 63	$ 100		$ 180	$ 67	$ 40	$
Learning	$ 16							$ 37				
Meat & Cheese		$ 40		$ 20			$ 20				$ 25	
Miscellaneous		$ 200	$ 1.320		$ 200	$ 139			$ 993			
Natural Remedies	$ 13								$ 13			
Pets	$ 215		$ 26			$ 30	$ 68	$ 115	$ 25		$ 34	$
Plants & Flowers	$ 65	$ 65	$ 65				$ 50	$ 60				$
Safety & Security								$ 30	$ 22	$ 22		
Skin Care												
Sleeping			$ 18									
Toys & Accessories							$ 29	$ 185	$ 744			$

Filter Details:
Category = Electronics
AND
Dollar Sales > 80
AND
Customer Region = North-West
AND
Year = 1997

Metrics / Customer City / Subcategory	Dollar Sales: Alta	Armstrong	Avery Heights	Lane	Mt. Everest	San Fransisco
Audio		$ 98		$ 123	$ 85	
Comfort			$ 118		$ 1.495	
Gadgets	$ 199					$ 199

From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

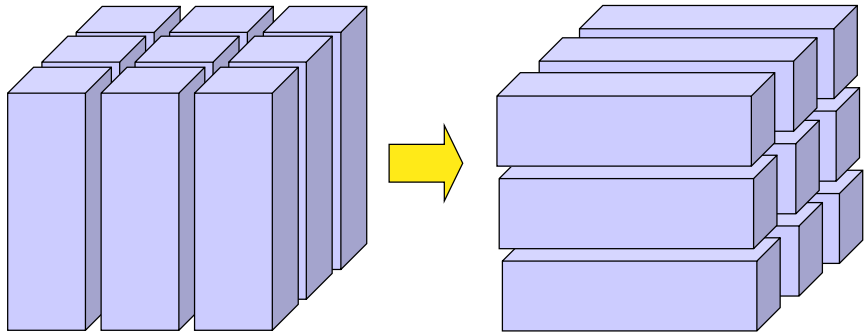
Elena Baralis
Politecnico di Torino

Pivot

- Reorganization of the multidimensional structure without varying the detail level
  - increases readability of the same information
  - multidimensional representation is always based on a "grid" (hierarchical spreadsheet)
    - two dimensions are the main grid axes
    - position of dimensions in the grid are changed

Elena Baralis
Politecnico di Torino

Pivot



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Pivot

Category	Year	Metrics	Dollar Sales
Electronics	1997		$ 10.616
	1998		$ 29.299
Food	1997		$ 5.300
	1998		$ 5.638
Gifts	1997		$ 16.315
	1998		$ 20.047
Health & Beauty	1997		$ 6.042
	1998		$ 5.665
Household	1997		$ 38.383
	1998		$ 50.391
Kid's Korner	1997		$ 2.559
	1998		$ 2.943
Travel	1997		$ 4.497
	1998		$ 4.792

	Metrics	Dollar Sales	
	Year	1997	1998
Category			
Electronics		$ 10.616	$ 29.299
Food		$ 5.300	$ 5.638
Gifts		$ 16.315	$ 20.047
Health & Beauty		$ 6.042	$ 5.665
Household		$ 38.383	$ 50.391
Kid's Korner		$ 2.559	$ 2.943
Travel		$ 4.497	$ 4.792

From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Pivot

Category	Year	Metrics / Customer Region	Dollar Sales North-East	Mid-Atlantic	South-East	Central	South	North-West	South-West	England	France	Germa
Electronics	1997		$ 138	$ 1.774	$ 384	$ 138	$ 2.346	$ 2.554	$ 2.184	$ 566	$ 199	$
	1998		$ 1.184	$ 4.529	$ 1.892	$ 7.232	$ 651	$ 9.488	$ 476	$ 2.683	$ 462	$ 7
Food	1997		$ 759	$ 682	$ 729	$ 262	$ 588	$ 469	$ 807	$ 156	$ 615	$ 1
	1998		$ 538	$ 925	$ 959	$ 677	$ 213	$ 1.503	$ 261	$ 165	$ 175	$ 1
Gifts	1997		$ 2.532	$ 1.355	$ 1.854	$ 1.413	$ 2.535	$ 2.132	$ 1.904	$ 908	$ 375	$ 1.0
	1998		$ 1.955	$ 2.785	$ 2.800	$ 2.695	$ 1.813	$ 2.844	$ 1.778	$ 1.158	$ 717	$ 6
Health & Beauty	1997		$ 624	$ 640	$ 1.317	$ 647	$ 588	$ 754	$ 654	$ 143	$ 292	$ 3
	1998		$ 611	$ 887	$ 566	$ 382	$ 499	$ 1.162	$ 1.044	$ 273	$ 72	
Household	1997		$ 5.354	$ 4.112	$ 5.410	$ 4.446	$ 3.058	$ 3.974	$ 2.654	$ 3.545	$ 2.875	$ 1.9
	1998		$ 5.787	$ 5.320	$ 5.416	$ 6.812	$ 4.334	$ 5.008	$ 7.588	$ 2.139	$ 3.649	$ 2.7
Kid's Korner	1997		$ 201	$ 398	$ 485	$ 186	$ 409	$ 323	$ 396	$ 105	$ 34	$
	1998		$ 247	$ 422	$ 441	$ 380	$ 221	$ 592	$ 290	$ 198	$ 19	$
Travel	1997		$ 624	$ 505	$ 564	$ 386	$ 300	$ 978	$ 416	$ 48	$ 38	
	1998		$ 608	$ 559	$ 1.096	$ 611	$ 464	$ 316	$ 573	$ 257	$ 198	$

Metrics / Customer Region	Dollar Sales North-East		Mid-Atlantic		South-East		Central		South		North-West		S
Year / Category	1997	1998	1997	1998	1997	1998	1997	1998	1997	1998	1997	1998	
Electronics	$ 138	$ 1.184	$ 1.774	$ 4.529	$ 384	$ 1.892	$ 138	$ 7.232	$ 2.346	$ 651	$ 2.554	$ 9.488	$
Food	$ 759	$ 538	$ 682	$ 925	$ 729	$ 959	$ 262	$ 677	$ 588	$ 213	$ 469	$ 1.503	
Gifts	$ 2.532	$ 1.955	$ 1.355	$ 2.785	$ 1.854	$ 2.800	$ 1.413	$ 2.695	$ 2.535	$ 1.813	$ 2.132	$ 2.844	$
Health & Beauty	$ 624	$ 611	$ 640	$ 887	$ 1.317	$ 566	$ 647	$ 382	$ 588	$ 499	$ 754	$ 1.162	
Household	$ 5.354	$ 5.787	$ 4.112	$ 5.320	$ 5.410	$ 5.416	$ 4.446	$ 6.812	$ 3.058	$ 4.334	$ 3.974	$ 5.008	$
Kid's Korner	$ 201	$ 247	$ 398	$ 422	$ 485	$ 441	$ 186	$ 380	$ 409	$ 221	$ 323	$ 592	
Travel	$ 624	$ 608	$ 505	$ 559	$ 564	$ 1.096	$ 386	$ 611	$ 300	$ 464	$ 978	$ 316	

From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Extensions of the SQL language

Elena Baralis
Politecnico di Torino

Extensions of the SQL language

- Interface tools require
  - new aggregate functions
    - aggregate functions exploited for economic analysis (moving average, median, ...)
    - position in the sort order (i.e., rank)
  - functions for report generation
    - partial and cumulative totals
- New OLAP functions in the ANSI standard
  - implemented starting from DB2 UDB 7.1, Oracle 8i v2

Elena Baralis
Politecnico di Torino

Extensions of the SQL language

- Interface tools require
  - operators for the computation of different group bys at the same time
- The SQL-99 (SQL3) standard has extended the SQL group by clause

Elena Baralis
Politecnico di Torino

Example data base

Sales(City,Month,Amount)

City	Month	Amount
Milano	7	110
Milano	8	10
Milano	9	70
Milano	10	90
Milano	11	35
Milano	12	135
Torino	7	70
Torino	8	35
Torino	9	80
Torino	10	95
Torino	11	50
Torino	12	120

Elena Baralis
Politecnico di Torino

SQL OLAP functions

- New class of aggregate functions (OLAP functions) characterized by
  - computation window, inside which the computation of aggregate functions is performed
    - cumulative totals and moving average can be computed
  - new aggregate functions to compute the rank in a given sort order

Elena Baralis
Politecnico di Torino

Computation window

- New **`window`** clause, characterized by
  - *partitioning*: Rows are grouped without collapsing them (different from **`group by`**)
    - no partitioning: a single group is defined
  - *row ordering*, separately in each partition (similar to **`order by`**)
  - *aggregation window*: For each row in the partition, it defines the row group on which the aggregate function is computed

Elena Baralis
Politecnico di Torino

Example

- Show, for each city and month
  - sale amount
  - average on the current month and the two previous months, separately for each city

Elena Baralis
Politecnico di Torino

Example

- Partitioning on city
  - average computation is reset when the city changes
- Ordering by month, to compute the moving average on the current month and the two preceding months
  - without ordering the computation is meaningless
- Aggregation window size: the current row and the two preceding rows

Elena Baralis
Politecnico di Torino

Example

```
SELECT City, Month, Amount,
       AVG(Amount) OVER Wavg AS MovingAvg
FROM Sales
WINDOW Wavg AS (PARTITION BY City
                ORDER BY Month
                ROWS 2 PRECEDING)
```


Elena Baralis
Politecnico di Torino

Example

```
SELECT City, Month, Amount,
       AVG(Amount) OVER (PARTITION BY City
                         ORDER BY Month
                         ROWS 2 PRECEDING)
       AS MovingAvg
FROM Sales
```


Elena Baralis
Politecnico di Torino

Result

City	Month	Amount	MovingAvg
Milano	7	110	110
Milano	8	10	60
Milano	9	90	70
Milano	10	80	60
Milano	11	40	60
Milano	12	140	90
Torino	7	70	70
Torino	8	30	50
Torino	9	80	60
Torino	10	100	70
Torino	11	50	60
Torino	12	150	100

Partition 1

Partition 2

Elena Baralis
Politecnico di Torino

Observations

- Sort order is required, because the computation of the moving average considers rows in an ordered fashion
  - the window sort order does not enforce a predefined output sort order
- When the window is not complete, the computation takes place on the available rows
  - it is possible to require a **`NULL`** result for each incomplete window
- Several different computation windows may be specified

Elena Baralis
Politecnico di Torino

Aggregation window

- The moving window on which the aggregate function is computed may be defined
  - at the *physical level*: It builds the group by counting rows
    - example: the current row and the two preceding rows
  - at the *logical level*: It builds the group by defining an interval on the sort key
    - example: the current month and the two preceding months

Elena Baralis
Politecnico di Torino

Physical interval definition

- Between a lower bound and the current row

  ROWS 2 PRECEDING

- Between lower and upper bounds

  ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING

  ROWS BETWEEN 3 PRECEDING AND 1 PRECEDING

- Between the beginning (or the end) of a partition and the current row

  ROWS UNBOUNDED PRECEDING (o FOLLOWING)

Elena Baralis
Politecnico di Torino

Physical interval

- Appropriate for sequence data with no gaps
  - example: no month is missing in the sequence
  - more than a sort key can be specified
    - computation ignores breaks due to change in any sort key value
    - example: order by month and year
  - no mathematical expressions are needed to compute the window

Elena Baralis
Politecnico di Torino

Logical interval definition

- The **`range`** clause is used, with the same syntax as the physical interval
- A distance on the sort key between the interval bounds and the current value should be defined
- Example

```
RANGE 2 MONTH PRECEDING
```


Elena Baralis
Politecnico di Torino

Logical interval

- Appropriate for “sparse” data, with gaps in the sequence
  - example: a month is missing in the sequence
  - only a single sort key can be specified
  - the sort key can only be alphanumeric or date type (arithmetic expressions are allowed)

Elena Baralis
Politecnico di Torino

Applications

- Moving aggregate computations
  - computations on a window which moves over data
  - examples: moving average, moving sum
- Cumulative total computations
  - the (cumulative) total is incremented by adding an instance at a time
- Comparison between detailed data and aggregated data

Elena Baralis
Politecnico di Torino

Computation of a cumulative total

- Show, for each city and month
  - sale amount
  - cumulative sale amount for increasing months, separately for each city

Elena Baralis
Politecnico di Torino

Computation of a cumulative total

- Partition by city
  - the cumulative total is reset when the city changes
- Order by (ascending) month to compute the sum for increasing months
  - without sorting, the computation would be meaningless
- Size of the aggregation window
  - from the starting row of the partition to the current row

Elena Baralis
Politecnico di Torino

Computation of a cumulative total

```
SELECT City, Month, Amount,
       SUM(Amount) OVER (PARTITION BY City
                         ORDER BY Month
                         ROWS UNBOUNDED PRECEDING)
       AS CumeTot
FROM Sales
```


Elena Baralis
Politecnico di Torino

Computation of a cumulative total

City	Month	Amount	CumeTot
Milano	7	110	110
Milano	8	10	120
Milano	9	90	210
Milano	10	80	290
Milano	11	40	330
Milano	12	140	470
Torino	7	70	70
Torino	8	30	100
Torino	9	80	180
Torino	10	100	280
Torino	11	50	330
Torino	12	150	480

Partition 1

Partition 2

Elena Baralis
Politecnico di Torino

Comparison between detailed data and total data

- Show, for each city and month
  - sale amount
  - total sale amount on the whole time period for the current city

Elena Baralis
Politecnico di Torino

Comparison between detailed data and total data

- Partition by city
  - the total amount is reset when the city changes
- Sorting is not needed
  - the total amount is computed independently of the sort order of tuples
- The aggregation window is not needed
  - it is the whole partition

Elena Baralis
Politecnico di Torino

Comparison between detailed data and total data

```
SELECT City, Month, Amount,
       SUM(Amount) OVER (PARTITION BY City)
       AS TotalAmount
FROM Sales
```


Elena Baralis
Politecnico di Torino

Comparison between detailed data and total data

City	Month	Amount	TotalAmount
Milano	7	110	470
Milano	8	10	470
Milano	9	90	470
Milano	10	80	470
Milano	11	40	470
Milano	12	140	470
Torino	7	70	480
Torino	8	30	480
Torino	9	80	480
Torino	10	100	480
Torino	11	50	480
Torino	12	150	480

Partition 1

Partition 2

Elena Baralis
Politecnico di Torino

Comparison between detailed data and total data

- Show, for each city and month
  - sale amount
  - ratio between current row amount and grand total
  - ratio between current row amount and total amount by city
  - ratio between current row amount and total amount by month

Elena Baralis
Politecnico di Torino

Comparison between detailed data and total data

- Three different computation windows
  - grand total: no partitioning
  - total by city: partition by city
  - total by month: partition by month
- No sort is needed in any window
  - totals are independent of the sort order of tuples
- The aggregation window is always the whole partition

Elena Baralis
Politecnico di Torino

Comparison between detailed data and total data

```
SELECT City, Month, Amount
  Amount/SUM(Amount) OVER ()
  AS TotalFract
  Amount/SUM(Amount) OVER (PARTITION BY City)
  AS CityFract
  Amount/SUM(Amount) OVER (PARTITION BY Month)
  AS MonthFract
FROM Sales
```


Elena Baralis
Politecnico di Torino

Comparison between detailed data and total data

City	Month	Amount	TotalFract	CityFract	MonthFrct
Milano	7	110	110/950	110/470	110/180
Milano	8	10	10/950	10/470	10/40
Milano	9	90	90/950	90/470	90/170
Milano	10	80	80/950	80/470	80/180
Milano	11	40	40/950	40/470	40/90
Milano	12	140	140/950	140/470	140/290
Torino	7	70	70/950	70/480	70/180
Torino	8	30	30/950	30/480	30/40
Torino	9	80	80/950	80/480	80/170
Torino	10	100	100/950	100/480	100/180
Torino	11	50	50/950	50/480	50/90
Torino	12	150	150/950	150/480	150/290

Elena Baralis
Politecnico di Torino

Group by and window

- Windows can be used together with grouping performed by **group by**
- The "temporary table" generated by the execution of the **group by** clause (possibly with aggregate function computation) becomes the operand to which the computations in the **window** clause are applied

Elena Baralis
Politecnico di Torino

Example

- Assume that the **`Sales`** table contains information on sales with daily granularity
- Show, for each city and month
  - sale amount
  - average sale with respect to the current month and the two preceding months, separately for each city

Elena Baralis
Politecnico di Torino

Example

- Grouping by month is needed to compute the total amount by month before computing the moving average
  - the group by clause is used for computing the monthly total
- The temporary table generated by the group by computation is the operand on which the computation window is defined

Elena Baralis
Politecnico di Torino

Example

```
SELECT City, Month, SUM(Amount) AS TotMonth,
       AVG(SUM(Amount)) OVER (PARTITION BY City
                              ORDER BY Month
                              ROWS 2 PRECEDING)
       AS MovingAvg
FROM Sales
WHERE <join conditions>
GROUP BY City, Month
```


Elena Baralis
Politecnico di Torino

Ranking functions

- Functions computing the rank of a value inside a partition
  - **`rank()`** function: computes the rank by leaving an empty slot after a tie
    - example: after 2 first, the next rank is third
  - **`denserank()`** function: computes the rank by leaving an empty slot after a tie
    - example: after 2 first, the next rank is second

Elena Baralis
Politecnico di Torino

Example

- Show, for each city in december
  - sale amount
  - rank on amount

Elena Baralis
Politecnico di Torino

Example

- Partitioning is not needed
  - a single partition including all cities
- Order by amount to perform ranking
  - without sorting, the computation would be meaningless
- The aggregation window is the whole partition

Elena Baralis
Politecnico di Torino

Example

```
SELECT City, Amount,
       RANK() OVER (ORDER BY Amount DESC)
       AS Ranking
FROM Sales
WHERE Month = 12
```


Elena Baralis
Politecnico di Torino

Result

City	Amount	Ranking
Torino	150	1
Milano	140	2

Elena Baralis
Politecnico di Torino

Sorting the result

- A sorted result is obtained by means of the **`order by`** clause
  - may be different from the sort order in the computation window
- Example: sort the result in the former example by increasing city

Elena Baralis
Politecnico di Torino

Example

```
SELECT City, Amount,
       RANK() OVER (ORDER BY Amount DESC)
       AS Ranking
FROM Sales
WHERE Month = 12
ORDER BY City
```

City	Amount	Ranking
Milano	140	2
Torino	150	1

Elena Baralis
Politecnico di Torino

`group by` clause extensions

- Multidimensional spreadsheets compute several partial totals “in one shot”
  - total sale amount by month and city
  - total sale amount by month
  - total sale amount by city
- For the sake of efficiency avoid
  - multiple data reads
  - redundant data sorts

Elena Baralis
Politecnico di Torino

`group by` clause extensions

- SQL-99 standard extended the syntax of the **`group by`** clause
  - **`rollup`** computes aggregations on all groups obtained by removing one by one the columns in the grouping clause
  - **`cube`** computes aggregations on all combinations of the columns in the grouping clause
  - **`grouping sets`** computes aggregations on the group list in the grouping clause (grouping sets different from the previous clauses may be specified)
    - `()` for grand totals (no grouping)

Elena Baralis
Politecnico di Torino

Rollup: example

- Consider the following tables

  Time(<u>Tkey</u>,Day,Month,Year,…)

  Shop(<u>Skey</u>,City,Region,…)

  Product(<u>Pkey</u>,PName,Brand,…)

  Sales(<u>Skey</u>,<u>Tkey</u>,<u>Pkey</u>,Amount)

- Compute total sales in the year 2000 for the following attribute combinations
  - product, month, city
  - month, city
  - city

Elena Baralis
Politecnico di Torino

Rollup: example

```
SELECT City, Month, Pkey,
       SUM(Amount) AS TotSales
FROM Time T, Shop S, Sales V
WHERE T.Tkey = V.Tkey
  AND S.Skey = V.Skey
  AND Year = 2000
GROUP BY ROLLUP (City,Month,Pkey)
```

- The column sort order in **`rollup`** determines which aggregates are computed

Elena Baralis
Politecnico di Torino

Rollup: result

City	Month	Pkey	TotSales
Milano	7	145	110
Milano	7	150	10
Milano	…	…	…
Milano	7	**NULL**	8500
Milano	8	…	…
Milano	**NULL**	**NULL**	150000
Torino	…	…	150
Torino	…	**NULL**	2500
Torino	**NULL**	**NULL**	135000
…	…	…	…
NULL	**NULL**	**NULL**	25005000

- "Superaggregates" are represented by **NULL**

Elena Baralis
Politecnico di Torino

Cube: example

- Compute total sales in the year 2000 for *all* combinations of the following attributes
  - product, month, city
- The following aggregations should be computed
  - product, month, city
  - product, month
  - month, city
  - product, city
  - product
  - month
  - city
  - no grouping

Elena Baralis
Politecnico di Torino

Cube: example

```
SELECT City, Month, Pkey,
       SUM(Amount) AS TotSales
FROM Time T, Shop S, Sales V
WHERE T.Tkey = V.Tkey
  AND S.Skey = V.Skey
  AND Year = 2000
GROUP BY CUBE (City,Month,Pkey)
```

- The sort order of columns in **cube** is irrelevant

Elena Baralis
Politecnico di Torino

Cube computation

- Consider distributive and algebraic properties of aggregate functions
  - *distributive* aggregate functions (**`min`**, **`max`**, **`sum`**, **`count`**) may be computed from aggregations on a larger set of attributes (i.e., with larger granularity)
    - Example: from total sales by product and month, total sales by month may be computed
  - algebraic aggregate functions (**`avg`**, …) may be computed from aggregations on a larger set of attributes (i.e., with larger granularity), if appropriate support aggregations are stored
    - Example: average requires
      - the average value in the group
      - the cardinality of the group

Elena Baralis
Politecnico di Torino

Cube computation

- To increase the efficiency of cube computation, the distributive/algebraic properties of the aggregate functions are exploited
  - previously computed **`group by`** are exploited
  - **`rollup`** requires a single sort operation
  - the cube is a combination of several **`rollup`** operations (in the appropriate order)
  - previously executed sort operations are exploited (also partially)
    - it is possible to exploit sort on (A,B) to sort by (A,C)

Elena Baralis
Politecnico di Torino

Grouping Set: example

- Compute total sales in the year 2000 for the following groups
  - month
  - month, city, product
- A roll up would perform the computation of unnecessary groupings and aggregations

Elena Baralis
Politecnico di Torino

Grouping Set: example

```
SELECT City, Month, Pkey,
       SUM(Amount) AS TotSales
FROM Time T, Shop S, Sales S
WHERE T.Tkey = S.Tkey
  AND S.Skey = S.Skey
  AND Year = 2000
GROUP BY GROUPING SETS
      (Month, (City,Month,Pkey))
```


Elena Baralis
Politecnico di Torino

ETL Process

Elena Baralis
Politecnico di Torino

Extraction, Transformation and Loading (ETL)

- Prepares data to be loaded into the data warehouse
  - data extraction from (OLTP and external) sources
  - data cleaning
  - data transformation
  - data loading
- Eased by exploiting the staging area
- Performed
  - when the DW is first loaded
  - during periodical DW refresh

Elena Baralis
Politecnico di Torino

Extraction

- Data acquisition from sources
- Extraction methods
  - static: snapshot of operational data
    - performed during the first DW population
  - incremental: selection of updates that took place after last extraction
    - exploited for periodical DW refresh
    - immediate or deferred
- The selection of which data to extract is based on their quality

Elena Baralis
Politecnico di Torino

Extraction

- It depends on how operational data is collected
  - historical: all modifications are stored for a given time in the OLTP system
    - bank transactions, insurance data
    - operationally simple
  - partly historical: only a limited number of states is stored in the OLTP system
    - operationally complex
  - transient: the OLTP system only keeps the *current* data state
    - example: stock inventory
    - operationally complex

Elena Baralis
Politecnico di Torino

Incremental extraction

- Application assisted
  - data modifications are captured by ad hoc application functions
  - requires changing OLTP applications (or APIs for database access)
  - increases application load
  - hardly avoidable in legacy systems
- Log based
  - log data is accessed by means of appropriate APIs
  - log data format is usually proprietary
  - efficient, no interference with application load

Elena Baralis
Politecnico di Torino

Incremental extraction

- Trigger based
  - triggers capture interesting data modifications
  - does not require changing OLTP applications
  - increases application load
- Timestamp based
  - modified records are marked by the (last) modification timestamp
  - requires modifying the OLTP database schema (and applications)
  - deferred extraction, may lose intermediate states if data is transient

Elena Baralis
Politecnico di Torino

Comparison of extraction techniques

	Static	*Timestamps*	*Appilcation assisted*	*Trigger*	*Log*
Management of transient or semi-periodic data	No	Incomplete	Complete	Complete	Complete
Support to file-based systems	Yes	Yes	Yes	No	Rare
Implementation technique	Tools	Tools or internal developments	Internal developments	Tools	Tools
Costs of enterprise specific development	None	Medium	High	None	None
Use with legacy systems	Yes	Difficult	Difficult	Difficult	Yes
Changes to applications	None	Likely	Likely	None	None
DBMS-dependent procedures	Limited	Limited	Variabile	High	Limited
Impact on operational system performance	None	None	Medium	Medium	None
Complexity of extraction procedures	Low	Low	High	Medium	Low

From Devlin, Data warehouse: from architecture to implementation, Addisono-Wesley, 1997

Elena Baralis
Politecnico di Torino

Incremental extraction

4/4/2010

Cod	Product	Customer	Qty
1	Greco di tufo	Malavasi	50
2	Barolo	Maio	150
3	Barbera	Lumini	75
4	Sangiovese	Cappelli	45

6/4/2010

Cod	Product	Customer	Qty
1	Greco di tufo	Malavasi	50
2	Barolo	Maio	150
4	Sangiovese	Cappelli	145
5	Vermentino	Maltoni	25
6	Trebbiano	Maltoni	150

Incremental difference

Cod	Product	Customer	Qty	Action
3	Barbera	Lumini	75	D
4	Sangiovese	Cappelli	145	U
5	Vermentino	Maltoni	25	I
6	Trebbiano	Maltoni	150	I

From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Data cleaning

- Techniques for improving data quality (correctness and consistency)
  - duplicate data
  - missing data
  - unexpected use of a field
  - impossible or wrong data values
  - inconsistency between logically connected data
- Problems due to
  - data entry errors
  - different field formats
  - evolving business practices

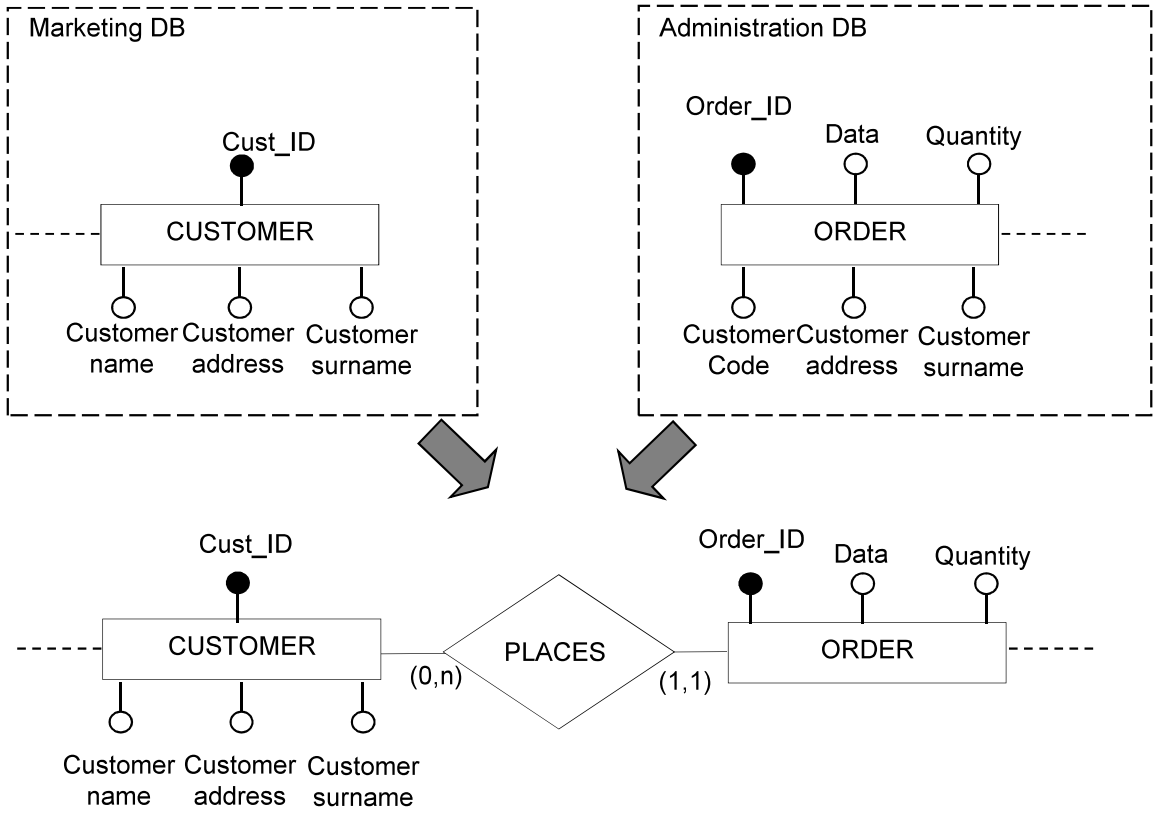
Elena Baralis
Politecnico di Torino

Data cleaning

- Each problem is solved by an ad hoc technique
  - data dictionary
    - appropriate for data entry errors or format errors
    - can be exploited only for data domains with limited cardinality
  - approximate fusion
    - appropriate for detecting duplicates/similar data correlations
      - approximate join
      - purge/merge problem
  - outlier identification, deviations from business rules
- Prevention is the best strategy
  - reliable and rigorous OLTP data entry procedures

Elena Baralis
Politecnico di Torino

Approximate join

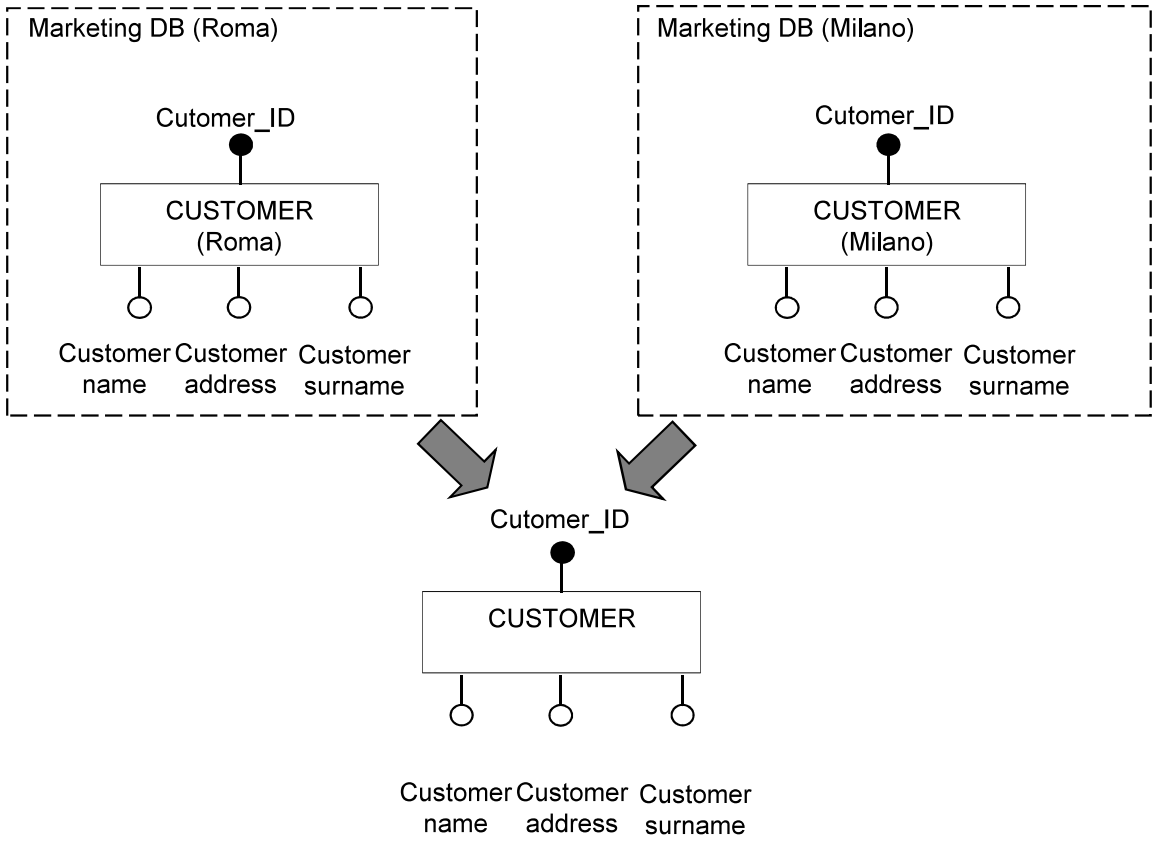


- The join operation should be executed based on common fields, not representing the customer identifier

From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Purge/Merge problem



- Duplicate tuples should be identified and removed
- A criterion is needed to evaluate record similarity

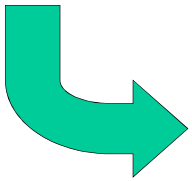
From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

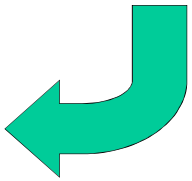
Data cleaning and transformation example

Elena Baralis
C.so Duca degli Abruzzi 24
20129 Torino (I)

Normalization



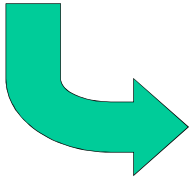
name:	Elena
surname:	Baralis
address:	C.so Duca degli Abruzzi 24
ZIP:	20129
city:	Torino
country:	I



Standardization

name:	Elena
surname:	Baralis
address:	Corso Duca degli Abruzzi 24
ZIP:	20129
city:	Torino
country:	Italia

Correction



name:	Elena
surname:	Baralis
address:	Corso Duca degli Abruzzi 24
ZIP:	10129
city:	Torino
country:	Italia

Adapted from Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Transformation

- Data conversion from operational format to data warehouse format
  - requires data integration
- A uniform operational data representation (reconciled schema) is needed
- Two steps
  - from operational sources to reconciled data in the staging area
    - conversion and normalization
    - matching
    - (possibly) significant data selection
  - from reconciled data to the data warehouse
    - surrogate keys generation
    - aggregation computation

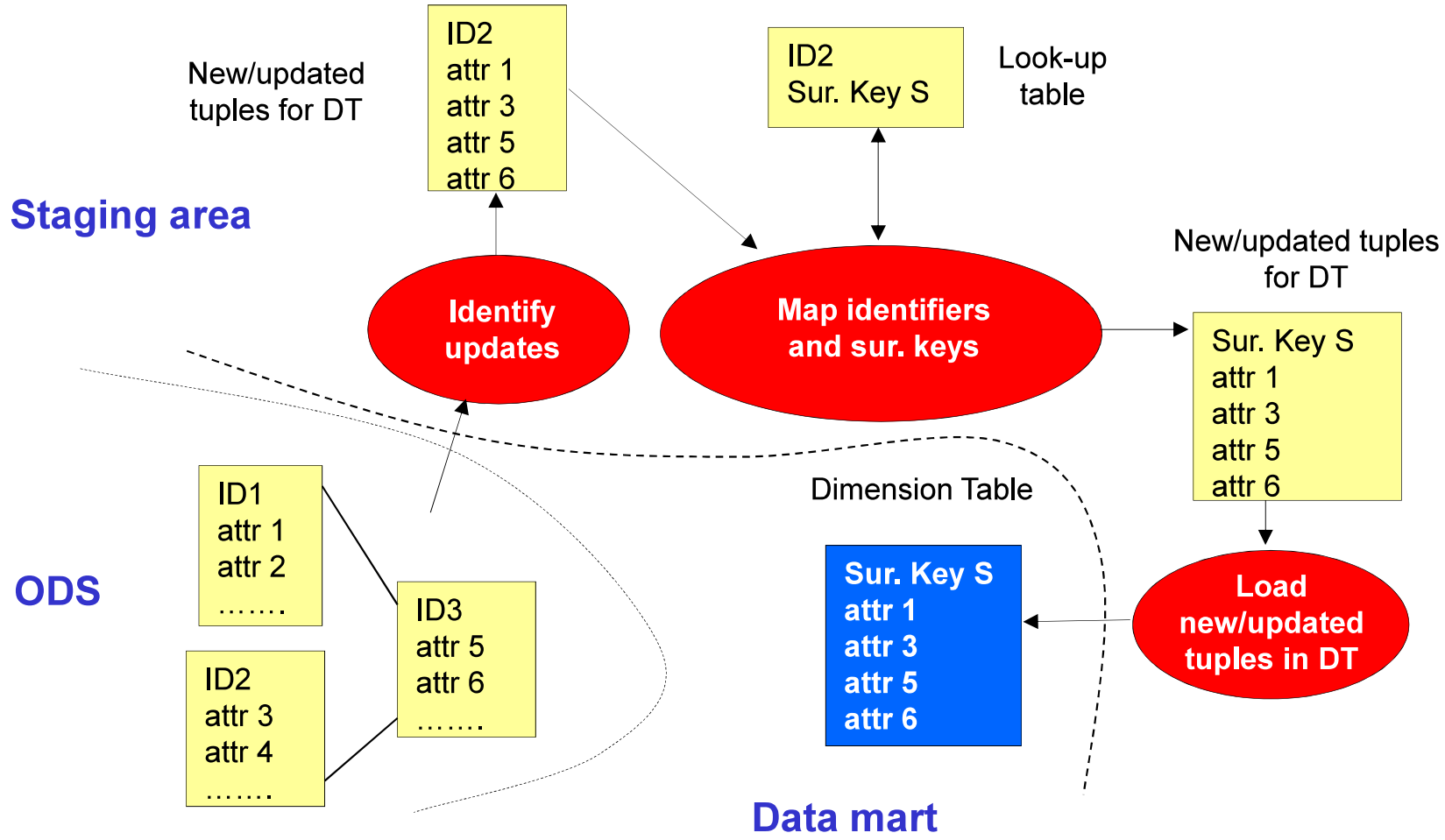
Elena Baralis
Politecnico di Torino

Data warehouse loading

- Update propagation to the data warehouse
- Update order that preserves data integrity
  1. dimensions
  2. fact tables
  3. materialized views and indices
- Limited time window to perform updates
- Transactional properties are needed
  - reliability
  - atomicity

Elena Baralis
Politecnico di Torino

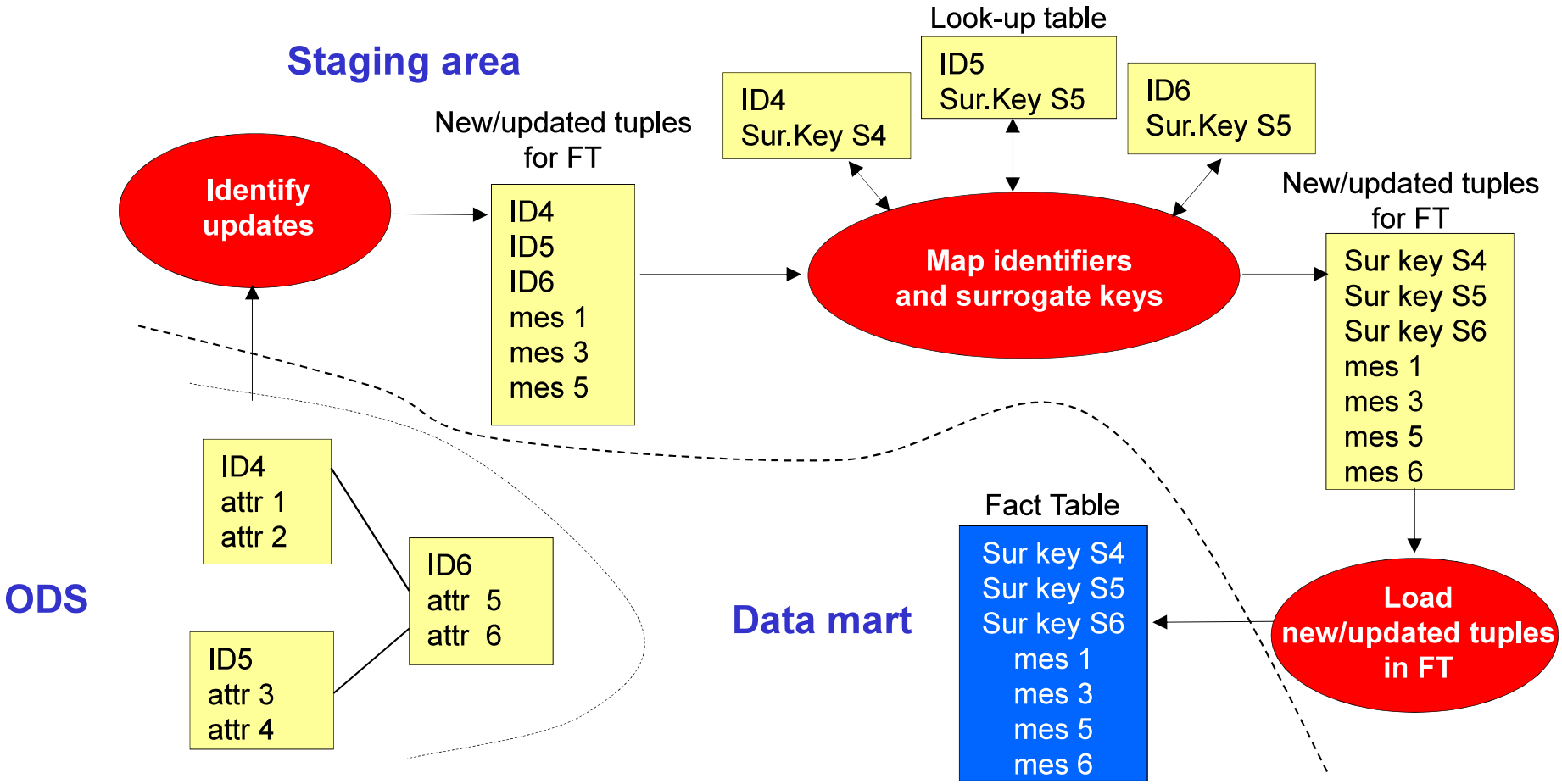
Dimension table loading



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

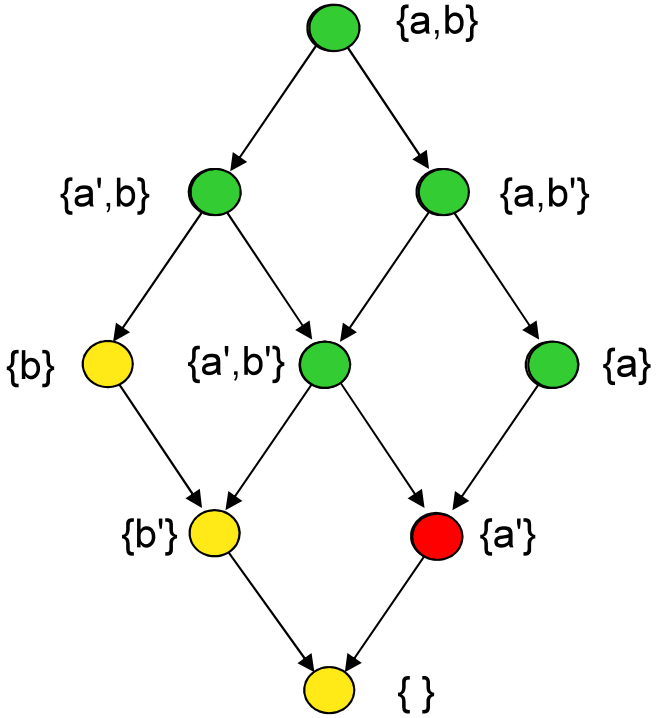
Fact table loading



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Materialized view loading



Tratto da Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

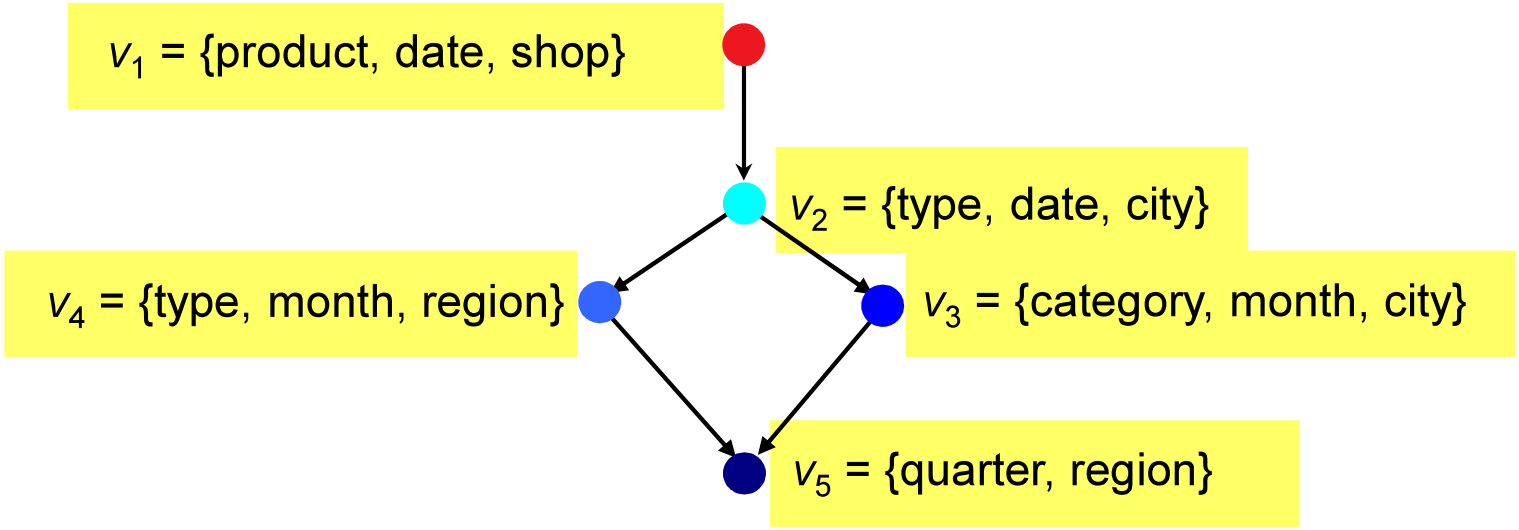
Elena Baralis
Politecnico di Torino

Materialized views

Elena Baralis
Politecnico di Torino

Materialized views

- Precomputed summaries for the fact table
  - explicitly stored in the data warehouse
  - provide a performance increase for aggregate queries



From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

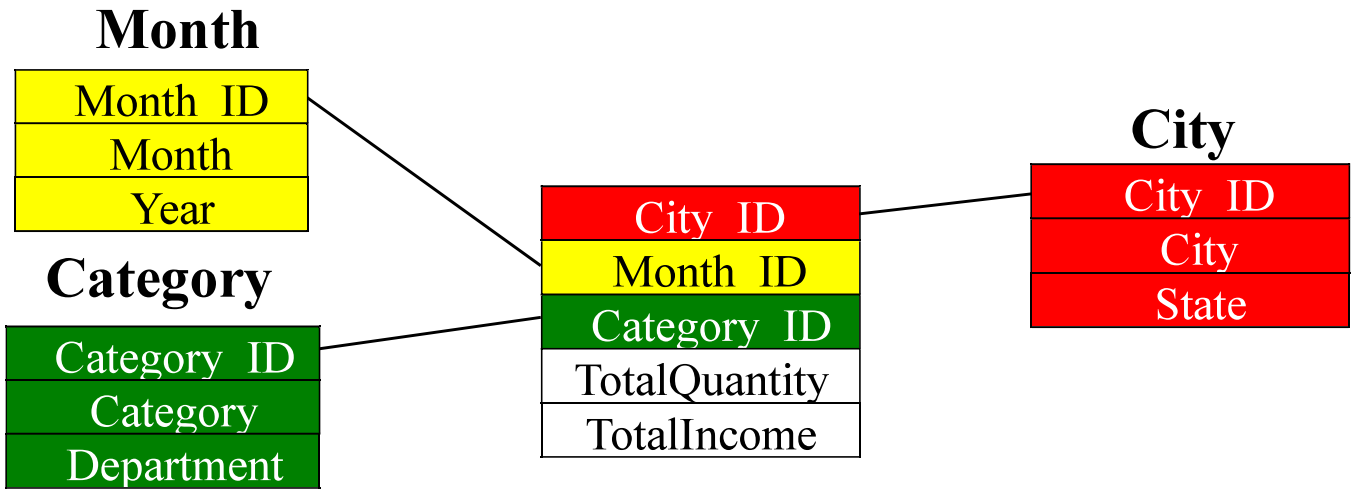
Elena Baralis
Politecnico di Torino

Materialized views

- Defined by SQL statements
- Example: definition of v_3
  - Starting from base tables or views with higher granularity

```
group by City, Category, Month
```

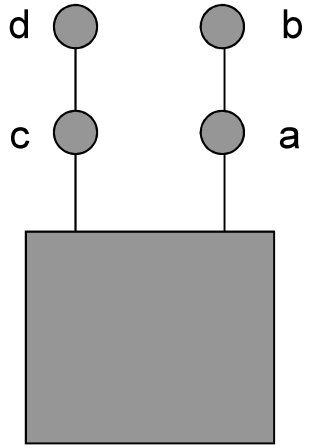
  - Aggregation (SUM) on `Quantity`, `Income` measures
  - Reduction of detail in dimensions

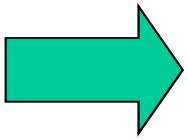


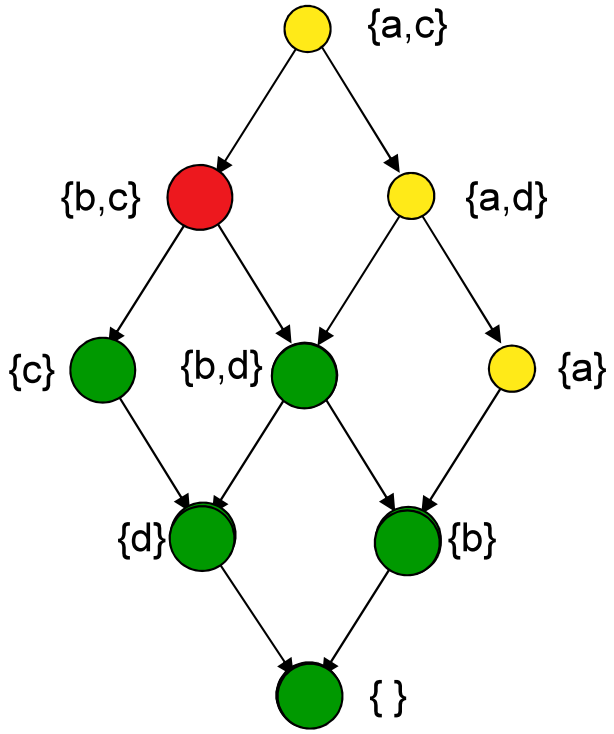
Elena Baralis
Politecnico di Torino

Materialized views

- Materialized views may be exploited for answering several different queries
  - not for all aggregation operators







Multidimensional lattice

From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

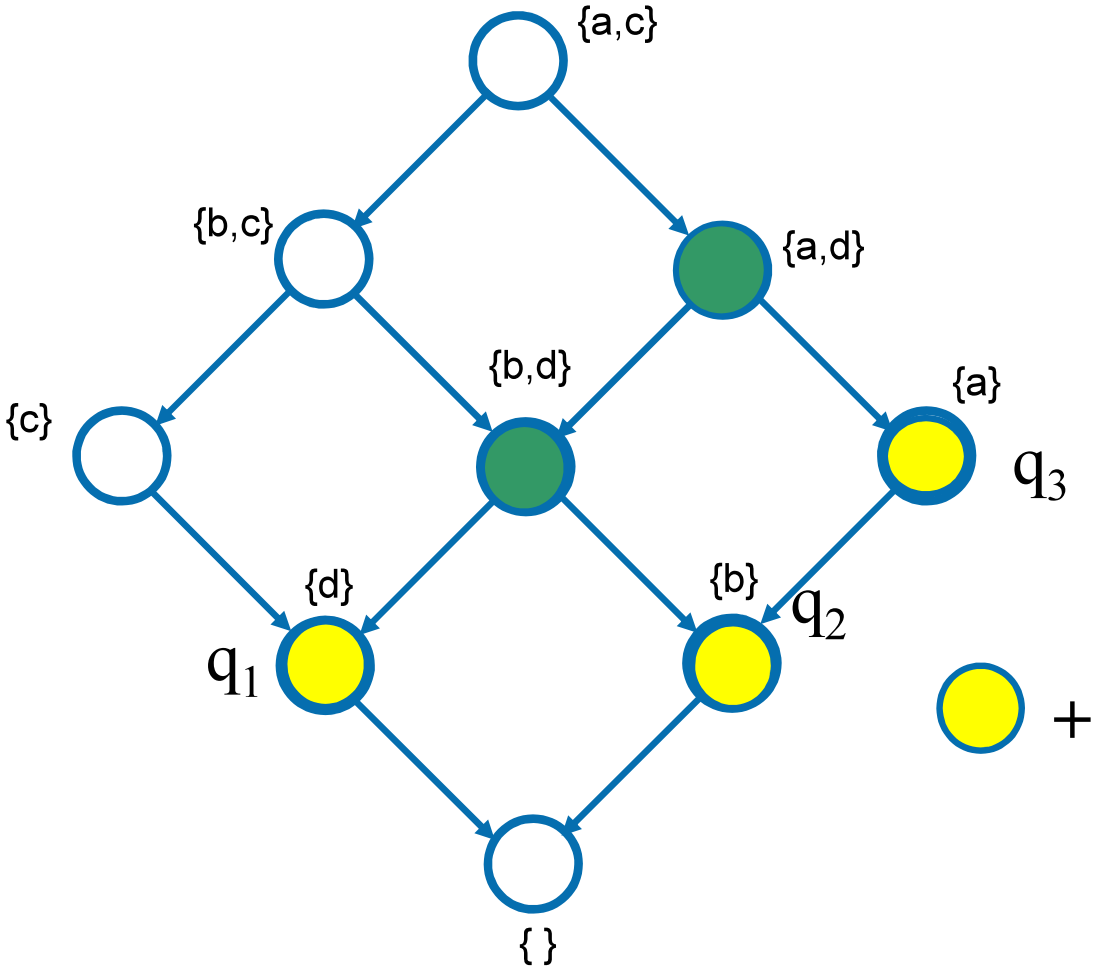
Elena Baralis
Politecnico di Torino

Materialized view selection

- Huge number of allowed aggregations
  - most attribute combinations are eligible
- Selection of the “best” materialized view set
- Cost function minimization
  - query execution cost
  - view maintainance (update) cost
- Constraints
  - available space
  - time window for update
  - response time
  - data freshness

Elena Baralis
Politecnico di Torino

Materialized view selection



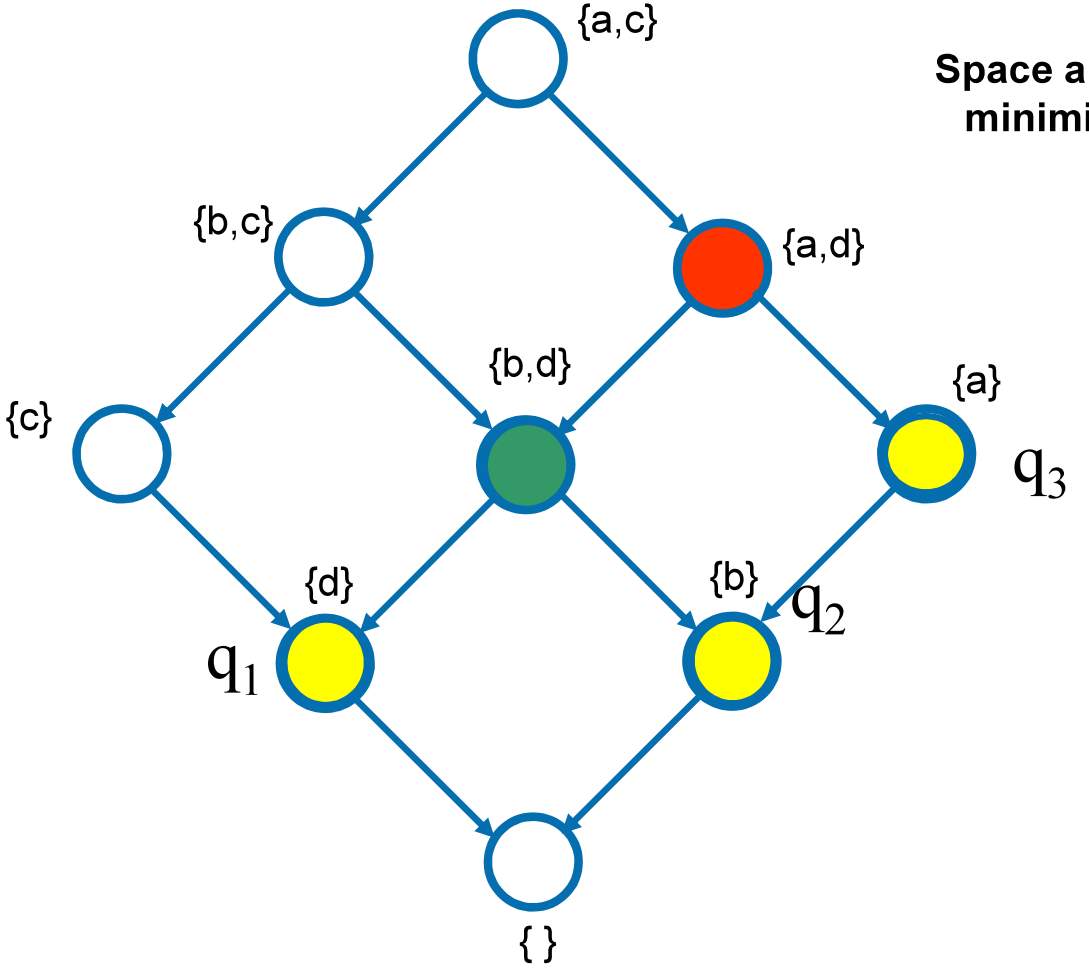
Multidimensional lattice

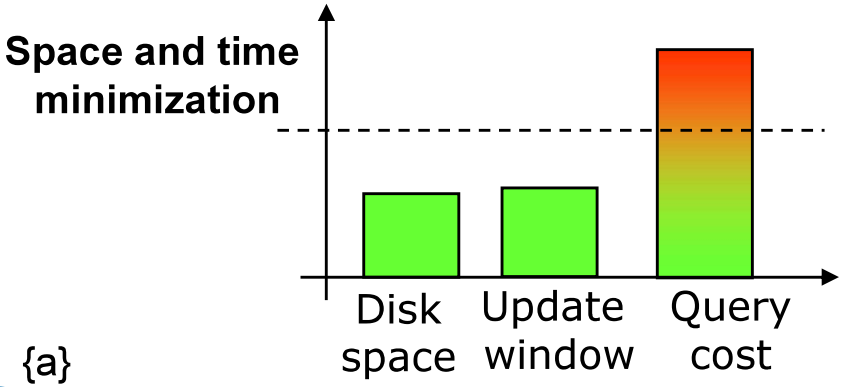
= *candidate views*, **possibly useful to increase workload query performance**

From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Materialized view selection

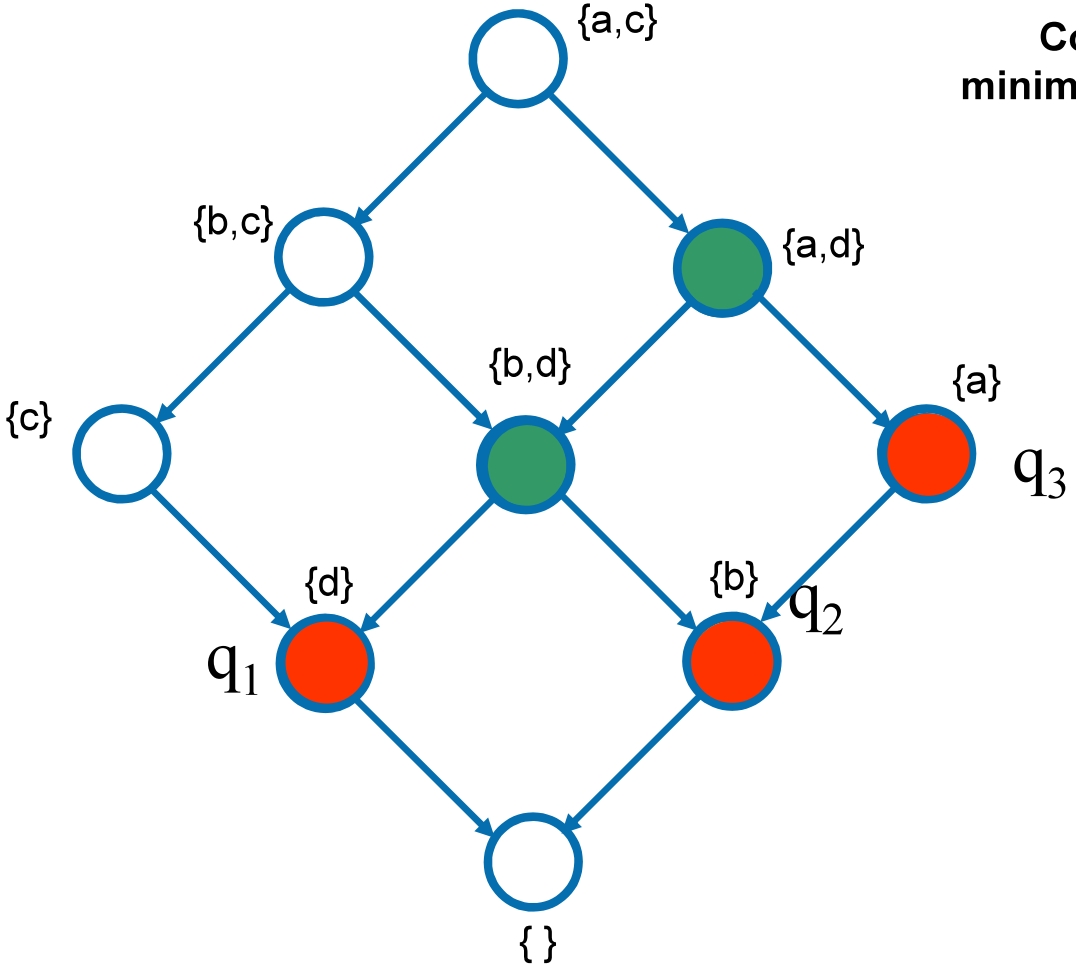


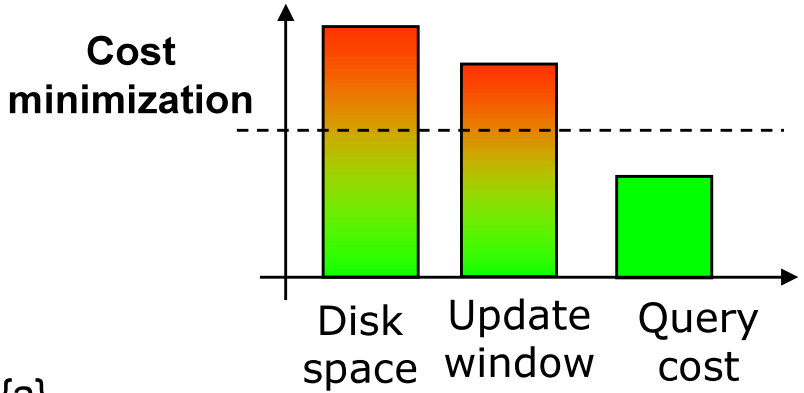


From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Materialized view selection

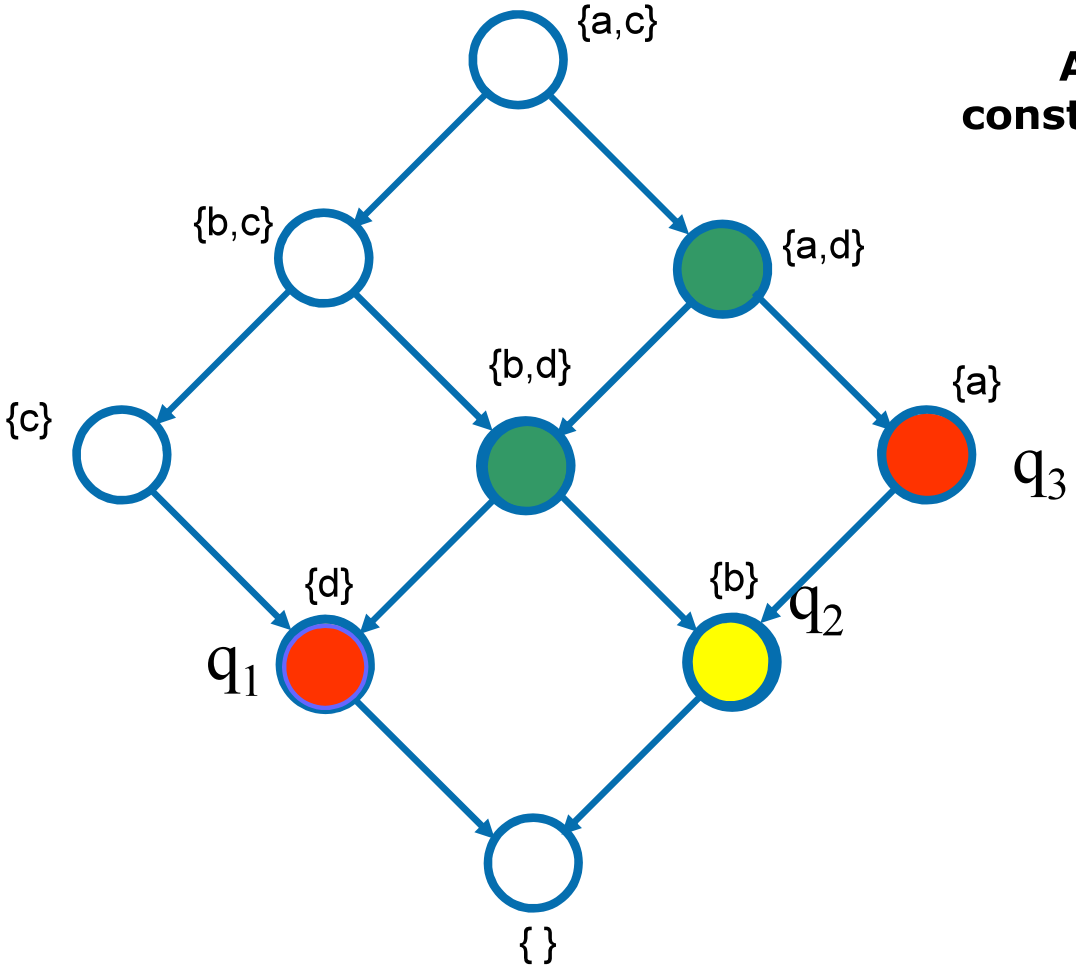


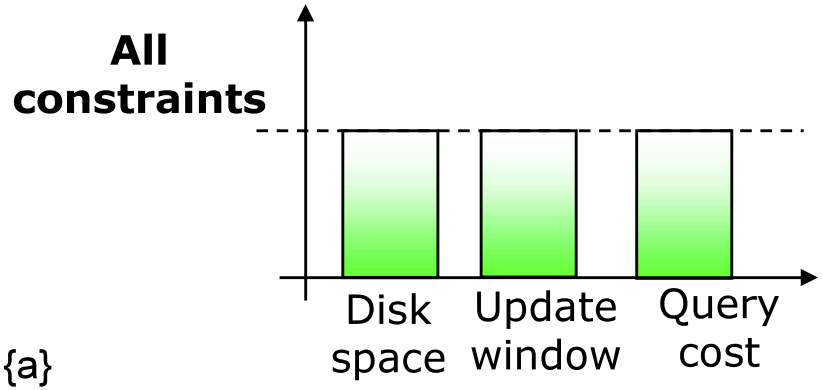


From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Materialized view selection





From Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Data warehousing in Oracle

Materialized views and SQL extensions to analyze data in Oracle data warehouses

Data Base and Data Mining Group of Politecnico di Torino

SQL extensions for data warehouse analysis

Data Base and Data Mining Group of Politecnico di Torino

Available OLAP functions

- Computation windows
  - window
- Ranking functions
  - rank, dense rank, ...
- Group by clause extensions
  - rollup, cube, ...

Physical aggregation example

- Example table
  - SALES(**City**, **Date**, Amount)
- Select, separately for each city and for each date, the amount and the average amount over the current and the previous two rows

Physical aggregation example

```
SELECT Date, Amount,
   AVG(Amount) OVER (
        PARTITION BY City
        ORDER BY Date
        ROWS 2 PRECEDING
          ) AS MovingAverage
FROM Sales
ORDER BY Date;
```

Logical aggregation example

- Example table
  - SALES(**City**, **Date**, Amount)
- Select, separately for each city and for each date, the amount and the average amount over the current row and the sales of the two previous days

Logical aggregation example

```
SELECT Date, Amount,
   AVG(Amount) OVER (
        PARTITION BY City
        ORDER BY Date
        RANGE BETWEEN INTERVAL '2'
        DAY PRECEDING AND CURRENT ROW
        )  AS Last3DaysAverage
FROM Sales
ORDER BY Date;
```

Example tables

- Schema
  - SUPPLIERS(**Cod_S**, Name, SLocation)
  - ITEM(**Cod_I**, Type, Color, Weight)
  - PROJECTS(**Cod_P**, Name, PLocation)
  - FACTS(**Cod_S**, **Cod_I**, **Cod_P**, SoldAmount)

Ranking example

- Select for each item the total amount sold and the ranking according to the total amount sold

Ranking example

```
SELECT COD_I, SUM(SoldAmount),
RANK() OVER (
        ORDER BY SUM(SoldAmount)
        )  AS SalesRank
FROM Facts
GROUP BY COD_I;
```

Ranking example

COD_I	SUM(SoldAmount)	DenseSalesRank
I2	300	1
I5	1100	2
I4	1300	3
I6	1300	3
I1	1900	5
I3	4500	6

Dense ranking

```
SELECT COD_I, SUM(SoldAmount),
DENSE_RANK() OVER (
        ORDER BY SUM(SoldAmount)
        ) AS DenseSalesRank
FROM Facts
GROUP BY COD_I;
```

Ranking example

COD_I	SUM(SoldAmount)	DenseSalesRank
I2	300	1
I5	1100	2
I4	1300	3
I6	1300	3
I1	1900	4
I3	4500	5

Double ranking

- Select for each item the code, the weight, the total amount sold, the ranking according to the weight and the ranking according to the total amount sold

Double ranking

```
SELECT Item.COD_I, Item.Weight,
   RANK() OVER (ORDER BY Item.Weight
                         ) AS WeightRank
   RANK() OVER (ORDER BY SUM(SoldAmount)
                         ) AS SalesRank
FROM Facts, Item
WHERE Facts.COD_I = Item.COD_I
GROUP BY Item.COD_I, Item.Weight
ORDER BY WeightRank;
```

Double ranking

COD_I	Weigh	SUM(SoldAmount)	WeightRank	SalesRank
I1	12	1900	1	5
I5	12	1100	1	2
I4	14	1300	3	3
I2	17	300	4	1
I3	17	4500	4	6
I6	19	1300	6	3

Top N ranking selection

- Select
  - the **top two** most sold items
  - their code
  - their weight
  - the total amount sold
  - and their ranking according to the total amount sold

Top N ranking selection

- Returning only the top two items can be performed by nesting the ranking query inside an outer query
- The outer query uses the nested ranking query as a table (after the FROM clause)
- The outer query selects the requested values of the rank field

Top N ranking selection

```
SELECT * FROM
   (SELECT COD_I, SUM(SoldAmount),
     RANK() OVER (ORDER BY SUM(SoldAmount))
                          AS SalesRank
    FROM Facts
    GROUP BY COD_I)
WHERE SalesRank<=2;
```

SUPPLIERS(**Cod_S**, Name, SLocation)
ITEM(**Cod_I**, Type, Color, Weight)
PROJECTS(**Cod_P**, Name, PLocation)
FACTS(**Cod_S**, **Cod_I**, **Cod_P**, SoldAmount)

Top N ranking selection

```
SELECT * FROM
    (SELECT COD_I, SUM(SoldAmount),
      RANK() OVER (ORDER BY SUM(SoldAmount))
                          AS SalesRank
     FROM Facts
     GROUP BY COD_I)
WHERE SalesRank<=2;
```

Temporary table created at runtime
and dropped at the end of the outer query

ROW_NUMBER

- ROW_NUMBER
  - in each partition it assigns a progressive number to each row

- Partition the items according to their type and enumerate in progressive order the data in each partition. In each partition the rows are sorted according to the weight

ROW_NUMBER

```
SELECT Type, Weight,
    ROW_NUMBER() OVER (
        PARTITION BY Type
        ORDER BY Weight
        )  AS RowNumberWeight
FROM Item;
```

ROW_NUMBER

Type	Weight	RowNumberWeight	
Bar	12	1	Partition 1
Gear	19	1	Partition 2
Screw	12	1	Partition 3
Screw	14	2	
Screw	16	3	
Screw	16	4	
Screw	16	5	
Screw	16	6	
Screw	17	7	
Screw	17	8	
Screw	18	9	
Screw	20	10	

CUME_DIST

- CUME_DIST
  - in each partition it assigns a weight between 0 and 1 to each row according to the number of values which precede the value of the attribute employed for the sorting in the partition
- Given a partition with N rows, for each row x the CUME_DIST is computed as follows:
  - CUME_DIST(x) = number of values, which precede or have the same value of the attribute employed for the sorting, divided by N

CUME_DIST example

- Partition the items according to the type and sort in each partition according to the weight of items. Assign to each row the corresponding value of CUME_DIST

CUME_DIST example

```
SELECT Type, Weight,
   CUME_DIST() OVER (
      PARTITION BY Type
      ORDER BY Weight
      )  AS CumeWeight
FROM Item;
```

Example CUME_DIST

Type	Weight	RowNumberWeight		
Bar	12	1	(=1/1)	Partition 1
Gear	19	1	(=1/1)	Partition 2
Screw	12	0.1	(=1/10)	Partition 3
Screw	14	0.2	(=2/10)	
Screw	16	0.6	(=6/10)	
Screw	16	0.6	(=6/10)	
Screw	16	0.6	(=6/10)	
Screw	16	0.6	(=6/10)	
Screw	17	0.8	(=8/10)	
Screw	17	0.8	(=8/10)	
Screw	18	0.9	(=9/10)	
Screw	20	1	(=10/10)	

NTILE

- NTILE(n)
  - Allows splitting each partition in n subgroups (if it is possible) containing the same number of records. An identifier is associated to each subgroup.

NTILE example

- Partition the items according to the type and split each partition in 3 sub-groups with the same number of data. In each partition the rows are ordered by the weight of items

NTILE example

```
SELECT Type, Weight,
   NTILE(3) OVER (
      PARTITION BY Type
      ORDER BY Weight
      )  AS Ntile3Weight
FROM ITEM;
```

NTILE example

Type	Weight	RowNumberWeight		
Bar	12	1		Partition 1
Gear	19	1		Partition 2
Screw	12	1		Partition 3
Screw	14	1	Subgroup 1	
Screw	16	1		
Screw	16	1		
Screw	16	2		
Screw	16	2	Subgroup 2	
Screw	17	2		
Screw	17	3		
Screw	18	3	Subgroup 3	
Screw	20	3		

Materialized views

Data Base and Data Mining Group of Politecnico di Torino

Materialized views

- The result is **precomputed** and stored on the disk
- They improve **response times**
  - Aggregations and joins are precomputed
- Usually they are associated to queries with **aggregations**
- They may be used also for non aggregating queries
- Materialized views can be used as a **table** in any query

Query rewriting

- The DBMS can change the execution of a query to **optimize performance**
- Materialized views can be **automatically** used by the DBMS **without user intervention**
  - Materialized views help answering queries very similar to the query which created them

Creating materialized views

CREATE MATERIALIZED VIEW *Name*
[BUILD {IMMEDIATE|DEFERRED}]
[REFRESH {COMPLETE|FAST|FORCE|NEVER}
{ON COMMIT|ON DEMAND}]
[ENABLE QUERY REWRITE]
AS
Query

Creating materialized views

- *Name*
  - materialized view **name**
- *Query*
  - query associated to the materialized view (i.e., query that **creates** the materialized view)

Creating materialized views

- BUILD
  - IMMEDIATE
    - **creates** the materialized view and **immediately loads** the query results into the view
  - DEFERRED
    - **creates** the materialized view but does **not** immediately load the query results into the view

Creating materialized views

- REFRESH
  - COMPLETE
    - **recomputes** the query result by executing the query on **all data**
  - FAST
    - **updates** the content of the materialized view using the changes **since the last refresh**

Creating materialized views

- REFRESH
  - FORCE
    - when possible, the **FAST** refresh is performed
    - otherwise the **COMPLETE** refresh is performed
  - NEVER
    - the content of the materialized view is **not updated** using Oracle standard procedures

Materialized views options

- ON COMMIT
  - an **automatic refresh** is performed when SQL operations affect the materialized view content
- ON DEMAND
  - the refresh is performed only upon explicit **request** of the user issuing the command
    - DBMS_MVIEW.REFRESH

Materialized views options

- ENABLE QUERY REWRITE
  - enables the DBMS to automatically use the materialized view as a basic block (i.e., a table) to improve other queries performance
  - available only in the high-end versions of DBMS (e.g., not available in Oracle Express)
  - when unavailable, the query must be rewritten by the user to access the materialized view

Creation constraints

- Depending on the DBMS and the query, you can create a materialized view associated to the query if some constraints are satisfied
  - constraints on the aggregating attributes
  - constraints on the tables and the joins
  - etc.
  - you must be aware of the constraint existence!

Materialized view example

- Tables
  - SUPPLIERS(**Cod_S**, Name, SLocation)
  - ITEM(**Cod_I**, Type, Color)
  - PROJECTS(**Cod_P**, Name, PLocation)
  - FACTS(**Cod_S**, **Cod_I**, **Cod_P**, Measure)

Materialized view example

- The materialized view query is
  - SELECT Cod_S, Cod_I, SUM(Measure)
    FROM Facts
    GROUP BY Cod_S, Cod_I;
- Options
  - Immediate data loading
  - Complete refresh only upon user request
  - The DBMS can use the materialized view to optimize other queries

Materialized view example

```
CREATE MATERIALIZED VIEW Sup_Item_Sum
BUILD IMMEDIATE
REFRESH  COMPLETE ON DEMAND
ENABLE QUERY REWRITE
AS
    SELECT Cod_S, Cod_I, SUM(Measure)
    FROM Facts
    GROUP BY Cod_S, Cod_I;
```

Fast refresh

- Requires proper structures to log changes to the tables involved by the materialized view query
- MATERIALIZED VIEW LOG
  - there is a log for each table of a materialized view
  - each log is associated to a single table and some of its attributes
  - it stores changes to the materialized view table

Fast refresh

- The REFRESH FAST option can be used only if the materialized view query satisfies some constraints
  - materialized view **logs** for the tables and attributes of the query must exist
  - when the GROUP BY clause is used, in the SELECT statement an **aggregation** function must be specified (e.g., COUNT, SUM, ...)

Materialized view log example

- Create a materialized view log associated to the FACTS table, on Cod_S, Cod_I and MEASURE attributes
  - enable the options SEQUENCE and ROWID
  - enable new values handling

Materialized view log example

```
CREATE MATERIALIZED VIEW LOG
   ON Facts
WITH SEQUENCE, ROWID
   (Cod_S, Cod_I, Measure)
INCLUDING NEW VALUES;
```

Example with fast refresh option

- The materialized view query is
  - SELECT Cod_S, Cod_I, SUM(Measure)
    FROM Facts
    GROUP BY Cod_S, Cod_I;
- Options
  - Immediate data loading
  - Automatic fast refresh
  - The DBMS can use the materialized view to optimize other queries

Example with fast refresh option

```
CREATE MATERIALIZED VIEW LOG ON Facts
WITH SEQUENCE, ROWID (Cod_S, Cod_I, Measure)
INCLUDING NEW VALUES;

CREATE MATERIALIZED VIEW Sup_Item_Sum2
BUILD IMMEDIATE
REFRESH  FAST ON COMMIT
ENABLE QUERY REWRITE
AS
    SELECT Cod_S, Cod_I, SUM(Measure)
    FROM Facts
    GROUP BY Cod_S, Cod_I;
```

Fast refreshing materialized views

- The user or a system job can request the materialized view update by issuing the command
  - DBMS_MVIEW.REFRESH('*view*', {'C'/'F'})
    - *view*: name of the view to update
    - 'C': COMPLETE refresh
    - 'F': FAST refresh

Fast refreshing materialized views

- Example
  - COMPLETE refresh of the materialized view "Sup_Item_Sum"

```
EXECUTE DBMS_MVIEW.REFRESH('Sup_Item_Sum', 'C');
```

Changing and deleting views

- Changing
  - ALTER MATERIALIZED VIEW *name* *options*;
- Deleting
  - DROP MATERIALIZED VIEW *name;*

Analyzing materialized views

- The command DBMS_MVIEW.EXPLAIN_MVIEW allows the materialized view inspection
  - refresh type
  - operations on which the fast refresh is enabled
  - query rewrite status (enabled, allowed, disabled)
  - errors

Execution plan

- Analyzing the execution plan of frequent queries allows us to know whether materialized views are used
- Query execution plans can be shown
  - enabling the auto trace in SQLPLUS> **set autotrace on;**
  - clicking on the **Explain** link in the Oracle web interface

Results Explain Describe Saved SQL History

Query Plan

Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			5	1	14	65
HASH	GROUP BY		5	1	14	65
HASH JOIN			7,809	1	13	101,517

Physical Design

Elena Baralis
Politecnico di Torino

Physical Design

- Workload Characterization:
  - Queries with aggregates that require access to a large portion of each table
  - Read-only access
  - Periodic update of data with eventual rebuilding of access physical structures (indexes, views, etc.)
- Physical Structures
  - Non-traditional types of indexes
    - Bitmap indexes, Join indexes, Bitmapped join indexes
    - B^+-tree index is not well-suited for:
      - Attributes with low cardinality domain
      - Queries with low selectivity
  - Materialized views:
    - Require the presence of an optimizer able to exploit them

Elena Baralis
Politecnico di Torino

Physical Design

- Optimizer characteristics
  - Must consider statistics while defining data access strategy (cost based)
  - Capability of aggregate navigation
- Physical design procedure
  - Selection of suited data structures to support the most frequent queries (or the most relevant)
  - Choice of structures that contribute to improve more queries at a time
  - Constraints:
    - Disk space
    - Available time for updating data

Elena Baralis
Politecnico di Torino

Physical Design

- Tuning:
  - *A posteriori* variation of support physical structures
  - Requires tools for workload monitoring
  - Often required for OLAP applications
- Parallelism
  - Data fragmentation
  - Queries parallelization
    - inter-query
    - intra-query
  - Join and Group By operations suitable to parallel execution

Elena Baralis
Politecnico di Torino

Physical Access Structures

- Physical access structures describe how data is stored on disk to provide efficient query execution
  - SQL select, update, …
- In relational systems
  - Physical data storage
    - Sequential structures (heap file, ordered sequential structure)
    - Hash structures
  - Indexing to increase access efficiency
    - Tree structures (B-Tree, B^+-Tree)
    - Unclustered hash index
    - Bitmap index

Elena Baralis
Politecnico di Torino

Heap file

- Tuples are sequenced in *insertion order*
  - insert is typically an *append* at the end of the file
- *All* the space in a block is completely exploited before starting a new block
- Delete or update may cause wasted space
  - Tuple deletion may leave unused space
  - Updated tuple may not fit if new values have larger size
- Sequential reading/writing is very efficient
- Frequently used in relational DBMS
  - jointly with unclustered (secondary) indices to support search and sort operations

Elena Baralis
Politecnico di Torino

Ordered sequential structures

- The order in which tuples are written depends on the value of a given key, called *sort key*
  - A sort key may contain one or more attributes
    - the sort key may be the primary key
- Appropriate for
  - Sort and group by operations on the sort key
  - Search operations on the sort key
  - Join operations on the sort key
    - when sorting is used for join

Elena Baralis
Politecnico di Torino

Ordered sequential structures

- Problem
  - preserving the sort order when inserting new tuples
    - it may also hold for update
- Solution
  - Leaving a percentage of free space in each block during table creation
    - On insertion, dynamic (re)sorting in main memory of tuples into a block
- Alternative solution
  - Overflow file containing tuples which do not fit into the correct block

Elena Baralis
Politecnico di Torino

Ordered sequential structures

- Typically used with B+-Tree clustered (primary) indices
  - the index key is the sort key
- Used by the DBMS to store intermediate operation results

Elena Baralis
Politecnico di Torino

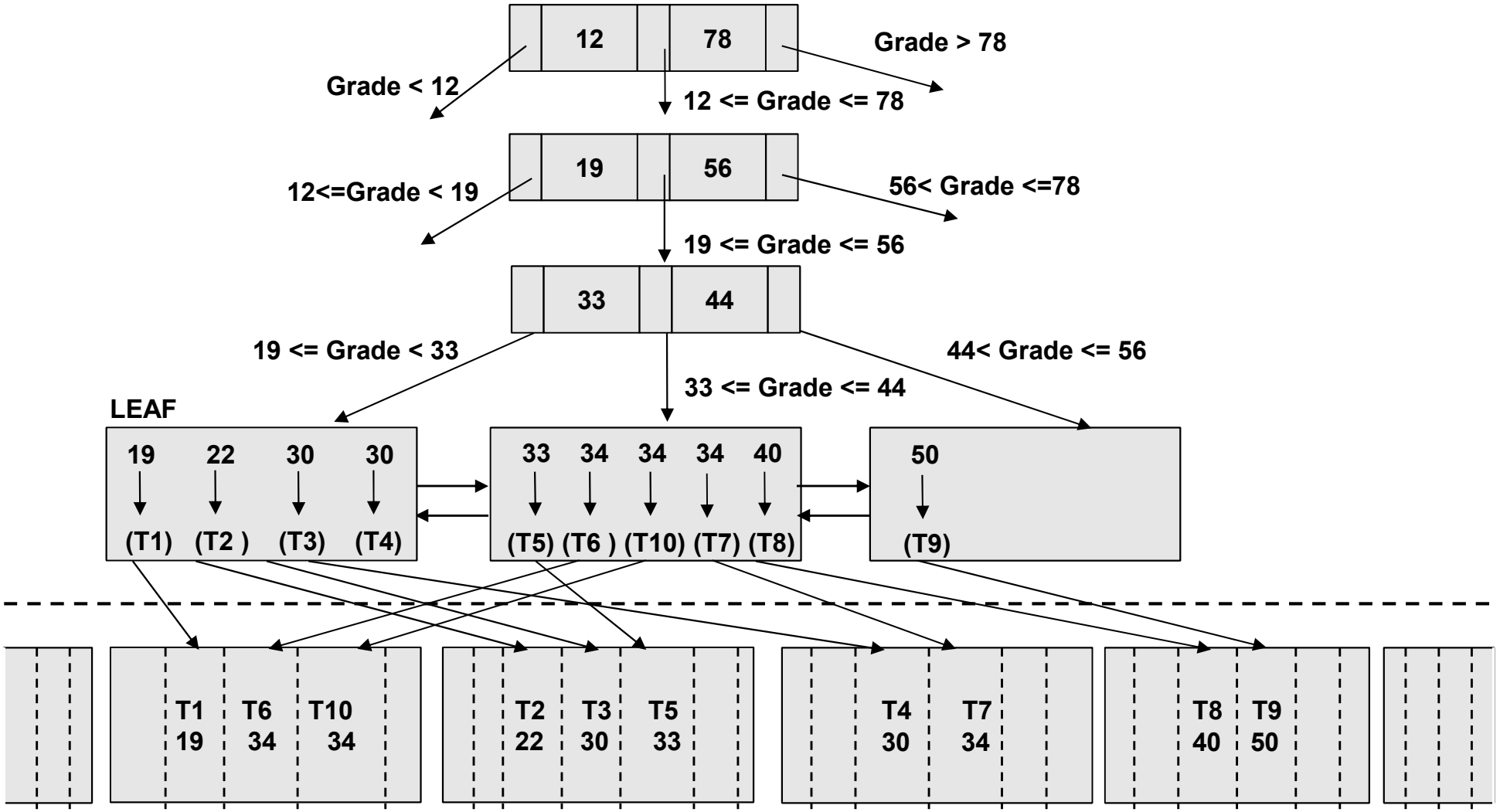
B+Tree

- Provide “direct” access to data based on the value of a key field
  - The key includes one or more attributes
- B stands for *balanced*
  - Leaves are all at the same distance from the root
  - Access time is constant, regardless of the searched value
- Unclustered
  - The leaf contains physical pointers to actual data
    - The position of tuples in a file is totally unconstrained
- Clustered
  - The tuple is contained into the leaf node
    - Constrains the physical position of tuples in a given leaf node
    - Typically used for primary key indexing

Elena Baralis
Politecnico di Torino

Example: Unclustered B+-Tree index

STUDENT (<u>StudentId</u>, Name, *Grade*)

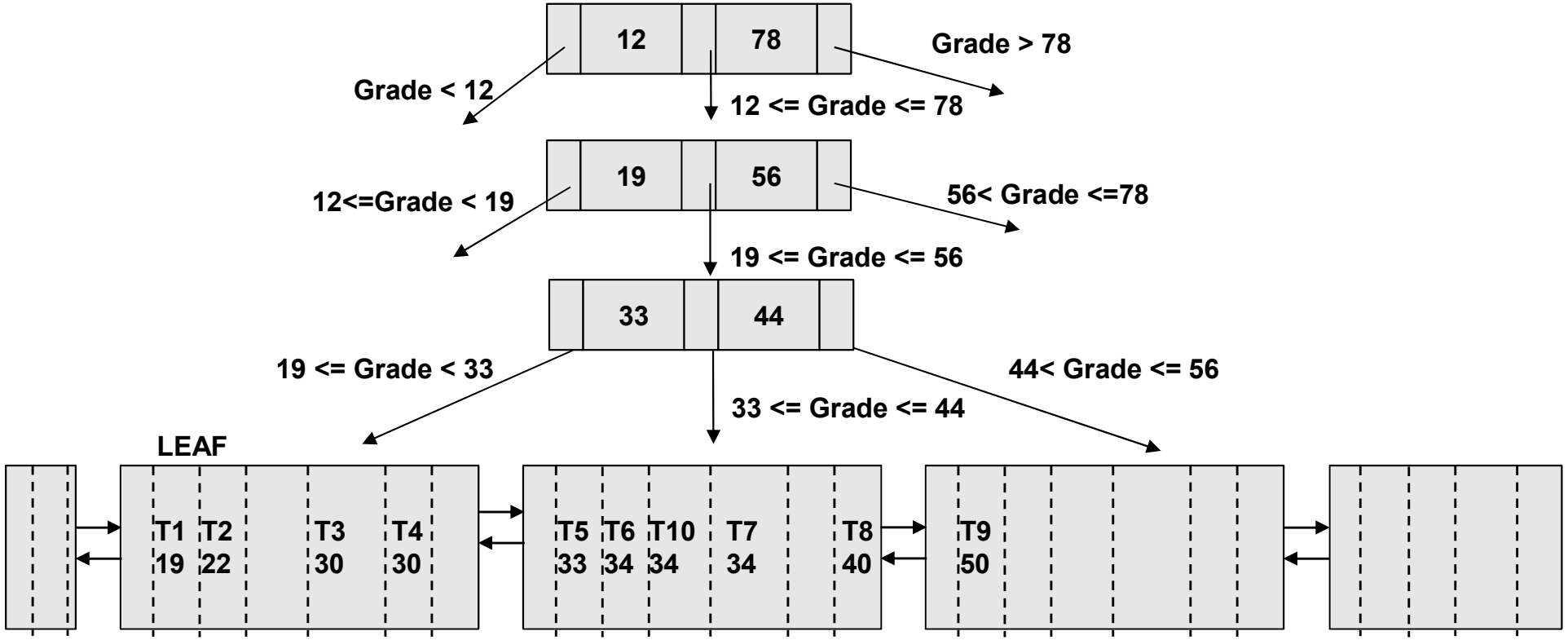


DATA FILE FOR STUDENT TABLE

Elena Baralis
Politecnico di Torino

Example: Clustered B+-Tree index

STUDENT (StudentId, Name, *Grade*)



DATA FILE FOR STUDENT TABLE

Elena Baralis
Politecnico di Torino

Advantages and disadvantages

- Advantages
  - Very efficient for range queries
  - Appropriate for sequential scan in the order of the key field
    - Always for clustered, not guaranteed otherwise
- Disadvantages
  - Insertions may require a split of a leaf
    - possibly, also of intermediate nodes
    - computationally intensive
  - Deletions may require merging uncrowded nodes and re-balancing

Elena Baralis
Politecnico di Torino

Bitmap Index

- Composed of a bit matrix
  - A column for each different value of the indexed attribute domain
  - A row for each tuple (RID in the table)
  - The position (*i,j*) is 1 if the tuple *i* has value *j*, 0 otherwise

Example: Index on the field *Position* in the *Employee* table
Engineer – Consultant – Manager – Programmer - Assistant – Accountant

RID	Eng.	Cons.	Man.	Prog.	Assis.	Acc.
1	0	0	1	0	0	0
2	0	0	0	1	0	0
3	0	0	0	0	1	0
4	0	0	0	1	0	0
5	0	0	0	0	0	1

Taken from Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Bitmap index

- It guarantees direct and efficient access to data based on the value of a *key field*
  - It is based on a *bit matrix*
- The bit matrix references data rows by means of RIDs (Row IDentifiers)
  - Actual data is stored in a separate structure
  - Position of tuples is not constrained

Elena Baralis
Politecnico di Torino

Bitmap index

- The bit matrix has
  - One column for each different value of the indexed attribute
  - One row for each tuple
- Position (i, j) of the matrix is
  - 1 if tuple i takes value j
  - 0 otherwise

RID	Val_1	Val_2	…	Val_n
1	0	0	…	1
2	0	0	…	0
3	0	0	…	1
4	1	0	…	0
5	0	1	…	0

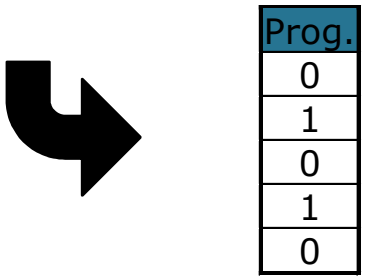
Elena Baralis
Politecnico di Torino

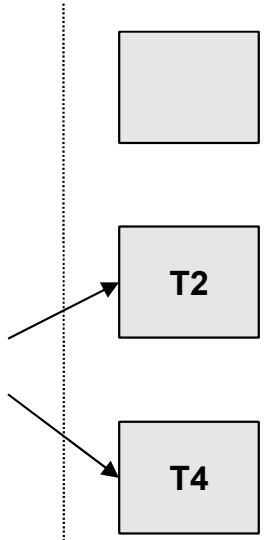
Example: Bitmap index

EMPLOYEE (Employeeld, Name, *Job*)

Domain of Job attribute = {Engineer, Consultant, Manager, Programmer, Secretary, Accountant}

RID	Eng.	Cons.	Man.	Prog.	Secr.	Acc.
1	0	0	1	0	0	0
2	0	0	0	1	0	0
3	0	0	0	0	1	0
4	0	0	0	1	0	0
5	1	0	0	0	0	0





Example: Index on the field *Position* in the *Employee* table

Engineer – Consultant – Manager – Programmer - Assistant – Accountant

DATA FILE FOR EMPLOYEE TABLE

Elena Baralis
Politecnico di Torino

Bitmap index

- Advantages
  - Very efficient for boolean expressions of predicates
    - Reduced to bit operations on bitmaps
  - Appropriate for attributes with limited domain cardinality
- Disadvantages
  - Not used for continuous attributes
  - Required space grows significantly with domain cardinality

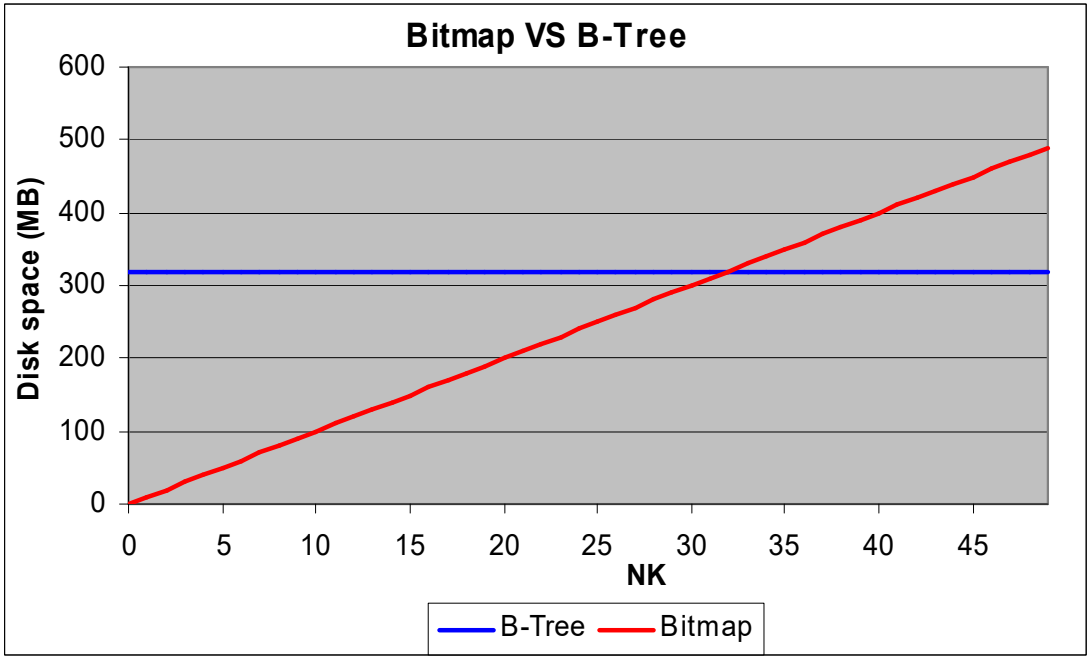
Elena Baralis
Politecnico di Torino

Bitmap Index

- Well-suited for dimensional attributes with low-cardinality domain
  - Storage requires limited space
  - If domain cardinality (NK) grows, the required space grows as well

B-tree NR×*Len*(Pointer)
Bitmap NR × NK × 1 bit

Len(*Pointer*) = 4×8 bit



Taken from Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

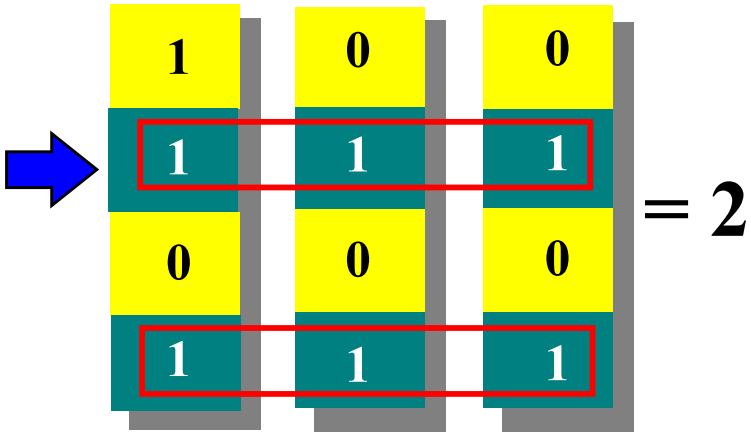
Elena Baralis
Politecnico di Torino

Bitmap Index

- Efficient for verifying Boolean expressions of predicates
  - Bit-wise and/or on bitmaps

Example: "*How many males in Romagna are insured?*"

RID	Gender	Ins.	Region
1	M	No	LO
2	M	Yes	E/R
3	F	No	LA
4	M	Yes	E/R



Example taken from Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Join index

- Precomputes the join between two tables
  - Stores the RID couples of tuples that satisfy the join predicate

Sale

IDStore	IDWeek	IDProd	Quantity	Income
123	13	41	100	10.00
123	24	17	150	15.00
367	36	94	350	35.00
367	27	11	120	12.00

Store

IDStore	Store	City	Country	SaleManager
123	N1	RM	I	R1
367	N3	MI	I	R2

RID Sale	RID Store
1	1
2	1
3	2
4	2

RID couples that satisfy Join condition:
Sells. IDStore = Store.IDStore

Example taken from Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Star index

- Precomputes the join between two or more tables
  - Stores the RID n-uples of the tuples that satisfy the join predicate

Week

WeekID	Week	Month
13	Jan1	Jan.
24	Jan2	Jan.

Shop

StoreID	Store	City	Country
123	N1	RM	I
367	N3	MI	I

Sale

StoreID	WeekID	ProdID	Quantity	Income
123	13	41	100	10.00
123	24	17	150	15.00
367	13	17	350	35.00
367	24	41	120	12.00

Product

ProdID	Product	Type	Category	Supplier
41	P1	A	X	F1
17	P2	A	X	F1

SaleRID	SRID	WID	PID
1	1	1	1
2	1	2	2
3	2	1	2
4	2	2	1

Example taken from Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Star index

- Advantages
  - Efficient computation of Joins involving initial index columns (or all columns)
- Disadvantages
  - Useful only for specific Join combinations
    - It is necessary to store a high number of indexes in order to achieve generalization
  - The storage space may become big
    - Joins always include the fact table

Elena Baralis
Politecnico di Torino

Bitmapped join index

- Bit matrix that precomputes the join between a dimension and the fact table
  - A column for each dimension RID
  - A row for each fact table RID
  - The position (*i,j*) is 1 if the tuple *i* of the dimension is joined with the tuple *j* of the fact table, 0 otherwise
- Can be used together with traditional bitmap indexes to compute complex queries with conditions on dimensions and multiple joins

RID of the STORE table

RID of SELLS table

RID	1	2	3	…
1	1	0	0	…
2	0	1	0	…
3	0	0	1	…
4	0	1	0	…
5	1	0	0	…
…	…	…	…	…

The row 4 of table SELLS is joined with the row 2 of the table STORE

Taken from Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Bitmapped join index

Executing a bit-wise OR, the system obtains RID_i that satisfy all conditions for a dimensional table

Bitmap Index on attribute $DT_i.b_i$

1

RID	Val_1	Val_2	…	Val_i	…	Val_h
1	1	0	…	0	…	0
2	0	0	…	0	…	1
3	0	1	…	0	…	0
4	0	0	…	1	…	0
5	0	0	…	1	…	0
…	…	…	…	…	…	…

Bitmapped join index $FT.a_i = DT_i.a_i$

RID	1	2	3	4	5	…
1	0	0	0	1	0	…
2	0	0	0	1	0	…
3	0	1	1	0	0	…
4	1	0	0	0	0	…
5	0	0	0	0	1	…
6	0	1	0	0	0	…
…	…	…	…	…	…	…

2

RID 4: 1, 1, 0, 0, 0, 0, …

3 Bit-wise OR

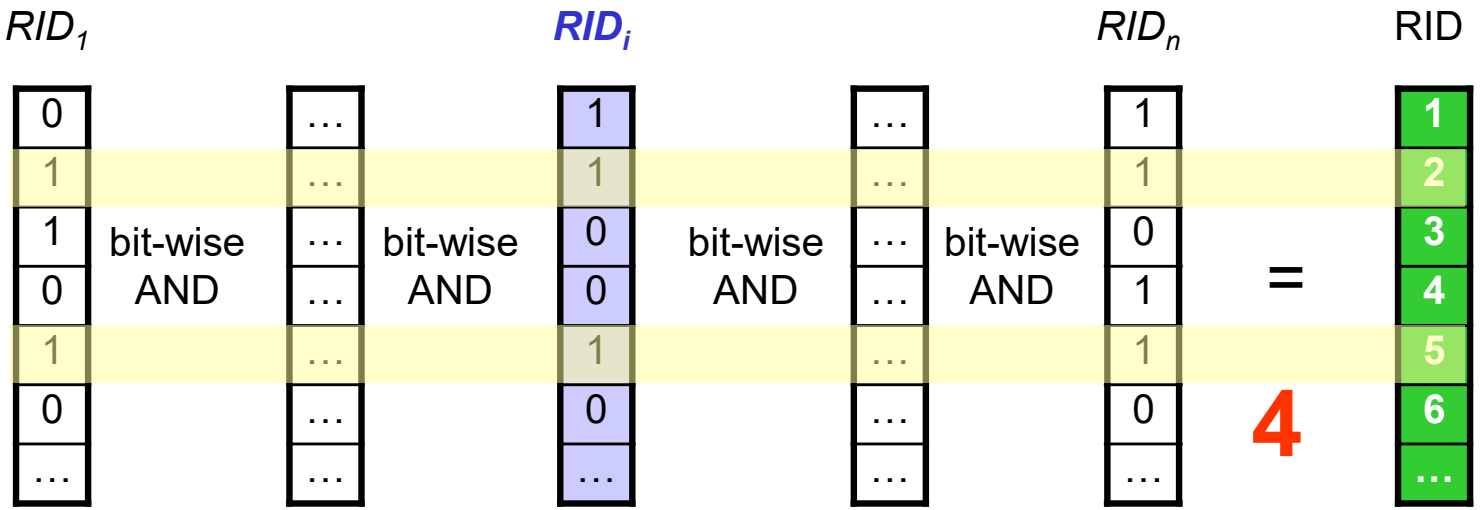
RID 5: 0, 0, 0, 0, 1, 0, …

= RID_i: 1, 1, 0, 0, 1, 0, …

Taken from Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

Bitmapped join index



The fact table tuples that satisfy the query are computed with a bit-wise AND between the *n* vector previously created

RIDs that satisfy all conditions

Taken from Golfarelli, Rizzi,"Data warehouse, teoria e pratica della progettazione", McGraw Hill 2006

Elena Baralis
Politecnico di Torino

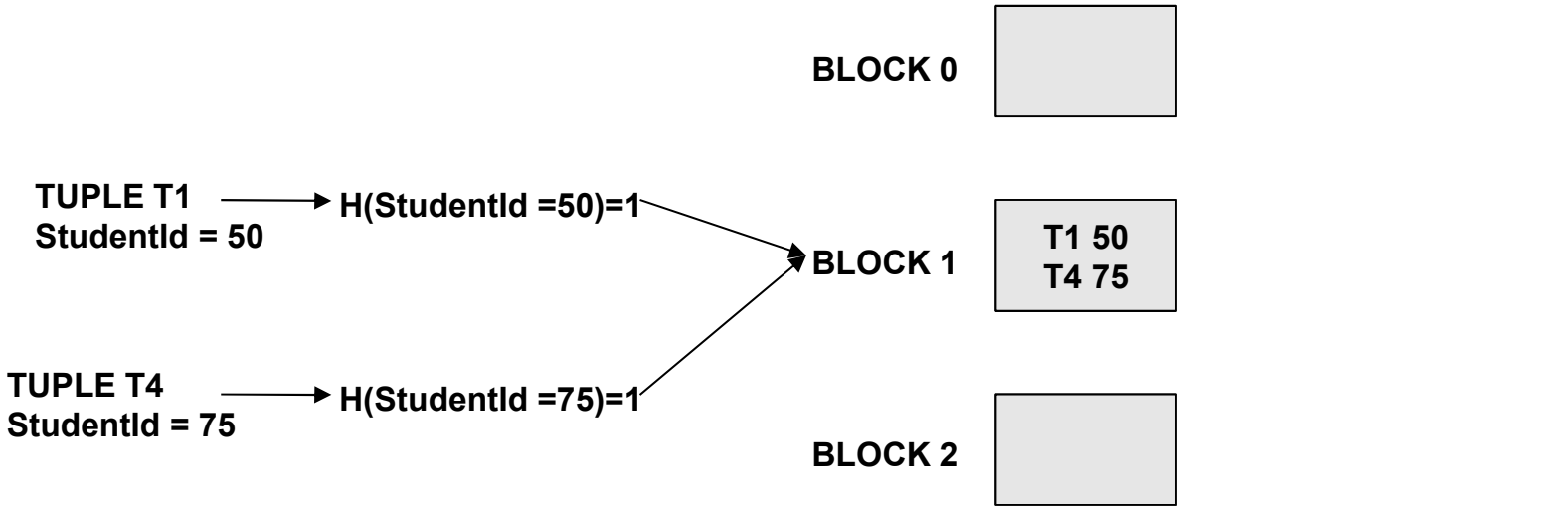
Hash structure

- It guarantees direct and efficient access to data based on the value of a *key field*
  - The hash key may include one or more attributes
- Suppose the hash structure has B blocks
  - The hash function is applied to the key field value of a record
    - It returns a value between 0 and B-1 which defines the position of the record
  - Blocks should never be completely filled
    - To allow new data insertion

Elena Baralis
Politecnico di Torino

Example: hash index

STUDENT (*StudentId*, Name, Grade)



DATA FILE FOR STUDENT TABLE

Elena Baralis
Politecnico di Torino

Hash index

- Advantages
  - Very efficient for queries with equality predicate on the key
  - No sorting of disk blocks is required
- Disadvantages
  - Inefficient for range queries
  - Collisions may occur

Elena Baralis
Politecnico di Torino

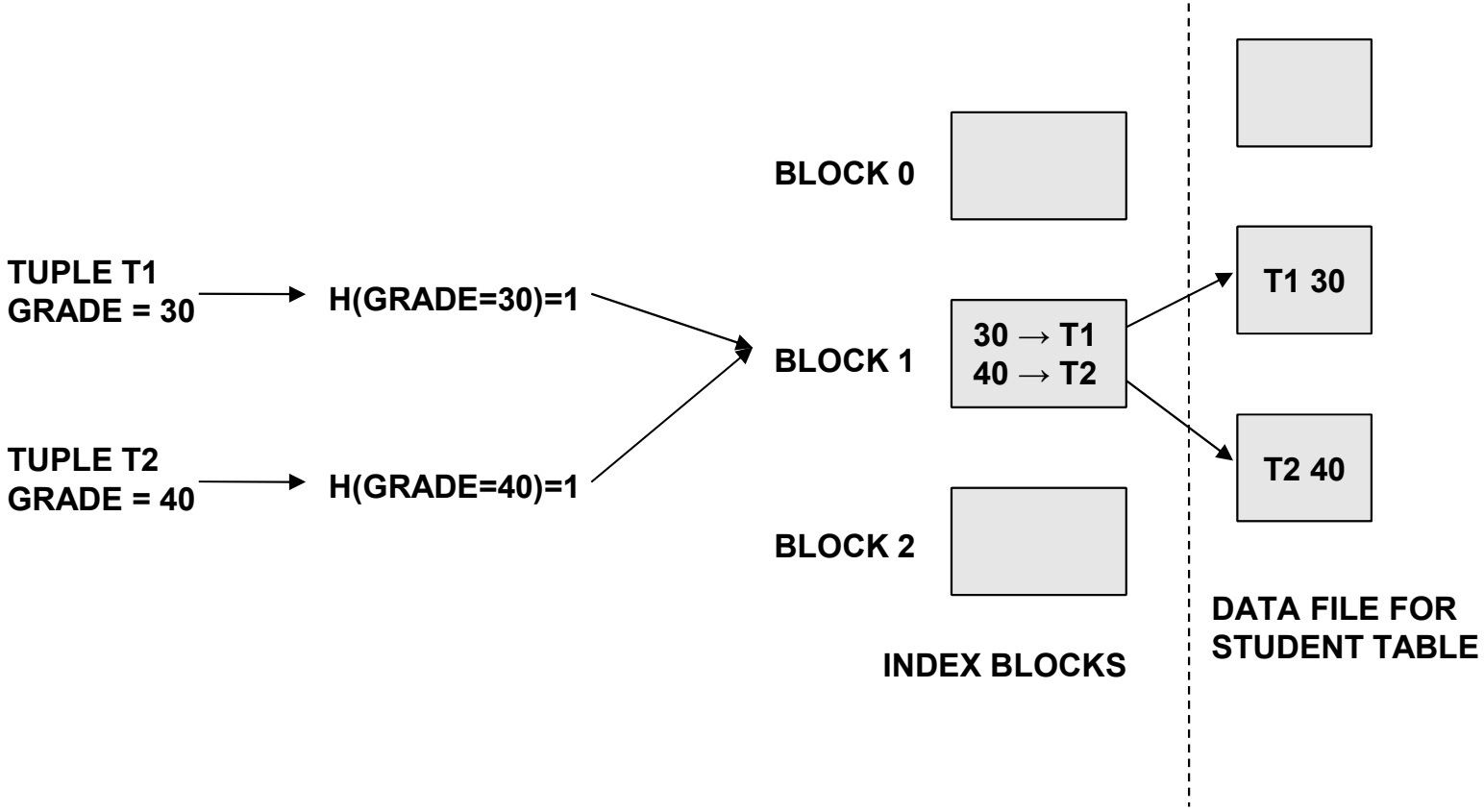
Unclustered hash index

- It guarantees direct and efficient access to data based on the value of a *key field*
  - Similar to hash index
- Blocks contain pointers to data
  - Actual data is stored in a separate structure
  - Position of tuples is not constrained to a block
    - Different from hash index

Elena Baralis
Politecnico di Torino

Example: Unclustered hash index

STUDENT (StudentId, Name, *Grade*)



Elena Baralis
Politecnico di Torino

Index choice

- Indexing of dimensions
  - Attribute frequently involved in selection predicates
  - If the domain has high cardinality, B-tree index
  - If the domain has low cardinality, bitmap index
- Indexes for Join
  - It is seldom necessary to index only external keys of the fact table
  - Be careful when using Star Join Indexes (problems related to column ordering)
  - Bitmapped join index are recommended
- Indexes for Group By
  - Use Materialized Views

Elena Baralis
Politecnico di Torino