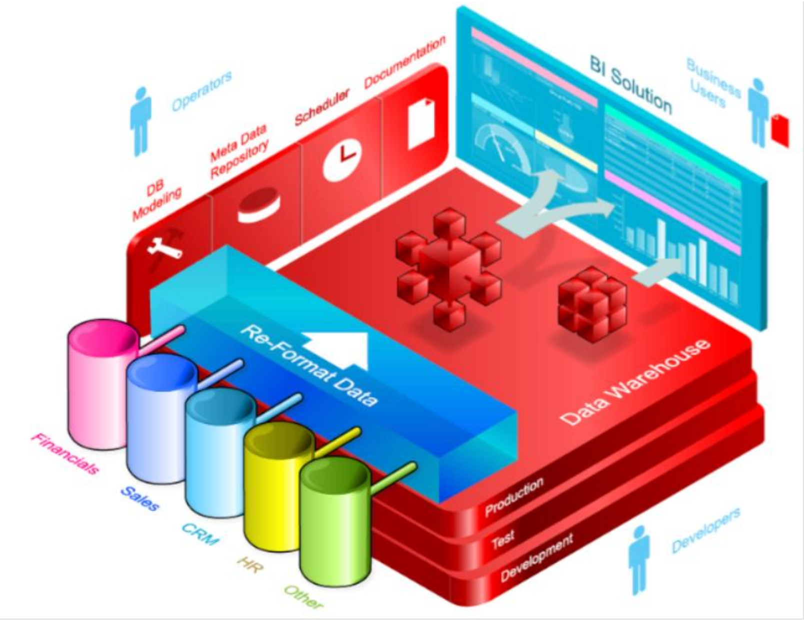
INFORMATIONAL INFORMATION SYSTEMS (IIS)

DATA WAREHOUSE



Rielaborato a partire da «Sistemi informativi aziendali – struttura e processi»
Autori - Maurizio Pighin, Anna Marzona
Casa editrice Pearson Italia

GOALS Of An Informational I.S.

- **Exploit operational data to create useful information for decision making and strategic planning;**
- **Enrich operational data with other sources;**
- **Overcome the limit of basic «decision making approach» based on spread sheets and reports**

Reporting

- × Static
- × Time consuming
- × Limited information
- × Biased information
- × …

Spreadsheets

- × Time consuming
- × Complex
- × Assuring data complexity is hard
- × Proliferation of custom-made spreadsheets
- × Limited quantity of data that can be stored

GOALS Of An Informational I.S.

- **Exploit operational data to create useful information for decision making and strategic planning;**
- **Enrich operational data with other sources;**
- **Overcome the limit of basic «decision making approach» based on spread sheets and reports**

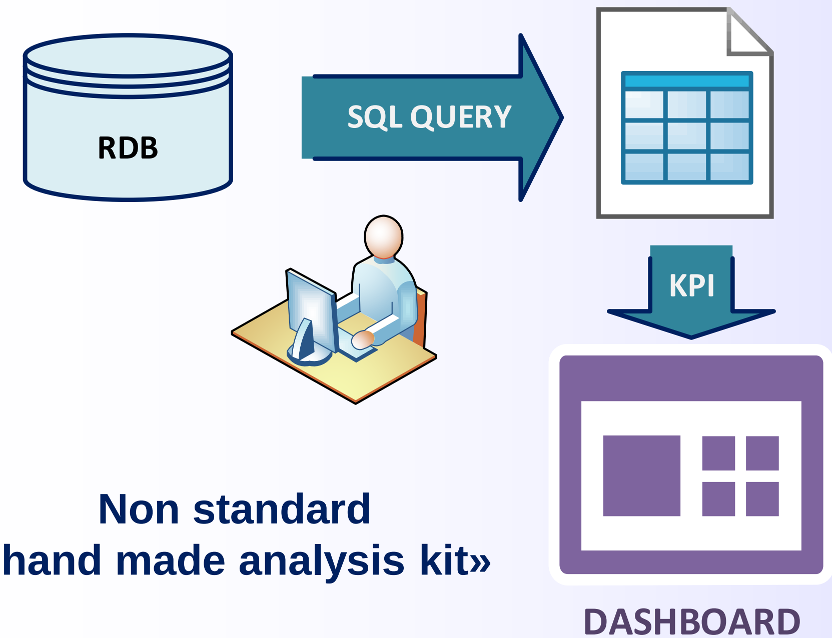
Reporting

- × Static
- × Time consuming
- × Limited information
- × Biased information
- × …

Time consuming «never ending cycle»

GOALS Of An Informational I.S.

- **Exploit operational data to create useful information for decision making and strategic planning;**
- **Enrich operational data with other sources;**
- **Overcome the limit of basic «decision making approach» based on spread sheets and reports**



Non standard «hand made analysis kit»

Spreadsheets

× Time consuming

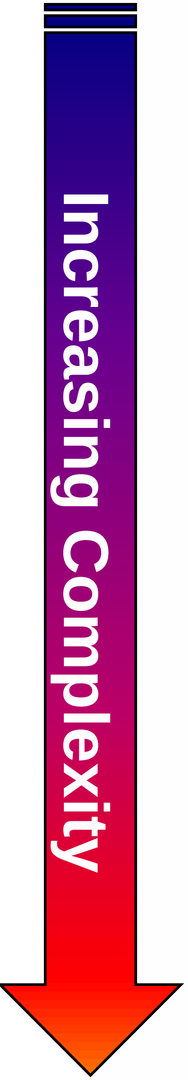
× Complex

× Assuring data complexity is hard

× Proliferation of custom-made spreadsheets

× Limited quantity of data that can be stored

Type of Queries



- **Query on operational IS are prefixed, precise and based on few data**
  - *Which is the discount rate of a customer?*
  - *Which Work Orders have been assigned to a department?*
  - *Which products are below the Reorder Point?*
  - *Which invoices have not been paid yet?*
- ***Query on informational IS are fuzzy, complex based on several data and depends on the «reasoning of the decision makers»***
  - *Has the margin of Product X increased? By how much?*
  - *Is there a correlation between educational level and purchasing propensity?*
  - *Which is the impact of transportation costs? How these costs change depending on the size and shape of transported material?*

Requirements of an IIS

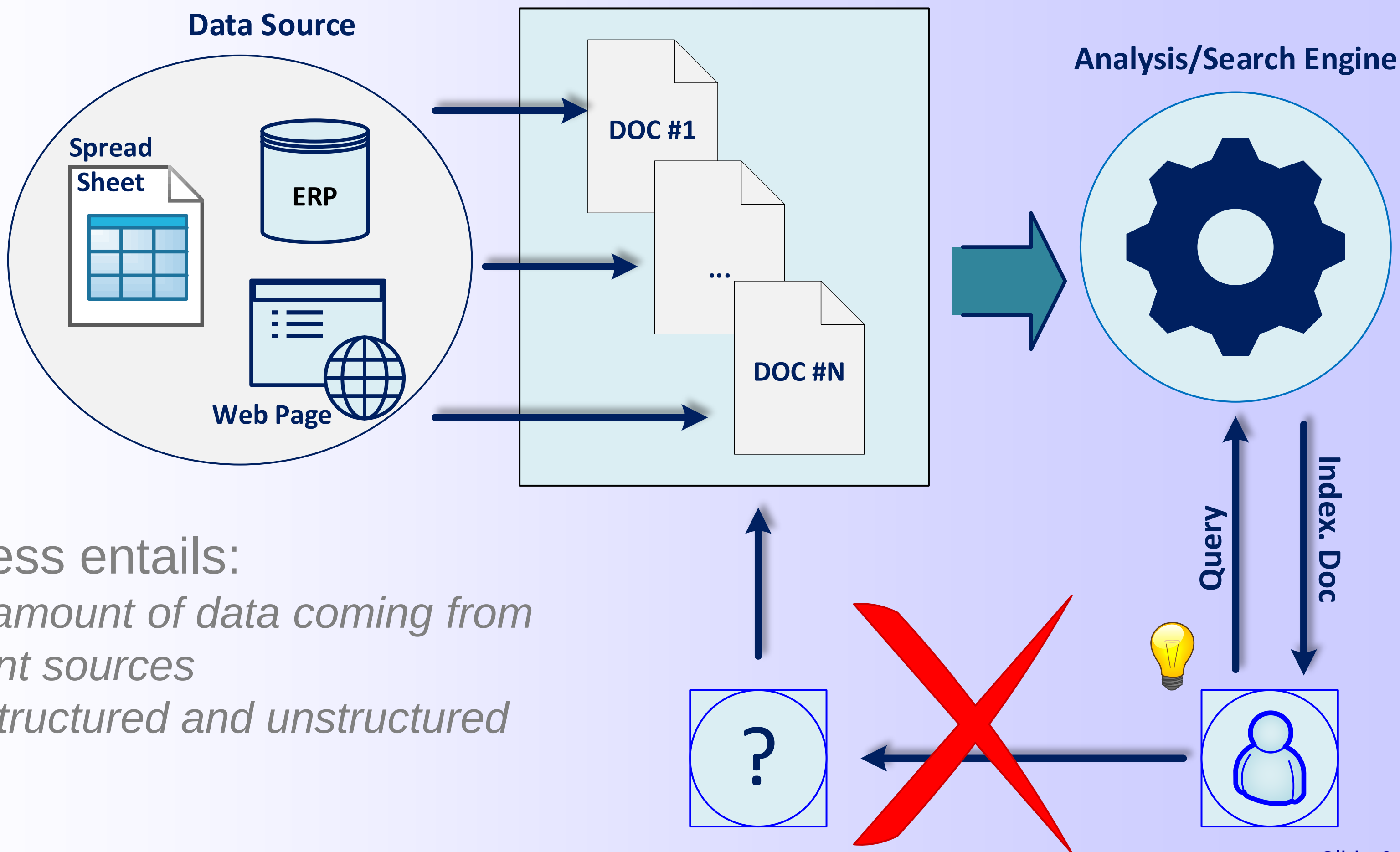
- **Data Base**
  - **Intuitiveness –** storing procedures are easy to be understood
  - **Efficiency** – Query are executed very rapidly
  - **Data coming from different sources**
  - **Data consistency –** Data are cleaned and updated in a consistent way
- **Data Analysis Tools**
  - **Reporting**
  - **Dash boarding**
  - **Tools for Interactive Analysis and for easy queries formulation**
  - **Data Mining (Optional)**

… Some Terminology

- **Data Warehouse**
  - *The Data Base (mostly multi-dimensional)*
- **Data Warehousing**
  - *Tools & Techniques to build and maintain a Data Warehouse*
- **Decision Support System (DSS)**
  - *Informatic tools, used during the decision making process, to help managers in the extraction and analysis of data that are stored in the operational Information system, or coming from other data sources*
- **Data Mining**
  - *Tools & Techniques to extract/find hidden/unknown/unexpected relationship among the data*
- **Business Intelligence**
  - *Extraction of data needed for business analysis, generated by transactions at the operational level*
- **Knowledge Management**
  - *Data filing and data retrieval extended to all data and documents (also non structured) generated by the business*

Knowledge Management

- Techniques used to **«Reorganize knowledge generated by a business»** in a way that allows an easy and rapid data and documents retrieval



- The process entails:
  - *Huge amount of data coming from different sources*
  - *Both structured and unstructured data*

A Comparison of IIS and OIS

Informational I.S.

- **GOALS**
  - ***Create knowledge from row data***
  - ***Describe historical trends, understand root causes and formulates solutions***
  - ***Evaluate/Simulate the effects of strategic actions***
- **STRUCTURE**
  - ***Data are <u>organized in terms of events</u> and/or subjects that are relevant for a company***

Operational I.S.

- **GOALS**
  - ***Execution, simplification and automation of current activities***
  - ***Improve efficiency of daily standard and activities***
- **STRUCTURE**
  - ***Data are organized <u>accordingly to the processes</u> that generate transactions that are meaningful for the company.***

A Comparison of IIS and OIS

IIS

- Historicity
  - *Historical data*
  - *Many years are covered*
- Detail Level
  - *Highly aggregated data*
  - *Many hierarchies are possible*
- Data Accessibility
  - *Read only accesses*
  - *Data updated periodically and in batches*
- Key Users
  - *Top managers, generally with low PC programming skills*

OIS

- Historicity
  - *Current state*
  - *Data of the last few years*
- Detail Level
  - *Punctual Data*
  - *Maximum level of detail*
- Data Accessibility
  - *Interactive and continuous*
  - *Read and write*
- Key Users
  - *Operational Staff*
  - *Departments' heads*

OLTP Vs OLAP

- **OIS → On Line Transaction Processing (OLTP)**
- **IIS → On Line Analytical Processing (OLAP)**

OLTP

- ✓ Easy, predefined and short transactions
- ✓ Detailed, up to date and consistent data
- ✓ Integrated and unique RDB
- ✓ Read & Write of few records
- ✓ **ACID Properties**
  - ✓ ***Atomicity***
  - ✓ ***Consistency***
  - ✓ ***Isolation***
  - ✓ ***Durability***

OLAP

- ✓ Complex and "casual" queries
- ✓ Interactive GUI
- ✓ Historical and aggregated data
- ✓ Data coming from several database
- ✓ Data stored in a central multi-dimensional Data Warehouse
- ✓ Read only access
- ✓ Data updated at discrete time
- ✓ Generation of unknown information and knowledge
- ✓ **FASMI Properties**

ACID Properties

- A transaction is a single logical unit of work, which accesses and possibly modifies the contents of a database (using read and write operations).
- To maintain consistency in a database, before and after the transaction, ACID properties are followed.

ATOMICY (A)

– *The entire transaction takes place at once or does not takes place at all*

CONSISTENCY (C)

– *The DB must be consistent before and after the transaction*

ISOLATION (I)

– *Multiple transactions occur simultaneously and without interference*

DURABILITY (D)

– *The change of a successful transaction occurs even if a system failure occurs.*

FASMI Properties – OLAP Report 1995

F - Fast

- Fast Interactive use
- Waiting Time must not be a hurdle, it must not interrupt user’s reasoning

A - Analytic

- Dashboard & reporting function
- Statistical computation on both historic and newly generated data

S - Shared

- Used by managers of different areas, also with multiple and simultaneous accesses
- Data security must be grant

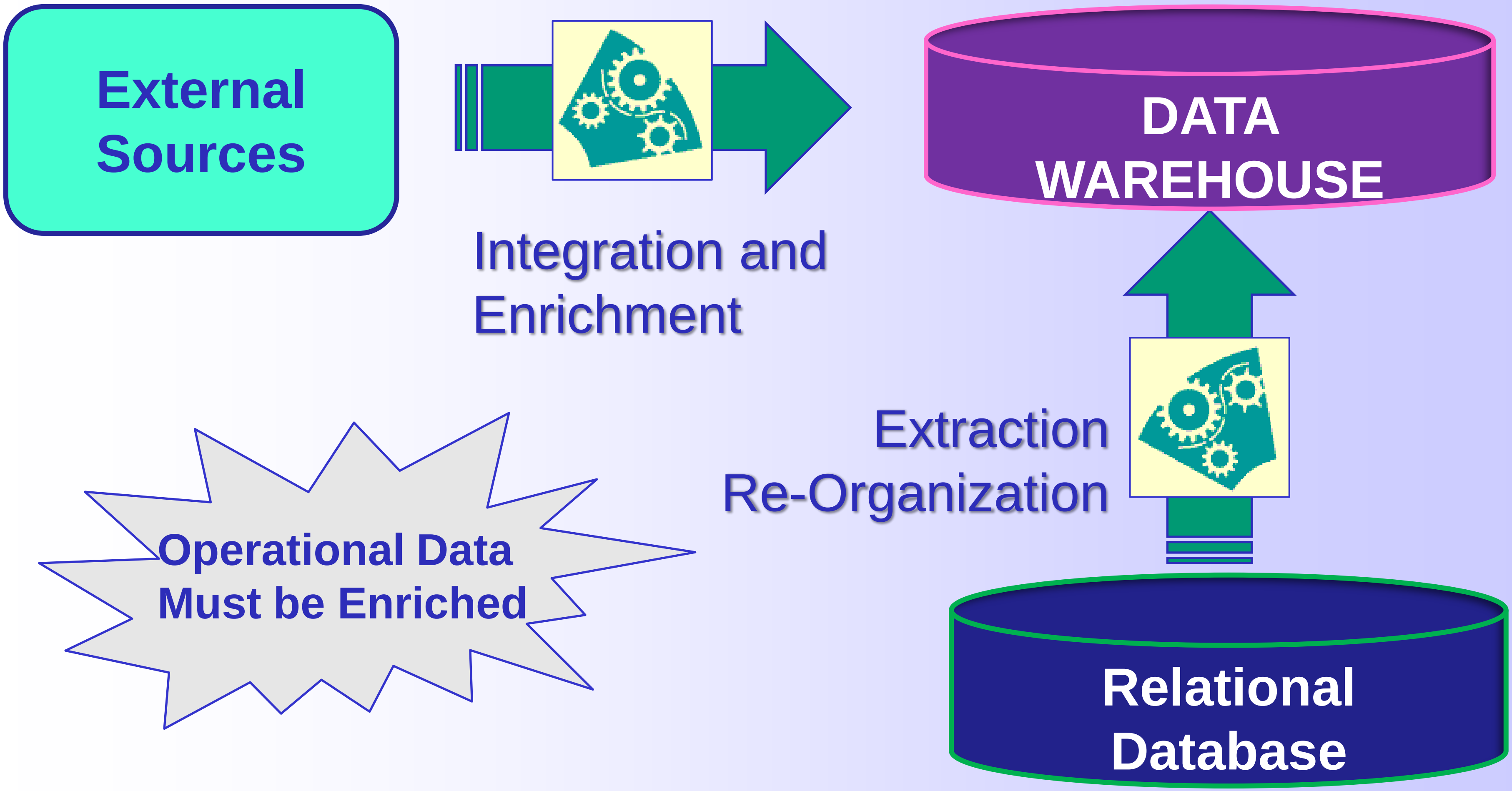
M - Multidimensional

- Same data must be considered from different perspective
- Different dimensions for the analysis

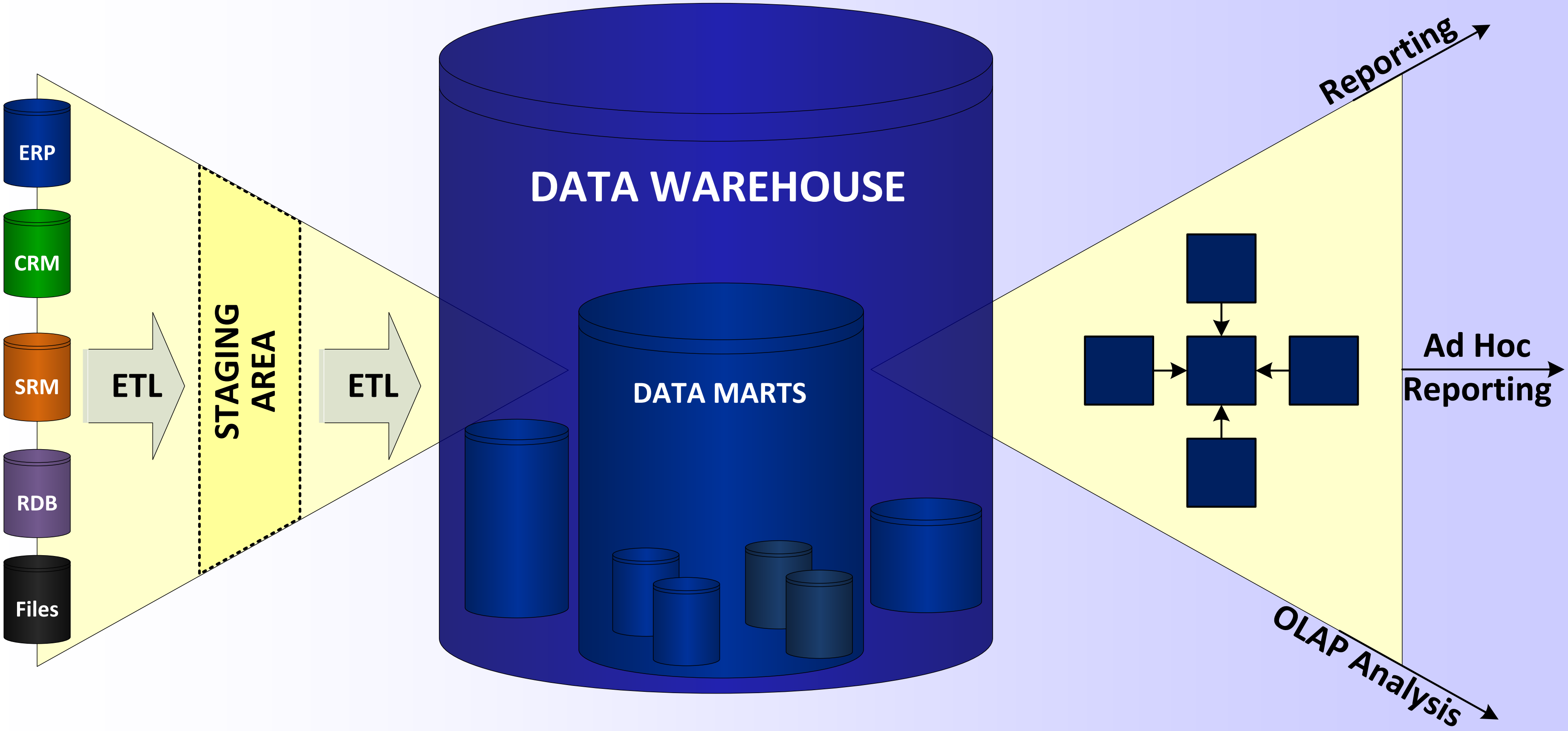
I - Informational

- Generated information must be saved and reused
- Information should be converted in knowledge

Architecture of IIS



Architecture of IIS



ETL (Extraction, Transformation Loading)

Architecture of IIS

Data comes from different Relational Databases, but also non structured data can be loaded in the Data Warehouse

Sources

- Original Input RDBs
- Access using RDBMS

Data Warehouse

- Central Multi Dimensional Data Base
- Contains all relevant data

Staging Area (optional)

- An intermediate are, where data are temporarily stored before being processed, transformed, checked , etc.
- ETL (Extraction, Transformation Loading)

Data Mart

- Sub Set of the Data Warehouse
- Smaller and specific multi-dimensional Data Bases used for specific analysis

Architecture of IIS

- **Mono Level Architecture (not a real IIS)**
  - OLAP engines works, directly, on the RDB of the OIS
  - It is a partial solution
- **Dual Level Architecture (common)**
  - Multiple Sources + Data Warehouses
  - Data Marts
- **Three Level Architecture (rare)**
  - Multiple Sources + Data Warehouses
  - Data Marts
  - **Staging Area**
    - Transfer and data transforming procedure
    - Data Analysis Tools

Multi Dimensional Model

- Data are organized in terms of **«subjects and events to which they participate»**
- Information is coded using **multi dimensional matrices (or tensors)**
  - Each matrix represents a specific event
  - Each element quantifies, through a measure, a specific event
  - Each dimension corresponds to a specific coordinate of the event

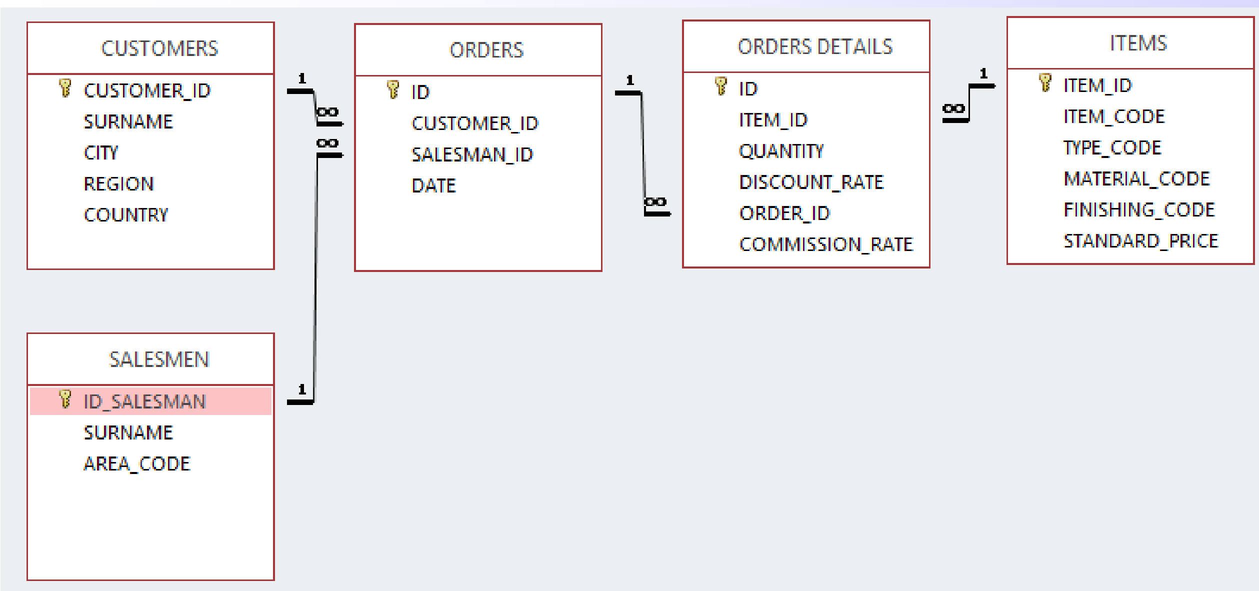
Yearly Income	Years							
Country	2010	2011	2012	2013	2014	2015	2016	Total Income
Italy	€ 4.046	€ 6.360	€ 6.882	€ 9.790	€ 7.232	€ 10.222	€ 8.369	**€ 52.901**
France	€ 8.012	€ 6.392	€ 7.126	€ 6.105	€ 6.481	€ 8.102	€ 11.755	**€ 53.973**
UK	€ 5.419	€ 5.458	€ 6.470	€ 3.901	€ 4.935	€ 7.656	€ 5.671	**€ 39.510**
Spain	€ 5.505	€ 5.680	€ 5.671	€ 5.445	€ 7.448	€ 7.096	€ 6.118	**€ 42.963**
Austria	€ 1.545	€ 505	€ 770	€ 1.829	€ 2.332	€ 2.379	€ 2.672	**€ 12.032**
Germany	€ 5.650	€ 2.057	€ 2.351	€ 1.491	€ 2.394	€ 2.977	€ 2.474	**€ 19.394**
Switerland	€ 322	€ 2.565	€ 2.114	€ 1.132	€ 1.543	€ 1.660	€ 3.159	**€ 12.495**

Pivot Table are a good example. In this case:

- **The matrix represents Sales**
- **Income is the way in which sales are quantified**
- **Country and Years are the dimensions**
- **Adding filtering fields would add dimensions**

Multi Dimensional Model

- Let us consider an example – The Meta_Data (i.e., Tables) needed to records customers' orders
- The **Relational DB is organized according to the ordering process**
  - The Customer and the Salesman are the "actors" of this process.
  - So we need the **customers' and the salesmen registry (master data)**
  - To complete an order there is the need to access to the product list and to "explode it" into its lines (each line correspond to a specific item included in the order).
  - The **ORDERS, ORDERS_DETAIL and ITEMS** Tables are used to this scope.



- So, to comply with the structure of the process and due to normalization rules, data concerning a single sale is spread over four distinct Tables
- In other words the <u>information is fragmented</u>.
- How can we recombine it to make it usable?

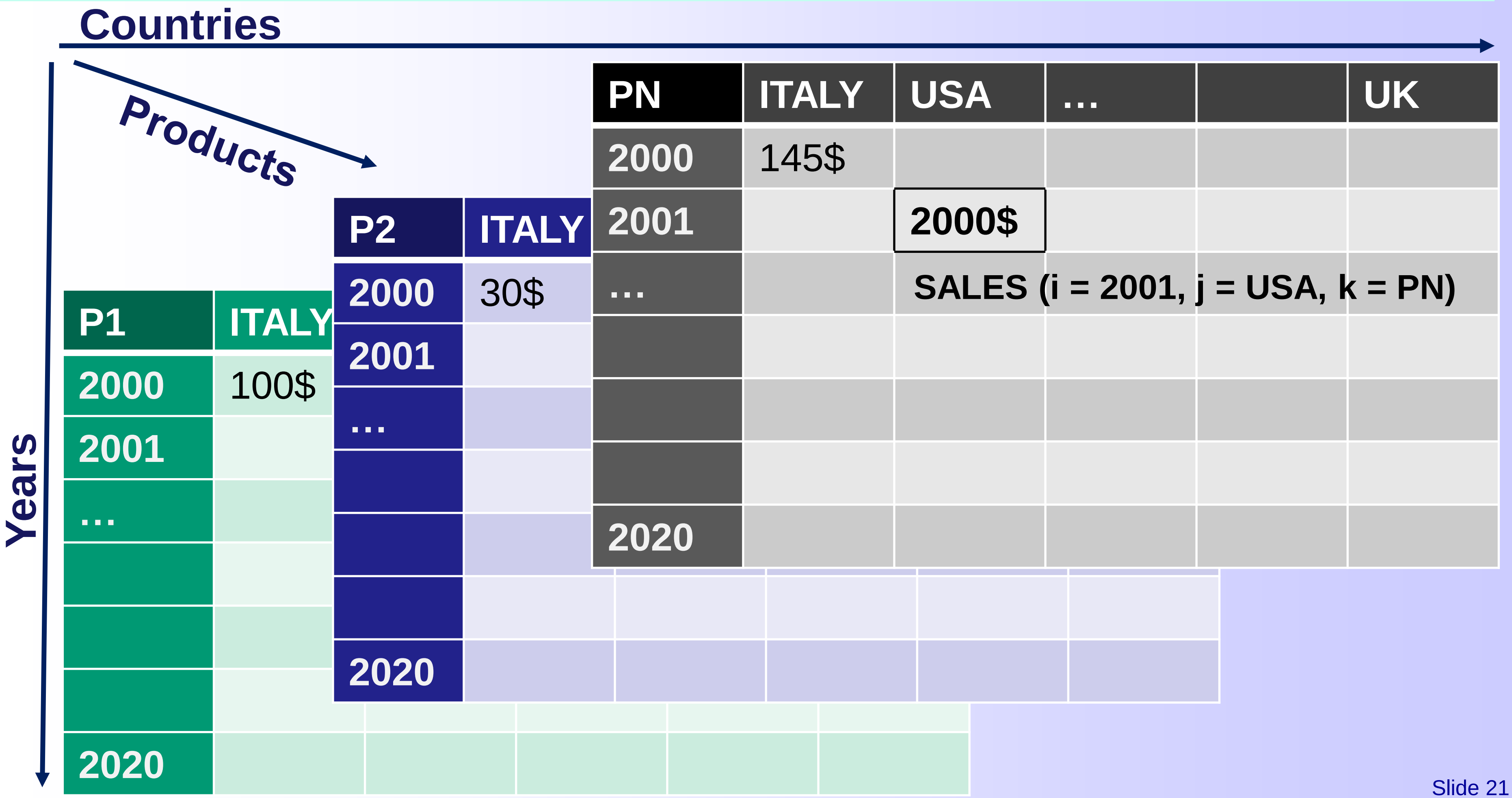
Multi Dimensional Model

- We want to get information about **sales** i.e., our fact is a sale event, that is each element of the sensor will contain information about a specific sale

- But how? We need to decide the way/metrics needed to quantify the sale. For example: Total Value, or Total Quantity should fit.
- Is this sufficient? No, we also need to define the dimensions of our fact.
- For example we could be interested in:
  - *Time*
  - *Product*
  - *Customer*
  - *Salesman*
  - *Country*
- In this way we could answer to the following queries:
  - *How much are the total sales in a specific year?*
  - *And for a specific customer?*
  - *And in a specific country?*

Multi Dimensional Model

- For instance with a single metric (say Total Value) and three dimensions (say Time, Country and Product), the tensors would be as in the following example



Multi Dimensional Model

- We need to **create a View** (i.e., a macro and redundant table) containing all the data that we need
- Specifically all selected dimensions and metrics will be the fields of the view
- Once we have this view we can navigate through the data as we prefer

- A simple, but long join query, is therefore needed

```
SELECT ORDERS.ID, CUSTOMERS.*, SALESMEN.*, ORDERS.DATE, _
   [ORDERS DETAILS].ITEM_ID, ITEMS.ITEM_CODE, [ORDERS DETAILS].QUANTITY,_
   ITEMS.STANDARD_PRICE, [ORDERS DETAILS].DISCOUNT_RATE,  _
   [ORDERS DETAILS].[QUANTITY]*[ITEMS].[STANDARD_PRICE]*_
   (1-[ORDERS DETAILS].[DISCOUNT_RATE]) AS TOTAL, _
   [ORDERS DETAILS].[QUANTITY]*[ITEMS].[STANDARD_PRICE]* _
   (1-[ORDERS DETAILS].[DISCOUNT_RATE])* _
   [ORDERS DETAILS].[COMMISSION_RATE] AS Commission
FROM (CUSTOMERS INNER JOIN (SALESMEN INNER JOIN ORDERS ON _
   SALESMEN.[ID_SALESMAN] = ORDERS.[SALESMAN_ID]) ON CUSTOMERS.[CUSTOMER_ID] _
   = ORDERS.[CUSTOMER_ID]) INNER JOIN (ITEMS INNER JOIN [ORDERS DETAILS] ON _
   ITEMS.[ITEM_ID] = [ORDERS DETAILS].[ITEM_ID]) ON ORDERS.ID = _
   [ORDERS DETAILS].[ORDER_ID]
ORDER BY ORDERS.DATE
```

Multi Dimensional Model

An observation

- In the query above we did not use any group operating function. So we did not make any data aggregation.
- In this case the query returns as many records (say R) are the ones included in the ORDER DETAILS Table. Each one is an event, i.e., an element of the tensor.
- The granularity of the Data Warehouse is the same as the original DB
- Also, if Y, C and P are the number of years, countries and products, respectively, the total number of possible events is Y x C x P. So R, out of a total of Y x C x P possible event will be quantified, the other remain empty
- The tensor is “sparse” (non densely populated)
- Clearly, we should have performed some kind of data aggregation
- For instance, if we are not interested in daily sales, we could have summed sales per month, or even for year.
- Alternatively we could have summed sales per country, etc.
- This is absolutely licit and very common. In this case the granularity of the Data Warehouse differs from that of the original Data Base

Multi Dimensional Model

- We need to create a View (i.e., a macro and redundant table) containing all the data that we need
- Once we have this view we can navigate through the data as we prefer
- To this aim a JOIN QUERY is needed
- The query is easy, but very long
- The obtained View is shown below

ID	CUST	CUSTOM	CITY	REGION	COUNTRY	ID_SA	SALESM	AREA_C	DATE	ITEM_	ITEM_CC	QUAN	STANDAR	DISCO	Total	Commission
1	1	Carlozzi	Roma	Lazio	Italia	1	Bianchi	Area 1	01/11/2017	1	A_1	10	100,00 €	0	1000	100
1	1	Carlozzi	Roma	Lazio	Italia	1	Bianchi	Area 1	01/11/2017	2	A_2	5	500,00 €	0,1	2250	225
2	1	Carlozzi	Roma	Lazio	Italia	2	Rossi	Area 2	02/11/2017	1	A_1	10	100,00 €	0,1	900	90
2	1	Carlozzi	Roma	Lazio	Italia	2	Rossi	Area 2	02/11/2017	2	A_2	5	500,00 €	0	2500	250
3	1	Carlozzi	Roma	Lazio	Italia	1	Bianchi	Area 1	03/11/2017	3	A_3	10	1.000,00 €	0	10000	1000
3	1	Carlozzi	Roma	Lazio	Italia	1	Bianchi	Area 1	03/11/2017	4	A_3	5	200,00 €	0	1000	100
4	1	Carlozzi	Roma	Lazio	Italia	2	Rossi	Area 2	04/11/2017	5	B_1	10	300,00 €	0	3000	450
5	1	Carlozzi	Roma	Lazio	Italia	3	Verdi	Area 1	05/11/2017	1	A_1	5	100,00 €	0,2	400	40
6	1	Carlozzi	Roma	Lazio	Italia	2	Rossi	Area 2	06/11/2017	2	A_2	1	500,00 €	0	500	50
7	2	Mengoni	Perugia	Umbria	Italia	2	Rossi	Area 2	07/11/2017	3	A_3	2	1.000,00 €	0	2000	200
8	3	Menozzi	Parma	Emilia Rom	Italia	3	Verdi	Area 1	08/11/2017	4	A_3	3	200,00 €	0	600	90
9	3	Menozzi	Parma	Emilia Rom	Italia	3	Verdi	Area 1	09/11/2017	5	B_1	4	300,00 €	0	1200	120
9	3	Menozzi	Parma	Emilia Rom	Italia	3	Verdi	Area 1	09/11/2017	5	B_1	5	300,00 €	0	1500	150
10	4	Caldaveri	Firenze	Toscana	Italia	3	Verdi	Area 1	10/11/2017	4	A_3	6	200,00 €	0,3	840	84
11	5	Bixel	London	Central	UK	3	Verdi	Area 1	11/11/2017	3	A_3	7	1.000,00 €	0	7000	700
12	5	Bixel	London	Central	UK	4	Bruni	Area 2	12/11/2017	2	A_2	8	500,00 €	0	4000	600
13	5	Bixel	London	Central	UK	4	Bruni	Area 2	13/11/2017	1	A_1	9	100,00 €	0	900	135
13	5	Bixel	London	Central	UK	4	Bruni	Area 2	13/11/2017	2	A_2	10	500,00 €	0,4	3000	450

Multi Dimensional Model

ID	CUST	CUSTOM	CITY	REGION	COUNTRY	ID_SA	SALESM	AREA_C	DATE	ITEM_	ITEM_CC	QUAN	STANDAR	DISCO	Total	Commission
1	1	Carlozzi	Roma	Lazio	Italia	1	Bianchi	Area 1	01/11/2017	1	A_1	10	100,00 €	0	1000	100
1	1	Carlozzi	Roma	Lazio	Italia	1	Bianchi	Area 1	01/11/2017	2	A_2	5	500,00 €	0,1	2250	225
2	1	Carlozzi	Roma	Lazio	Italia	2	Rossi	Area 2	02/11/2017	1	A_1	10	100,00 €	0,1	900	90
2	1	Carlozzi	Roma	Lazio	Italia	2	Rossi	Area 2	02/11/2017	2	A_2	5	500,00 €	0	2500	250
3	1	Carlozzi	Roma	Lazio	Italia	1	Bianchi	Area 1	03/11/2017	3	A_3	10	1.000,00 €	0	10000	1000
3	1	Carlozzi	Roma	Lazio	Italia	1	Bianchi	Area 1	03/11/2017	4	A_3	5	200,00 €	0	1000	100
4	1	Carlozzi	Roma	Lazio	Italia	2	Rossi	Area 2	04/11/2017	5	B_1	10	300,00 €	0	3000	450
5	1	Carlozzi	Roma	Lazio	Italia	3	Verdi	Area 1	05/11/2017	1	A_1	5	100,00 €	0,2	400	40
6	1	Carlozzi	Roma	Lazio	Italia	2	Rossi	Area 2	06/11/2017	2	A_2	1	500,00 €	0	500	50
7	2	Mengoni	Perugia	Umbria	Italia	2	Rossi	Area 2	07/11/2017	3	A_3	2	1.000,00 €	0	2000	200
8	3	Menozzi	Parma	Emilia Rom	Italia	3	Verdi	Area 1	08/11/2017	4	A_3	3	200,00 €	0	600	90
9	3	Menozzi	Parma	Emilia Rom	Italia	3	Verdi	Area 1	09/11/2017	5	B_1	4	300,00 €	0	1200	120
9	3	Menozzi	Parma	Emilia Rom	Italia	3	Verdi	Area 1	09/11/2017	5	B_1	5	300,00 €	0	1500	150
10	4	Caldaveri	Firenze	Toscana	Italia	3	Verdi	Area 1	10/11/2017	4	A_3	6	200,00 €	0,3	840	84
11	5	Bixel	London	Central	UK	3	Verdi	Area 1	11/11/2017	3	A_3	7	1.000,00 €	0	7000	700
12	5	Bixel	London	Central	UK	4	Bruni	Area 2	12/11/2017	2	A_2	8	500,00 €	0	4000	600
13	5	Bixel	London	Central	UK	4	Bruni	Area 2	13/11/2017	1	A_1	9	100,00 €	0	900	135
13	5	Bixel	London	Central	UK	4	Bruni	Area 2	13/11/2017	2	A_2	10	500,00 €	0,4	3000	450

- Elements of the matrix (i.e. the facts) are the sales quantified in terms of units sold
- Dimensions are: (i) Country and (ii) Product Type and (iii) Salesman
- Note that the third dimension is shown as a filtering condition
- The geographic dimensions has a hierarchy, that is Country → Region → City

Data can be loaded on a Spread Sheet to create a Pivot Table

Cognome Agente (più elementi)

Somma di QUANTITA	Etichette di colonna			
Etichette di riga	**A_1**	**A_2**	**A_3**	**Totale complessivo**
⊟ **Italia**	**10**	**5**	**15**	**30**
⊟ **Lazio**	**10**	**5**	**15**	**30**
Roma	10	5	15	30
⊞ **UK**	**9**	**18**		**27**
Totale complessivo	**19**	**23**	**15**	**57**

Multi Dimensional Model

- Suppose we want to use **years** and **countries** as dimensions, and **sale quantities** as metric.
- In Access we can use the ***"Cross Table Format"*** to create a 2Dim Matrix (the Pivot query option is no longer available)

- SELECT Query, made on the View that we have obtained before;

SELECT Country, Year(Data) _
SUM(Quantity**) AS** [TOT Q],
FROM VIEW
GROUP BY Country, Year(Data)

Country	2016	2017	2018
ITALY	3	19	3
UK			5
USA	4	6	2

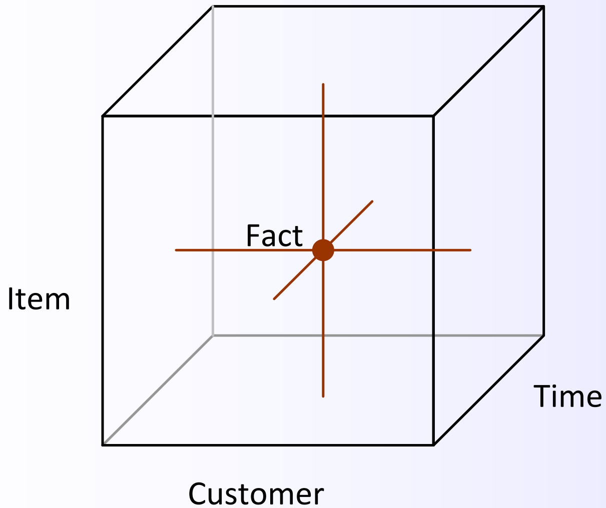
The query in crosstab format

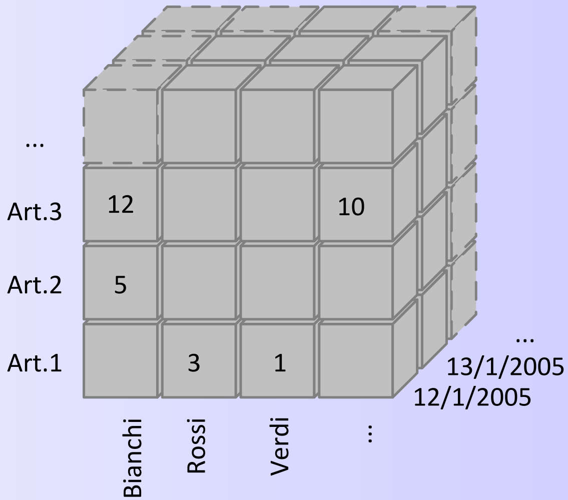
Year	Country	Tot_Q
2016	ITALY	3
2016	USA	4
2017	ITALY	19
2017	USA	6
2018	ITALY	3
2018	UK	5
2018	USA	2

The query in standard Table format

Multi Dimensional Model

- Information is coded using multi dimensional matrices
- **Hyper Cube: graphical representation of a multi dimensional matrix representing a certain event**
  - **Granular Fact** - An element at the intersection of the coordinates
  - **Measure** - A value that quantifies a fact
  - **Dimension** - The value of one coordinate of a fact





Fact – Measures and Aggregability

- **How to identify a fact?**
  - *Fact (Dimension 1, ..., Dimension N)*
  - *Measure (Dimension 1, ..., Dimension N).<Measure Quantifier>*
- If **all dimensions are quantified** (at their maximum granularity level) what we get is **an elementary fact**
- **Aggregated events** can also be obtained
  - *If some dimensions are not considered (*,*,Dimension,*,*)*
  - *By aggregating data (drilling up along a dimensions) using specific grouping operators: Sum, Average, Max, Min, ….*
- **It is always possible to aggregate data along a dimension?**
- **Grouping operators can be used on every dimensions?**

Aggregability

ITEM		Warehouse	Date	On Hand
PP1007015	Polystyrene Panel 100x70x1.5	Raw Material	13/02/05	100
PP1007015	Polystyrene Panel 100x70x1.5	Acceptance	13/02/05	20
VA1010	Iron Screw 10mmx1	Raw Material	13/02/05	24002
	…			
PP1007015	Polystyrene Panel 100x70x1,5	Raw Material	14/02/05	110
PP1007015	Polystyrene Panel 100x70x1,5	Acceptance	14/02/05	0
VA1010	Iron Screw 10mmx1	Raw Material	14/02/05	23870
	…			

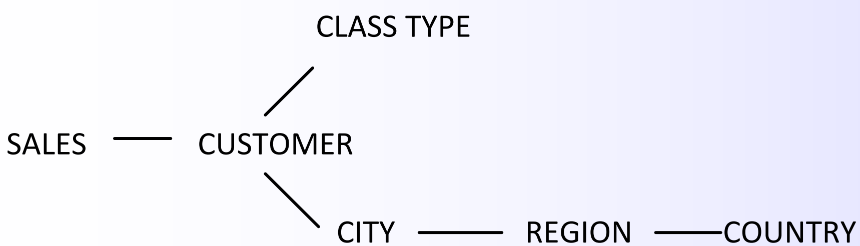
- (PP1007015,Raw Material,13/02/2005).OH = 100 ⋎
- (PP1007015,*,13/02/2005).OH = 100 + 20 = 120 ⋎
- (PP1007015, Raw Material, *).OH = 100 + 110 = 201 ? ✕
- (PP1007015, Raw Material,*).OH = AVG(100 + 110) ⋎
- (*,Raw Material, 12/01/2005).OH = 100 + 24002 ? ✕
- (*,Raw Material, 12/01/2005).$ this would be fine!!! ⋎

Additivity (i.e., Aggregation through Sum)

- **Level Measures are never additive relative to time**
  - A specific property of a fact depending, directly, on the time in which the fact has occurred (On Hand, Number of Orders, Etc.)
- **Unitary Measures are never additive**
  - A property of the subject to which the events belongs to and that depends, directly, on the time in which the fact has occurred (purchase price, discount rate, interest rate, etc.)
- **Flux Measures are always additive (∀ dimension)**
  - A property of a fact observed over a certain period, taken as reference (number of units sold, income, number of complaints, etc.)

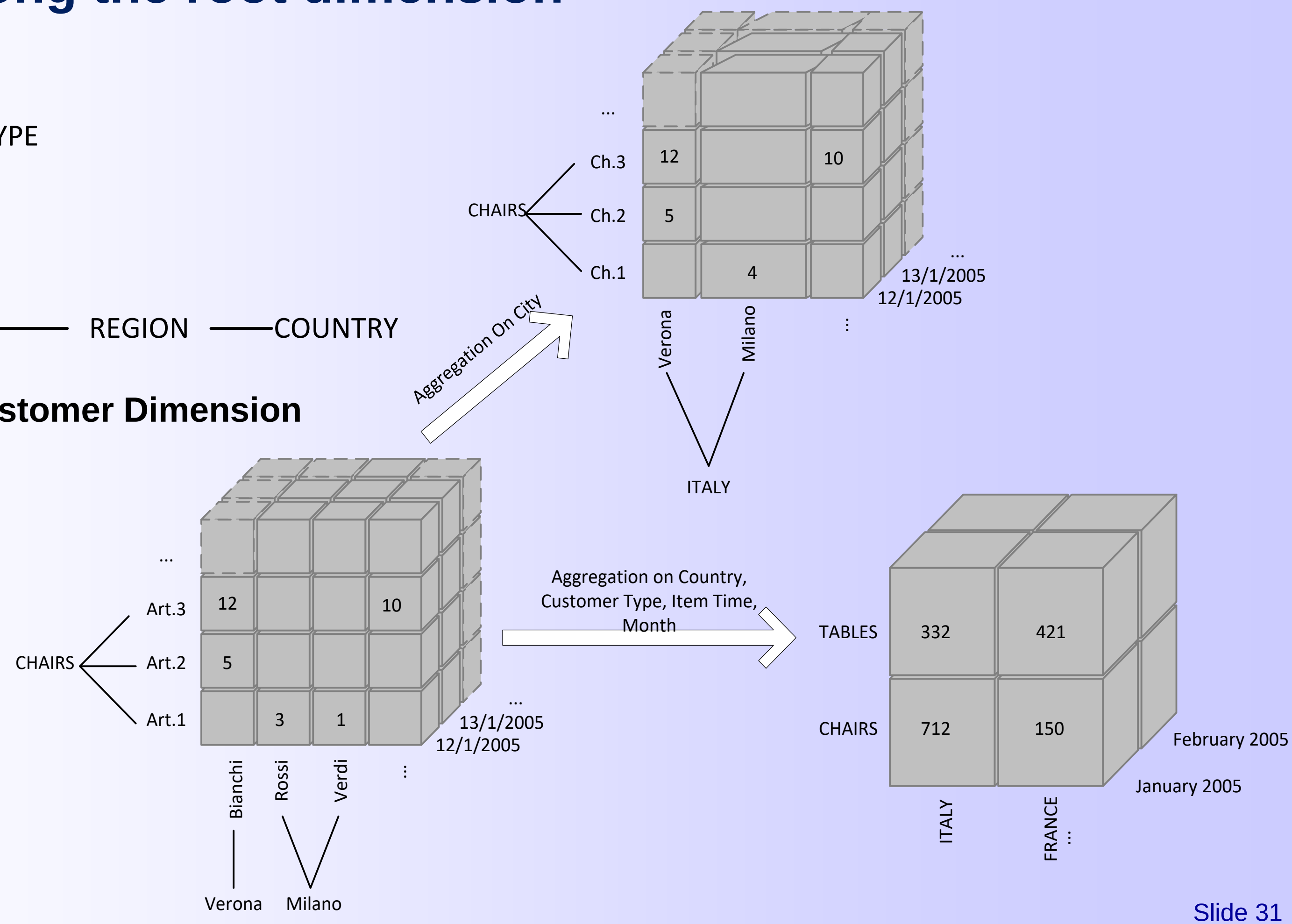
Hierarchies

- Dimensions may be the root node of a hierarchy
- Each node corresponds to a specific aggregation, made (drilling up) along the root dimension



2 Hierarchies on the Customer Dimension

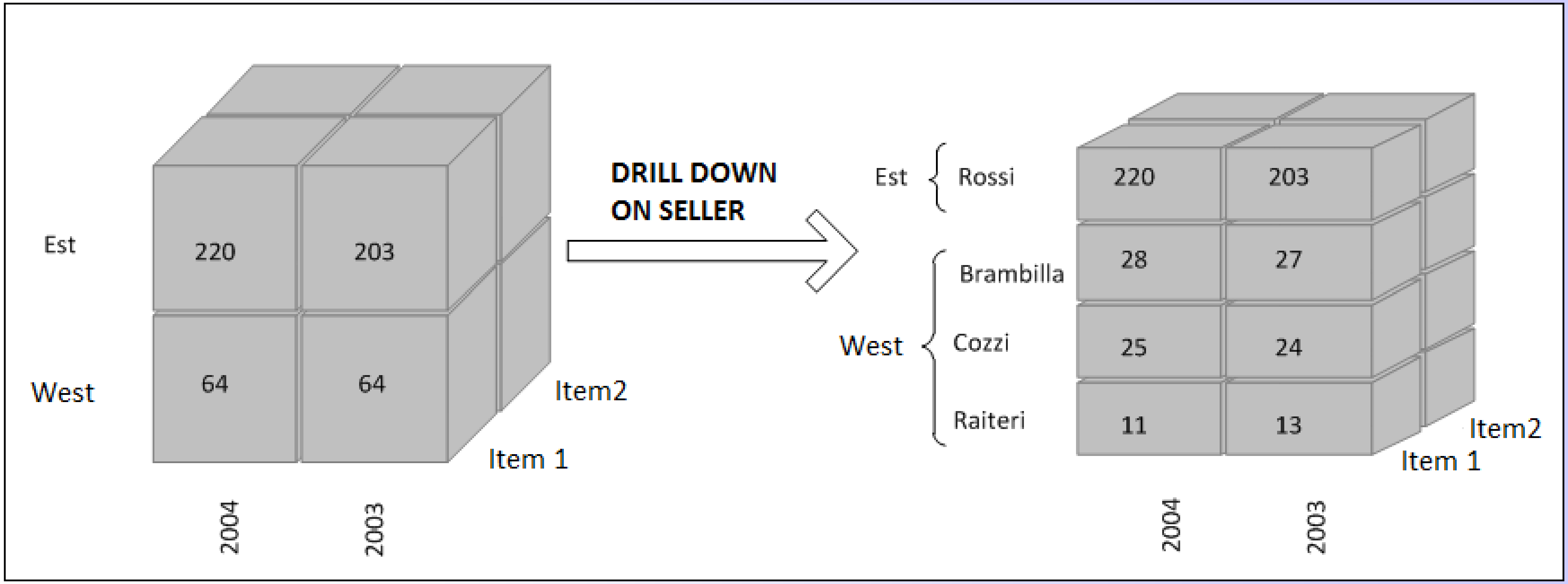
Some examples of aggregation



OLAP OPERATORS

Drill-Down

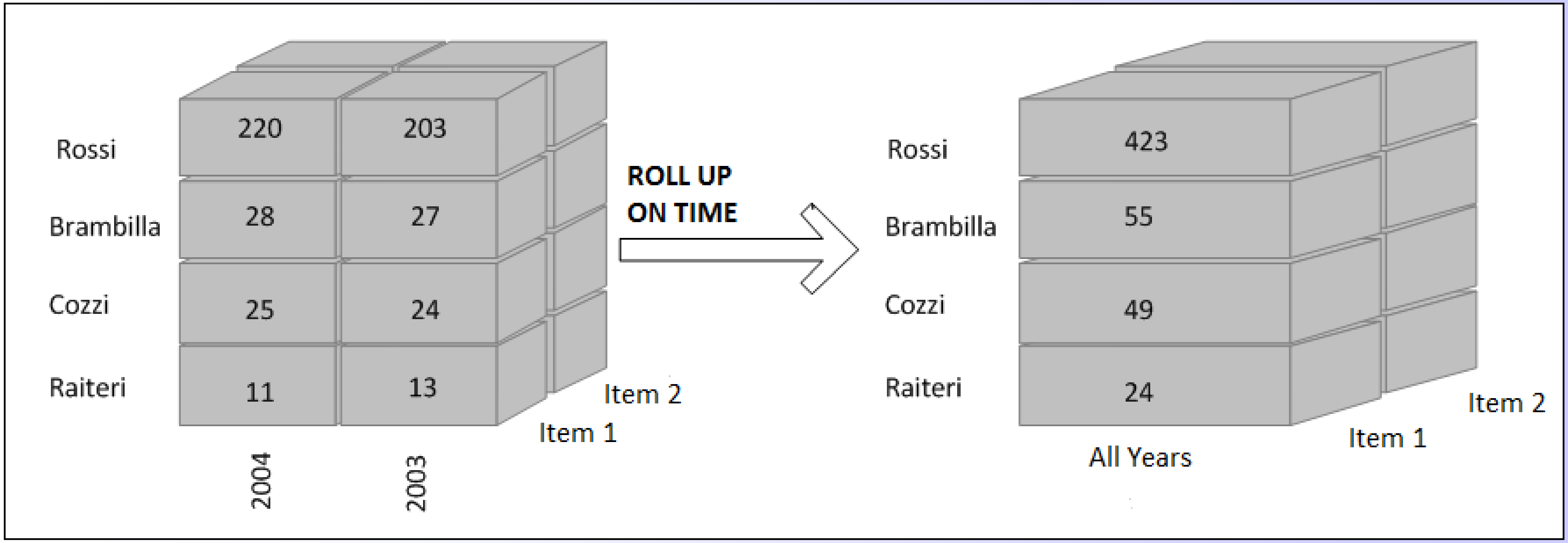
– The level of details is increased
  - Moving down along a hierarchy
  - Adding new dimensions



Operatori OLAP

Roll-Up

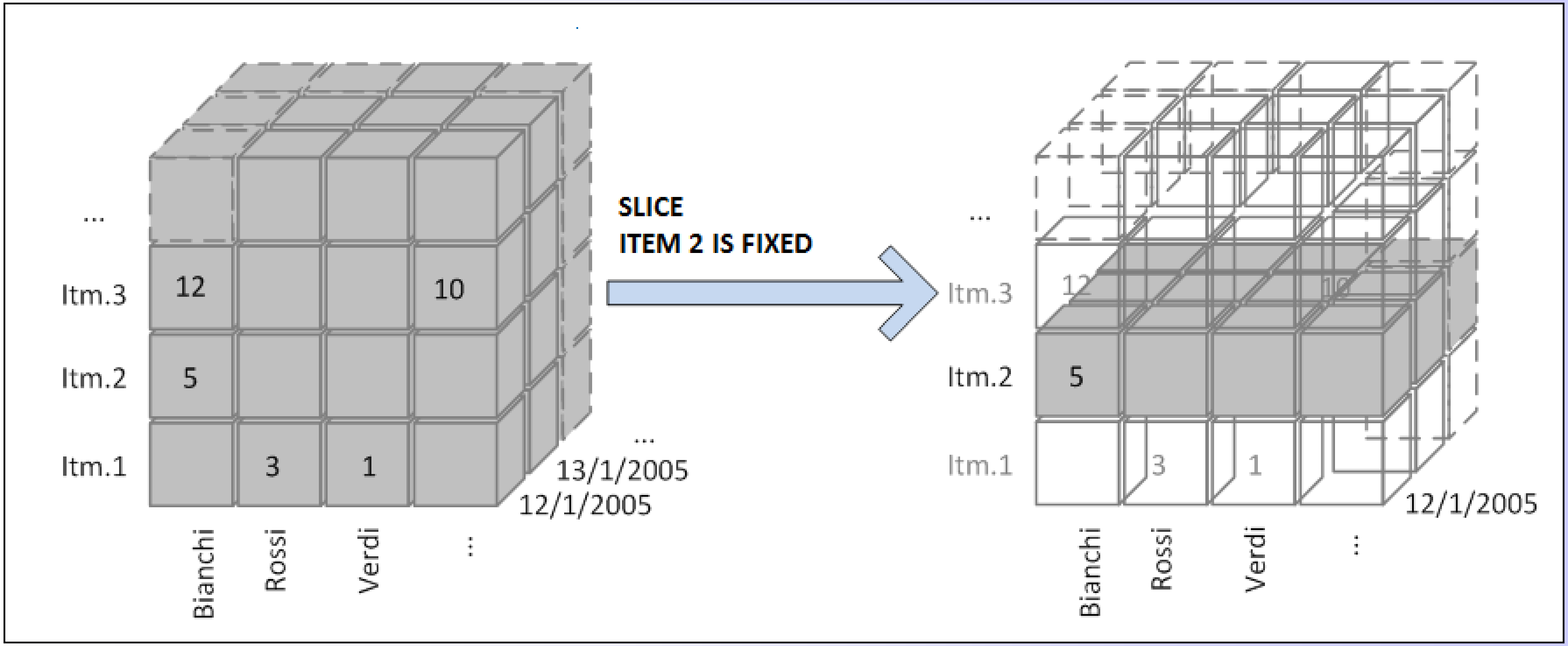
– Data are aggregated:
  - Going up along a hierarchy
  - Removing one dimension



Operatori OLAP

Slice

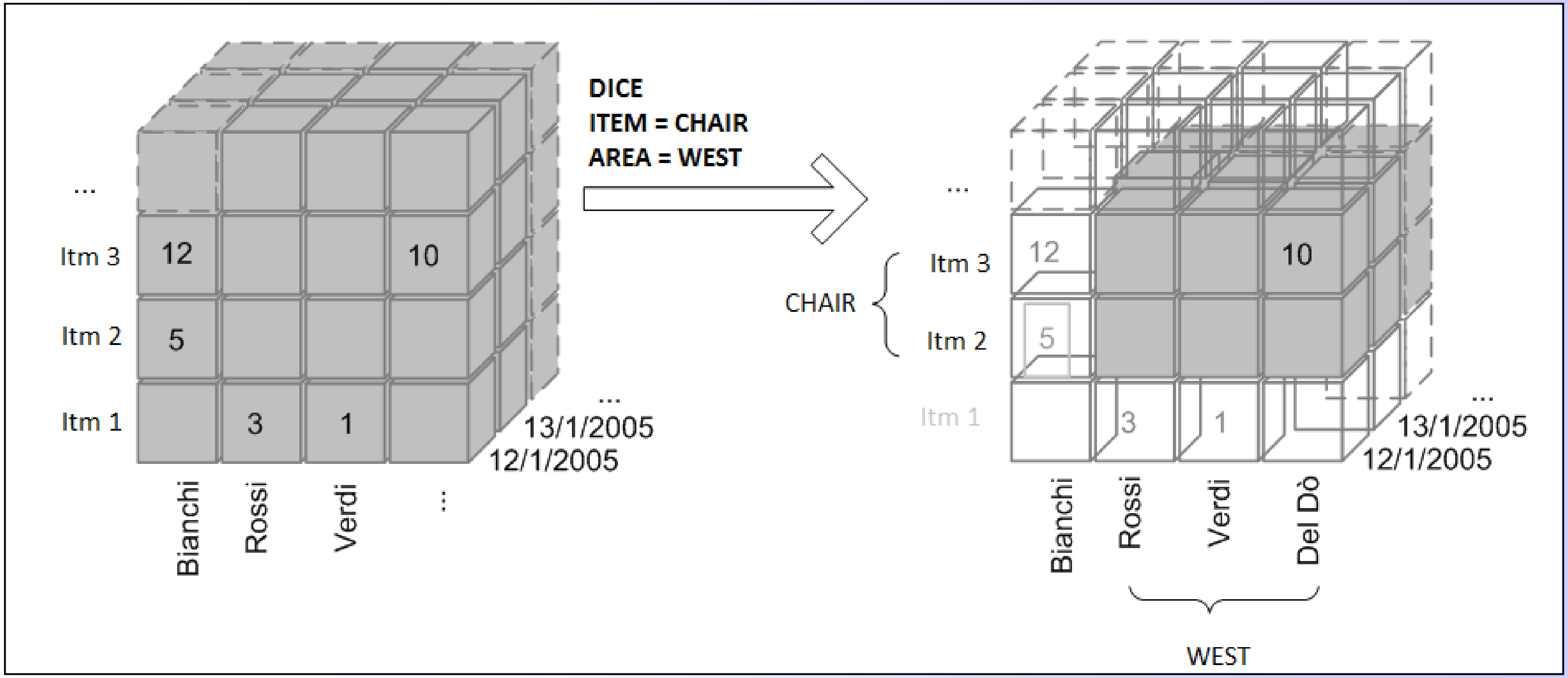
- The value of one dimension is fixed
- The portion of (filtered) data obtained in this way will be analyzed next



Operatori OLAP

Dice

- Data are filtered
- Two or more dimensions are fixed at a certain level



Operatori OLAP

Pivot

- Relationships, in terms of dimensions, are reversed
- A rotation of the cube is made

ITEM	Area	2013	2014
Item #1	Center	60	56
	Est	203	220
	West	64	64

ITEM	Year	Center	Est	West
Articolo #1	2013	60	203	64
	2014	56	220	64

Dimensional Fact Model

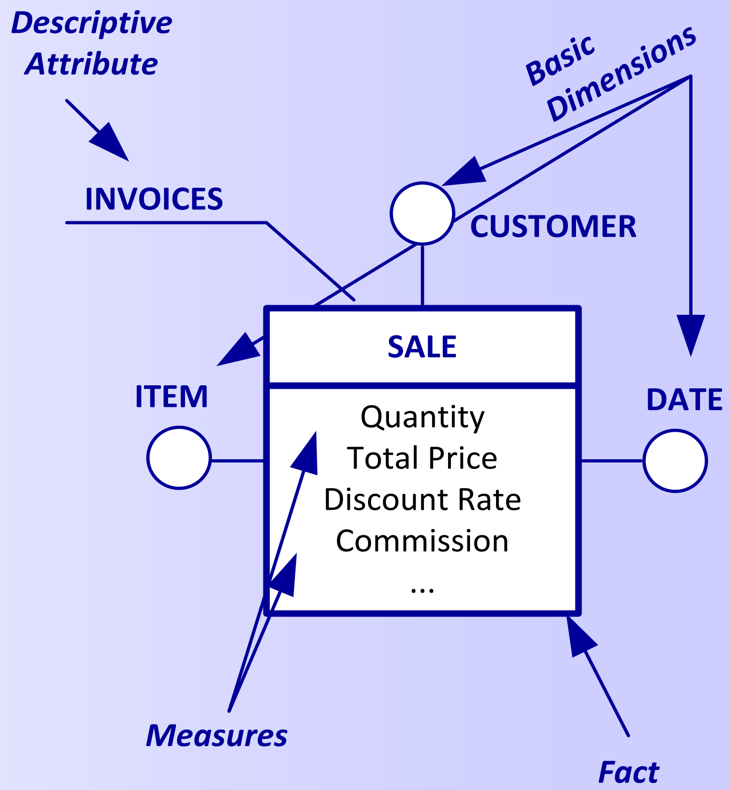
- Dimensional Fact Model is a graphical way to depict the facts around which the warehouse is structured
- Each Fact is represented through a **«Fact Scheme»**

Fact

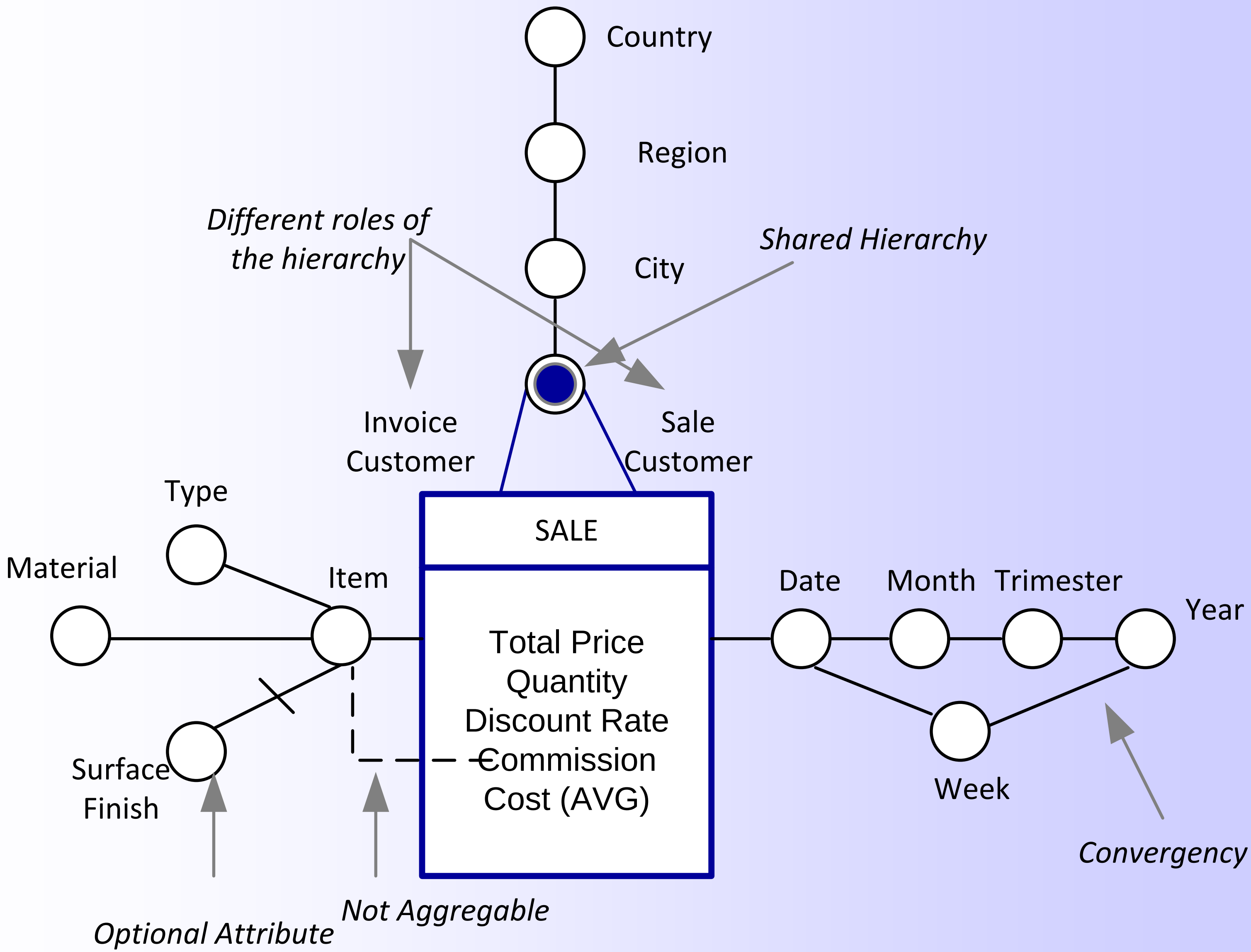
- Represented with a **rectangle** containing the **name and the measures** of the fact

Basic Dimensions

- Represented with **circles** connected to the fact



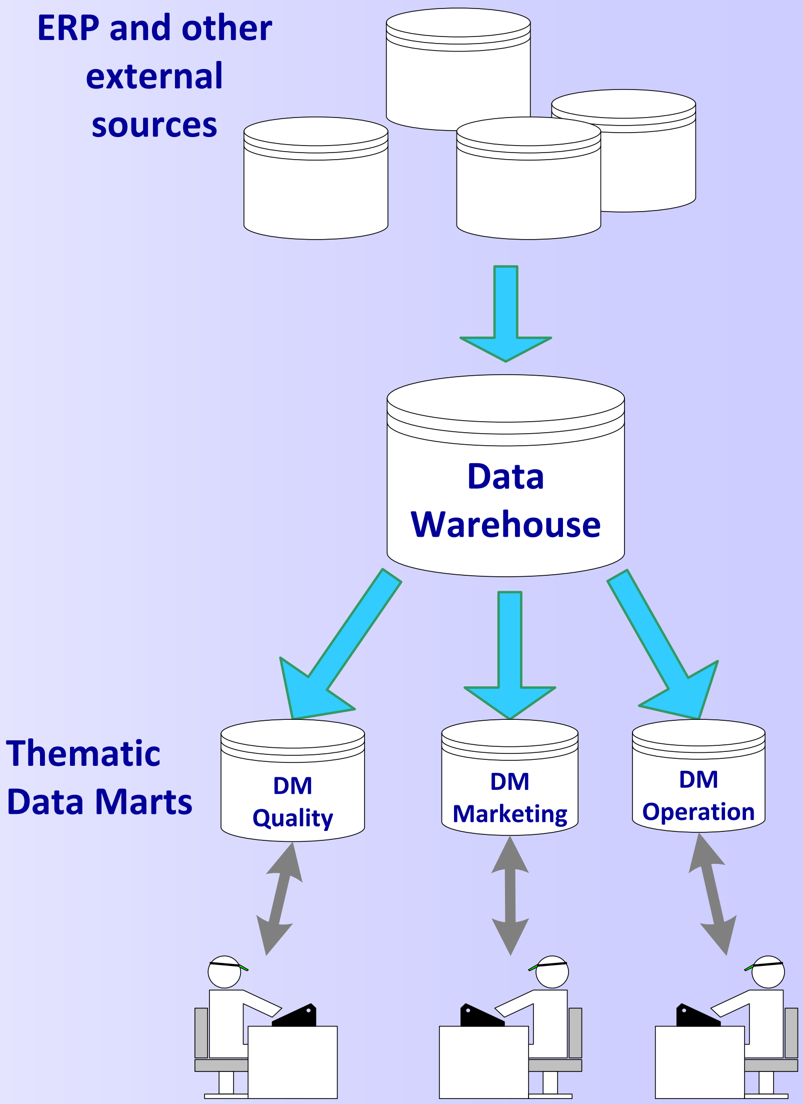
Dimensional Fact Model



Data Warehouse and Data Mart

A Data Mart is a sort of «Thematic Warehouse»

- It contains only the facts that are relevant for a certain area of research (or for a specific business function)
- Data pertain a limited temporal extension
- Data granularity is lower



LOGICAL MODELS (I)

- **ROLAP** – MOLAP – HOLAP
- **ROLAP → RELATIONAL OLAP**
  - A non normalized RDB is used to mimic a multi dimensional matrix
  - Queries are based on standard SQL

PRO

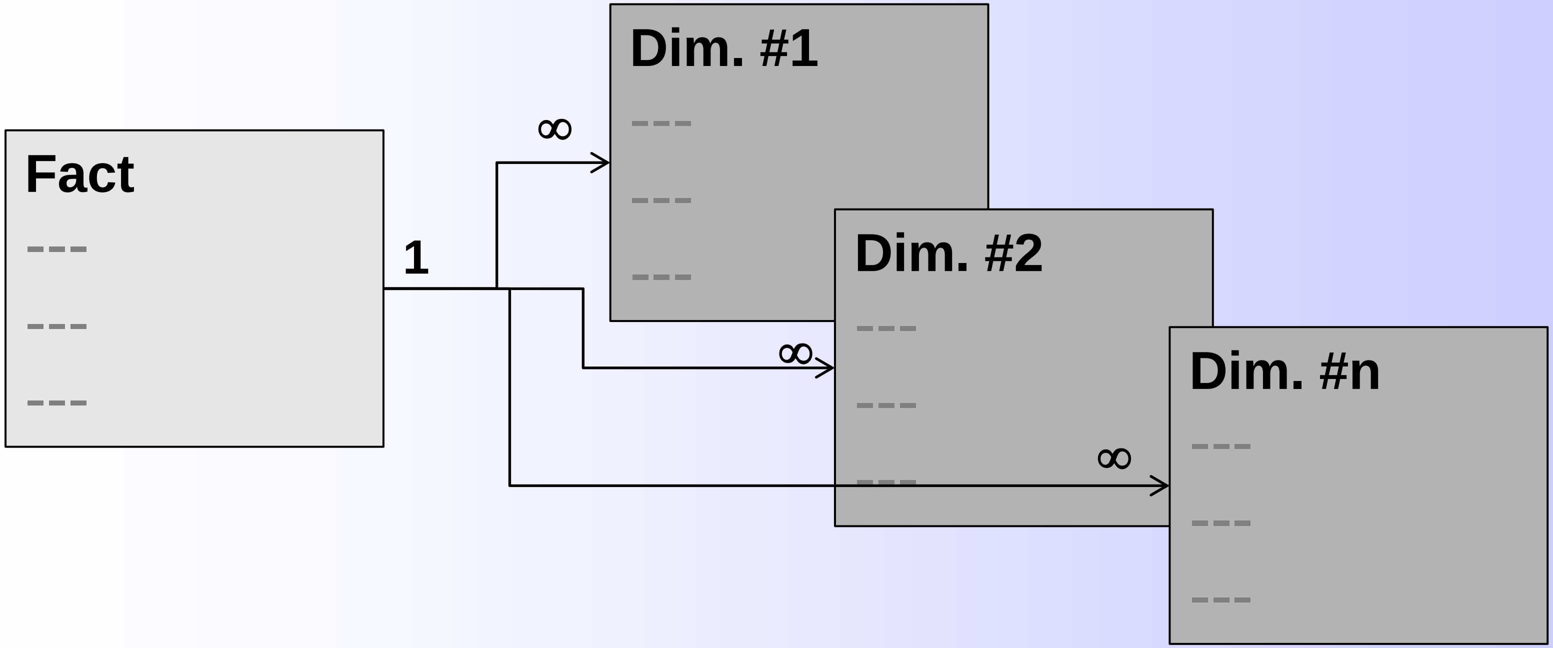
- ✓ **Optimal memory usage i.e., Matrix's sparsity does not occur**
- ✓ **Data can be retrieved using simple joint queries**
- ✓ **Skills concerning RDB are widespread**

CONS

- × **Queries' execution is not efficient, due to functions operating on groups**
- × **Data redundancy, due to non normalized RDB**
- × **Inclusion of pre-saved Views is needed to speed computation time**

STAR SCHEMA

– Most of the times ROLAP is based on a **«Star Schema»**
– Facts' multi-dimensional structure is realized using a RDB
– Facts and Dimensions are obtained using **Tables with OTM relationships**
– Tables are **not normalized**



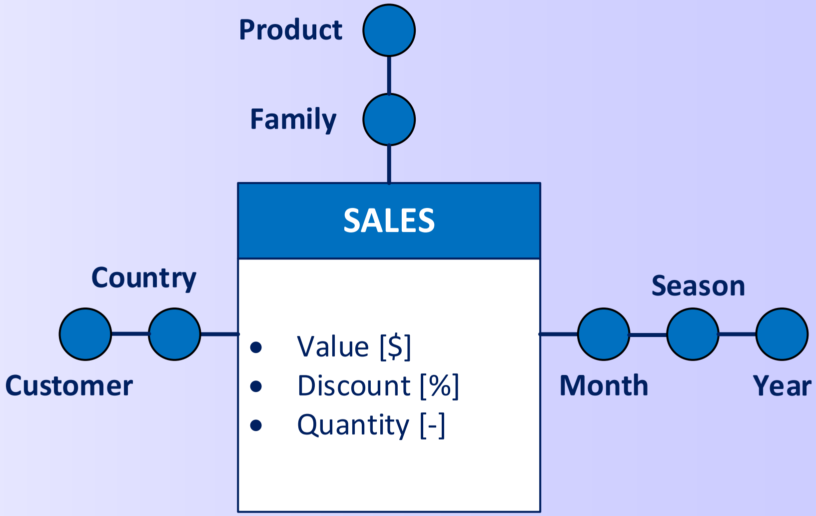
STAR SCHEMA

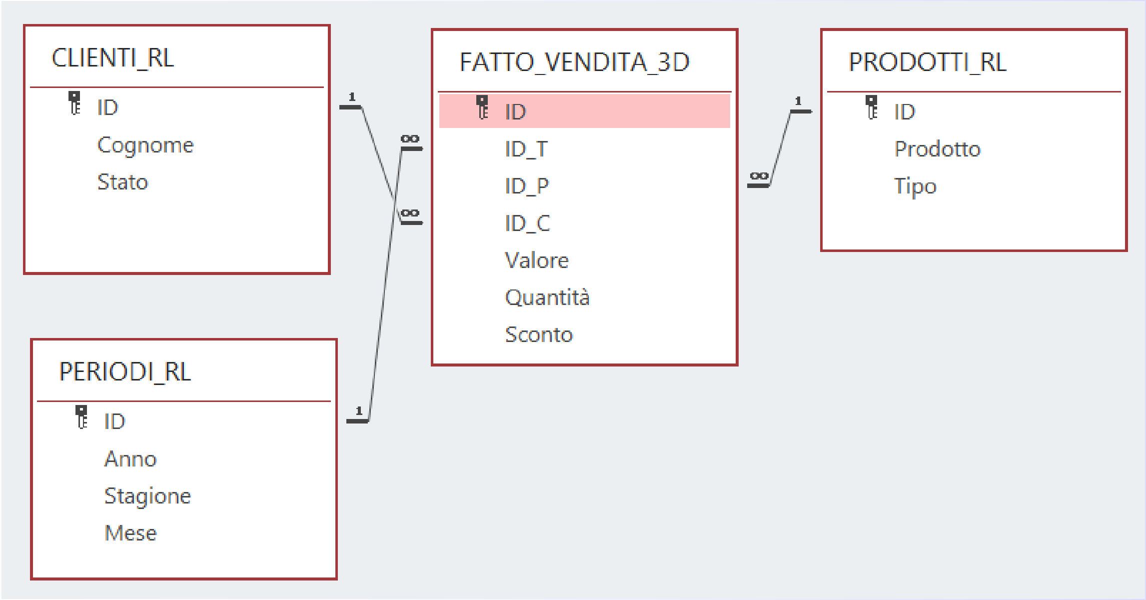
- **FACTS' TABLES**
  - One Table for each Fact
  - One Field for each Measure
  - One FK for each Basic Dimension

- **DIMENSIONS' TABLES**
  - One Table for each Basic Dimension
  - One Field for each Attribute of the Hierarchy (having, as root the current dimension)
  - Not Normalized
    - Ignores redundancy concerning hierarchies
    - Does not take advantage of shared hierarchies

STAR SCHEMA - Example

- Let us consider the following DFM
- The corresponding ROLAP DW, with star schema, is shown next





Note that, in a certain way, <u>the FKs</u> of the Fact Table <u>represent the codification of the three dimensions</u> of the 3D Tensor the ROLAP structure is related to

STAR SCHEMA – DIMENSIONS' TABLES

ID	Surname	Country
1	AAAAA	ITALY
2	BBBBB	ITALY
3	CCCCC	USA
4	DDDDD	USA
5	EEEEE	UK

ID	Year	Season	Month
1	2016	Summer	7
2	2016	Winter	12
3	2017	Spring	5
4	2017	Summer	6
5	2017	Summer	7
…	…	…	…

ID	Product	Type
1	P1	RM
2	P2	RM
3	P3	SF
4	P4	SF
5	P5	EP
…	…	…

- Dimensions' Tables create a mapping between coordinates and their label
- Table is the dimensions
- ID is the coordinate
- The other fields are the label associated to a certain coordinate
- For instance CUSTOMERS has 5 possible coordinates, the first one correspond to the label [AAAAA; ITALY]

- Example
  - ✓ (2,3,4) corresponds to
  - ✓ {[BBBBB, ITALY]; [2017, SPRING, 5]; [P4, SF]}

STAR SCHEMA – FACT TABLES

ID	ID_Time	ID_Prod	ID_Cust	Value	Quantity	Discount
1	1	2	3	10.00 €	1	0.00%
2	2	2	4	20.00 €	2	0.00%
3	4	3	3	17.00 €	2	15.00%
4	1	1	1	36.00 €	2	10.00%
5	2	3	6	106.50 €	4	7.50%
6	3	3	2	176.80 €	5	11.50%
7	1	4	4	47.50 €	5	5.00%
8	2	3	3	25.50 €	1	15.00%
….						

- Each record is a fact of the Data Warehouse
- ID is just the PK, it does not have any physical meaning
- The FKs are the coordinates, they define the location of a record (the fact) in the tensors
- The other fields are the metrics that quantifies the fact
- For example the 4-th record is at coordinates (1,1,1) so it corresponds to the sales of product P1 made in Italy (customer AAAA) in the month of July

HOW TO POPULATE THE DIM. TABLES?

- Populating the DIMENSIONS TABLE is rather easy
- Let be $<h_1,h_2,\ldots,h_n>$ the tuple defining the fields of the starting View that correspond to the hierarchy of a Dimensions
- In our example one of this tuple is <Year, Season, Month>
- What we have to do is to select, in the original View, all the tuples $<h_1,h_2,\ldots,h_n>$, that are distinct
- Each one of them is a specific coordinates of our tensor

SURNAME	COUNTRY	PRODUCT	TYPE	YEAR	SEASON	MONTH	QUANTITY	DISCOUNT	VALUE
DDDDD	USA	P1	RM	**2016**	**Summer**	**7**	1	0	10,00 €
BBBBB	ITALY	P3	SF	2016	Summer	7	1	0.15	25,50 €
BBBBB	ITALY	P4	SF	**2016**	**Winter**	**12**	2	0	80,00 €
DDDDD	USA	P6	EP	2016	Summer	7	2	0	120,00 €
DDDDD	USA	P5	EP	2016	Summer	7	1	0.1	45,00 €
AAAAA	ITALY	P1	RM	**2017**	**Winter**	**1**	1	0	10,00 €

Only the underlined tuples must be included in the TIME Dimension Table

HOW TO POPULATE THE DIM. TABLES?

- Populating the DIMENSIONS TABLE is rather easy
- Let be $<h_1,h_2,\ldots,h_n>$ the tuple defining the fields of the starting View that correspond to the hierarchy of a Dimensions
- What we have to do is to select, in the original View, all the tuples $<h_1,h_2,\ldots,h_n>$, that are distinct
- Each one of them is a specific coordinates of our tensor
- So the INSERT INTO Query is straightforward

INSERT INTO TIME_TABLE **(**Year, Season, Month**)**

SELECT DISTINCT Year, Season, Month

FROM ORIGINAL_VIEW

HOW TO POPULATE THE FACT TABLE?

- Populating the FACT TABLE is a little bit trickier
- Let us consider the following scheme

ORIGINAL VIEW

SURNAME	COUNTRY	PRODUCT	TYPE	YEAR	SEASON	MONTH	QUANTITY	DISCOUNT	VALUE
DDDDD	USA	P1	RM	2016	Summer	7	1	0	10,00 €
BBBBB	ITALY	P3	SF	2016	Summer	7	1	0.15	25,50 €
BBBBB	ITALY	P4	SF	2016	Winter	12	2	0	80,00 €
DDDDD	USA	P6	EP	2016	Summer	7	2	0	120,00 €
DDDDD	USA	P5	EP	2016	Summer	7	1	0.1	45,00 €
AAAAA	ITALY	P1	RM	2017	Winter	1	1	0	10,00 €

FACT TABLE

- ID is an auto-calculated field
- The metrics are directly copied from the view to the table
- What about the FKs?

ID	ID_Cn	ID_Pr	ID_Tm	QUANTITY	DISCOUNT	VALUE
1	?	?	?	1	0	10,00 €
2	?	?	?	1	0.15	25,50 €
3	?	?	?	2	0	80,00 €
4	?	?	?	2	0	120,00 €
5	?	?	?	1	0.1	45,00 €
6	?	?	?	1	0	10,00 €

HOW TO POPULATE THE FACT TABLE?

- In the Original View we have the “Tuples”, in the Fact Table we have the coordinates
- So we have to go back through the mapping defined in the Dimensions’ Tables, moving in the opposite direction
- For example the first record of the Original View has the following mappings <DDDDD, USA>, <P1, RM>, <2016, Summer, 7> that correspond to dimensions (4, 1, 1)
- So we have:

ORIGINAL VIEW

SURNAME	COUNTRY	PRODUCT	TYPE	YEAR	SEASON	MONTH	QUANTITY	DISCOUNT	VALUE
DDDDD	**USA**	**P1**	**RM**	**2016**	**Summer**	**7**	1	0	10,00 €
…	….								

ID	ID_Cn	ID_Pr	ID_Tm	QUANTITY	DISCOUNT	VALUE
1	**4**	**1**	**1**	1	0	10,00 €
…						

STAR SCHEMA – DIMENSIONS’ TABLES

ID	Surname	Country
1	AAAAA	ITALY
2	BBBBB	ITALY
3	CCCCC	USA
4	**DDDDD**	**USA**
5	EEEEE	UK

ID	Year	Season	Month
1	**2016**	**Summer**	**7**
2	2016	Winter	12
3	2017	Spring	5
4	2017	Summer	6
5	2017	Summer	7
…	…	…	…

ID	Product	Type
1	**P1**	**RM**
2	P2	RM
3	P3	SF
4	P4	SF
5	P5	EP
…	…	…

- For example the first record of the Original View has thefollowing mappings:
  - ✓ *<DDDDD, USA>,*
  - ✓ *<P1, RM>,*
  - ✓ *<2016, Summer, 7>*
- That correspond to dimensions (4,1,1)

HOW TO POPULATE THE FACT TABLE?

- So the question is, how can we automatically perform the mapping using a query written in SQL?
- The answer is quite easy, indeed, for each record of the original view we want to:
  - ✓ Get the values corresponding to the metrics fields,
  - ✓ Couple these values with the value of the primary key of all the Dimensions’ Tables,
  - ✓ Make the association using, as selection criteria, the values in the fields (of the original view) corresponding to the “tuples” coded in the dimension table
- So we need to make a JOIN Query (Cartesina Product), linking the Original View with each one of the Dimensions’ Tables

HOW TO POPULATE THE FACT TABLE?

- So the question is, how can we automatically perform the mapping using a query written in SQL?
- We need to make a JOIN Query, linking the Original View with each one of the Dimensions' Tables

INSERT INTO FACT_TABLE **(**ID_Tm, ID_Pr, ID_Cs, Value, Quantity**)**

SELECT D_TIME.ID, D_PRODUCTS.Id, D_CUSTOMERS.ID, Value, Quantity

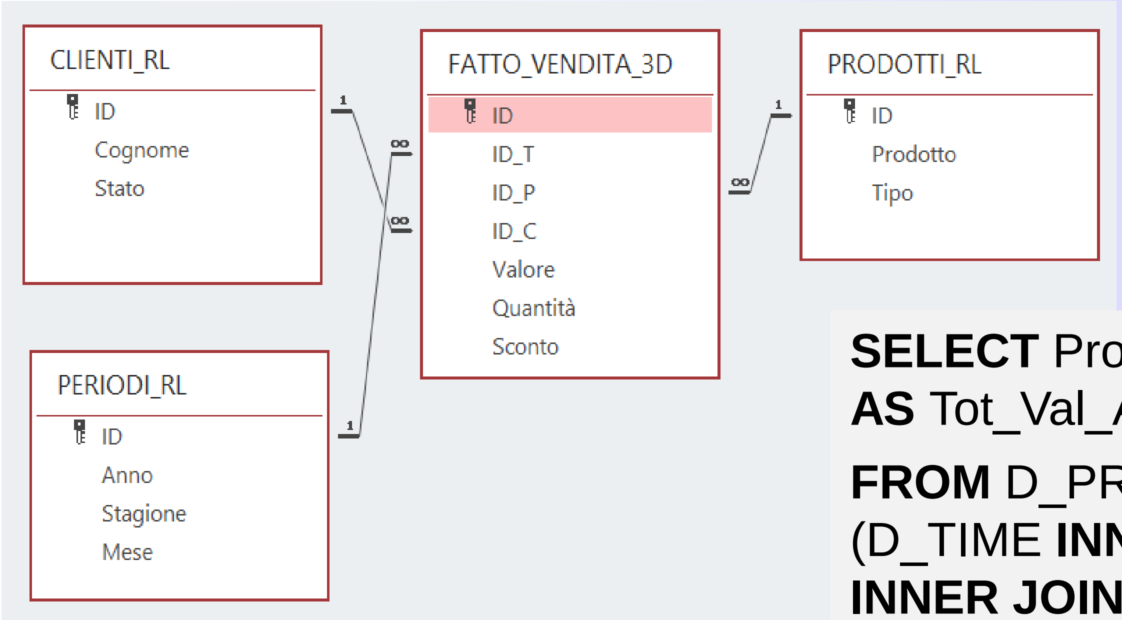
FROM D_TIME, D_PRODUCTS, D_CUSTOMERS, OR_VIEW

WHERE D_TIME.Year = OR_VIEW.Year **AND** D_TIME.Season = OR_VIEW.Season **AND** D_TIME.Month = OR_VIEW.Month **AND**

D_PRODUCTS.Product = OR_VIEW.Product **AND** D_PRODUCTS.Type = OR_VIEW.Type **AND** D_CUSTOMERS.Surname = OR_VIEW.Surname **AND** D_CUSTOMERS.Country = OR_VIEW.Country

STAR SCHEMA – Multidimensional Query

- How to create the query SALES(Country, Product,*).Value?
- We want the total sales per country and per products aggregated over all years.



Using the star-schema shown on the left the query turns out to be very easy

SELECT Product, Country, Sum(Values) **AS** Tot_Val_All_Year

FROM D_PRDOUCTS **INNER JOIN** (D_TIME **INNER JOIN** (D_CUSTOMERS **INNER JOIN** FACT_TAB **ON** D_CUSTOMERS.ID = FACT_TB.ID_Cr) **ON** D_TIME.ID = FACT_TB.ID_Tm) **ON** D_PRODUCTS.ID = FACT_TAB.ID_Pr)

GROUP BY Product, State

ORDER BY Product, State

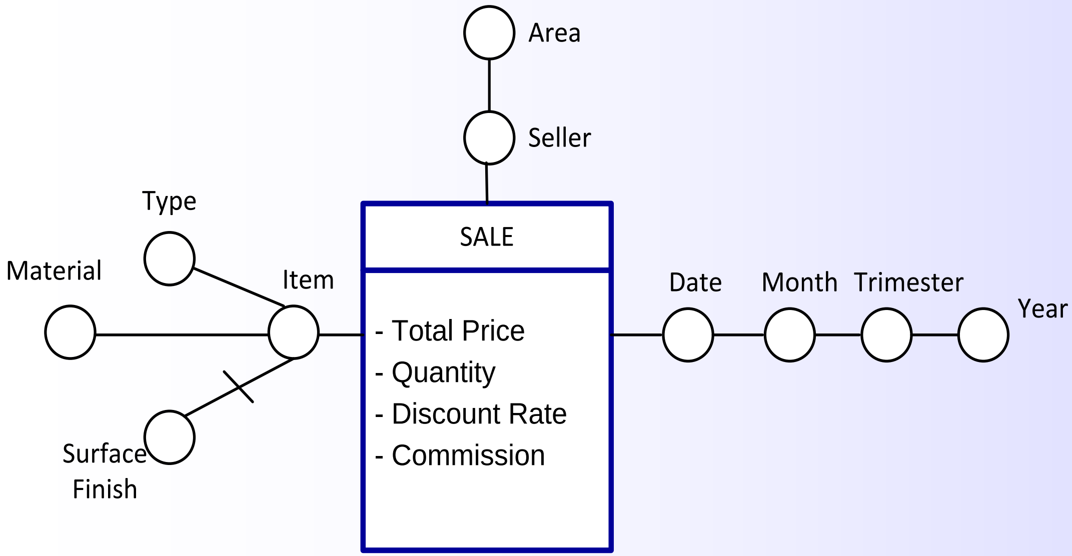
STAR SCHEMA - Example

- In this case the query is much more simple and **it can be easily automatized** (for instance using a wizard to write it)
- FACT(Dimension 1, …, Dimension n).<Measure Quantifier>
  - *Dimensions as fields of the SELECT statement*
  - *Inner join among the Fact Table* (i.e., the children table in the relationship) and *each Dimension Tables* included in the query
  - *A grouping function* operating on the <Measure Quantifier>
  - Fields corresponding to the *dimensions* of the query must be included in the GROUP BY clause

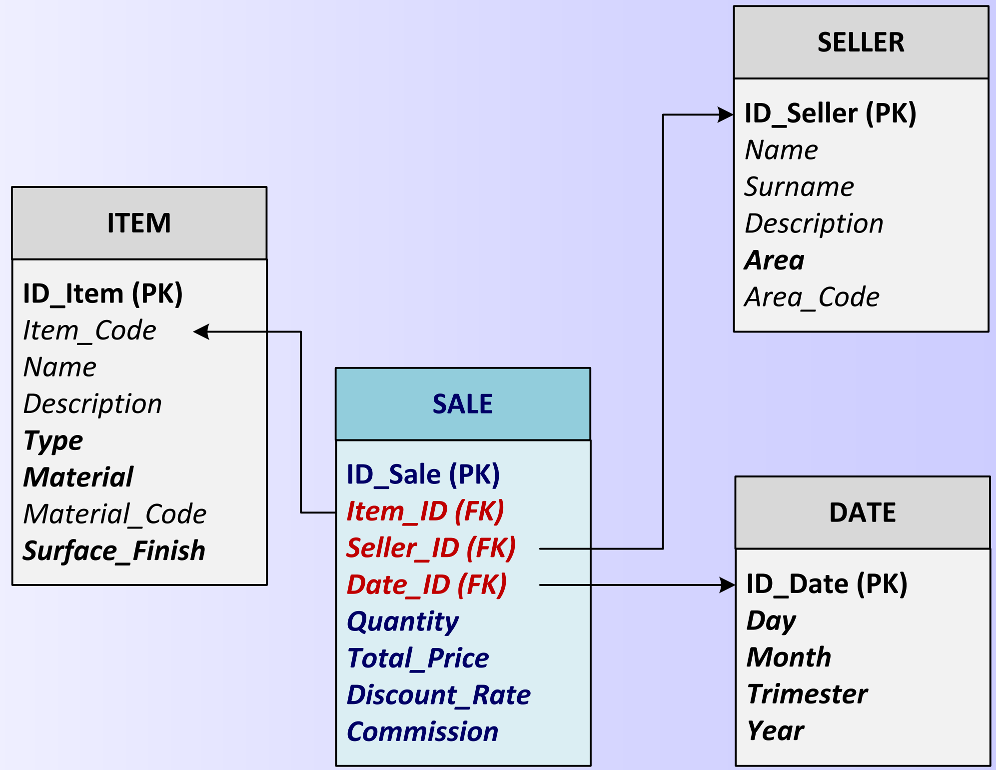
SALES(Area, Item).Revenue

```
SELECT ITEMS.Code, SALESMEN.Area,
  Sum(SALES.REVENUE) AS [Tot Revenue]
FROM SALESMEN INNER JOIN
  (ITEMS INNER JOIN  SALES ON ITEMS.ID
  = SALES.ItemID) ON SALESMEN.ID =
  SALES.SalesmanID
GROUP BY ITEMS.Code, _
SALESMEN.Area
```

STAR SCHEMA - Example



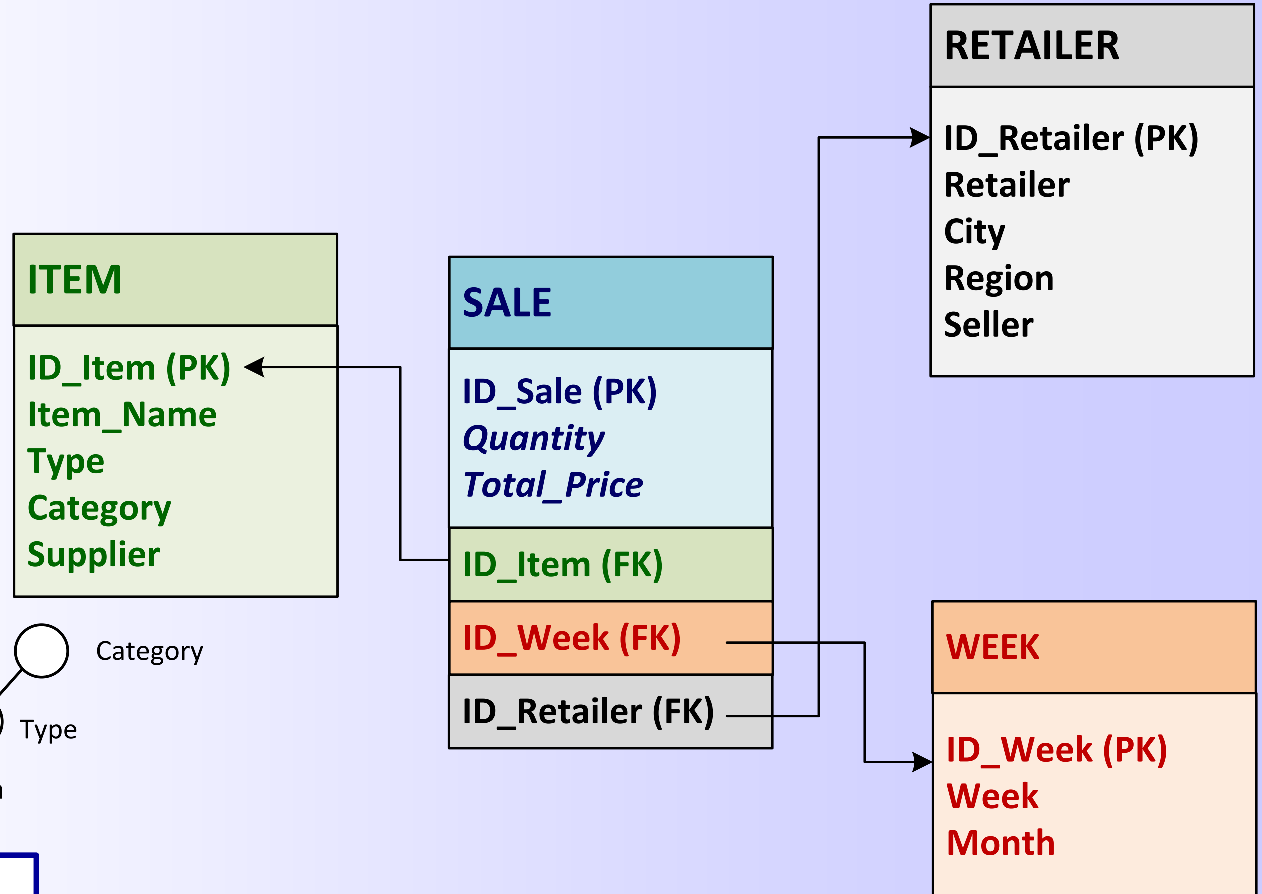
Physical realization on RDB with Star Schema

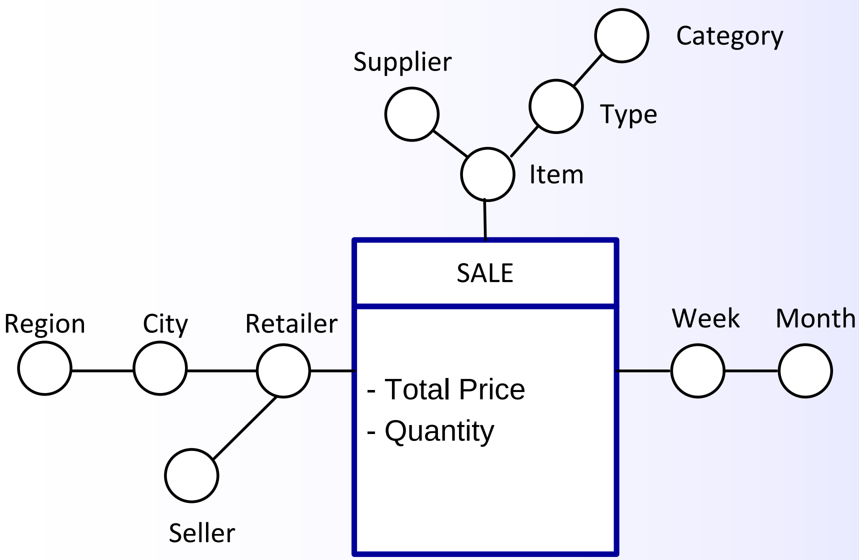


Conceptual representation through DFM diagram

STAR SCHEMA – Example (I)

- A Sale Fact
- Its DFM model
- Its Star Schema





STAR SCHEMA – Example (II)

ID_Retailer	Retailer	City	Region	Seller
1	R1	Rome	Lazio	S1
2	R2	Rome	Lazio	S1
3	R3	Milan	Lombardia	S2
4	R3	Milan	Lombardia	S2

ID_Sale	ID_Retailer	ID_Week	ID_Item	Quantity	Total Price
1	1	1	1	100	100 €
2	1	2	1	150	150 €
3	3	3	4	350	350 €
4	4	4	4	200	200 €
5	2	2	2	50	120 €
6	1	3	2	50	130 €

FACT TABLE

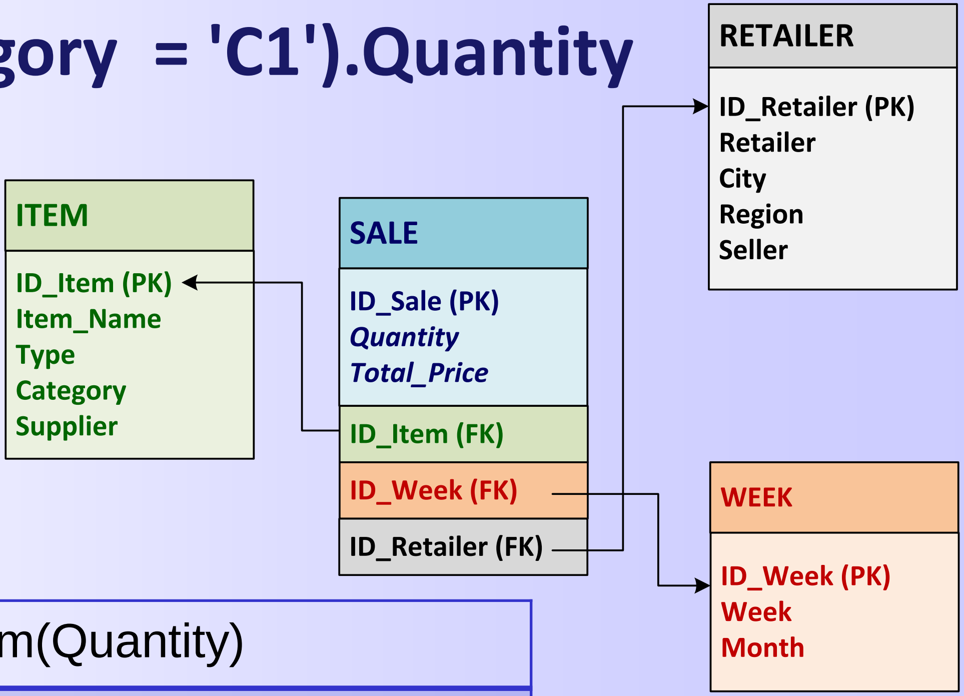
ID_Week	Week	Month
1	Jan #3	January
2	Jan #4	January
3	Feb #1	February
4	Feb #2	February

ID_Item	Item	Type	Category	Supplier
1	I1	A	C1	S1
2	I2	A	C1	S1
3	I3	B	C2	S1
4	I4	C	C1	S2

STAR SCHEMA – Example (III)

Sale(City, Week, Type, Category = 'C1').Quantity

How to write this query?!?



SELECT	City, Week, Type, Sum(Quantity)
FROM	WEEK, RETAILER, ITEM, SALE
WHERE	WEEK.ID_Week = SALE.IDC_Week ITEM.ID_Item = SALE.ID_Item RETAILER.ID_Retailer = SALE.ID_Retailer ITEM.Type = 'C1'
GROUP BY	City, Week, Type

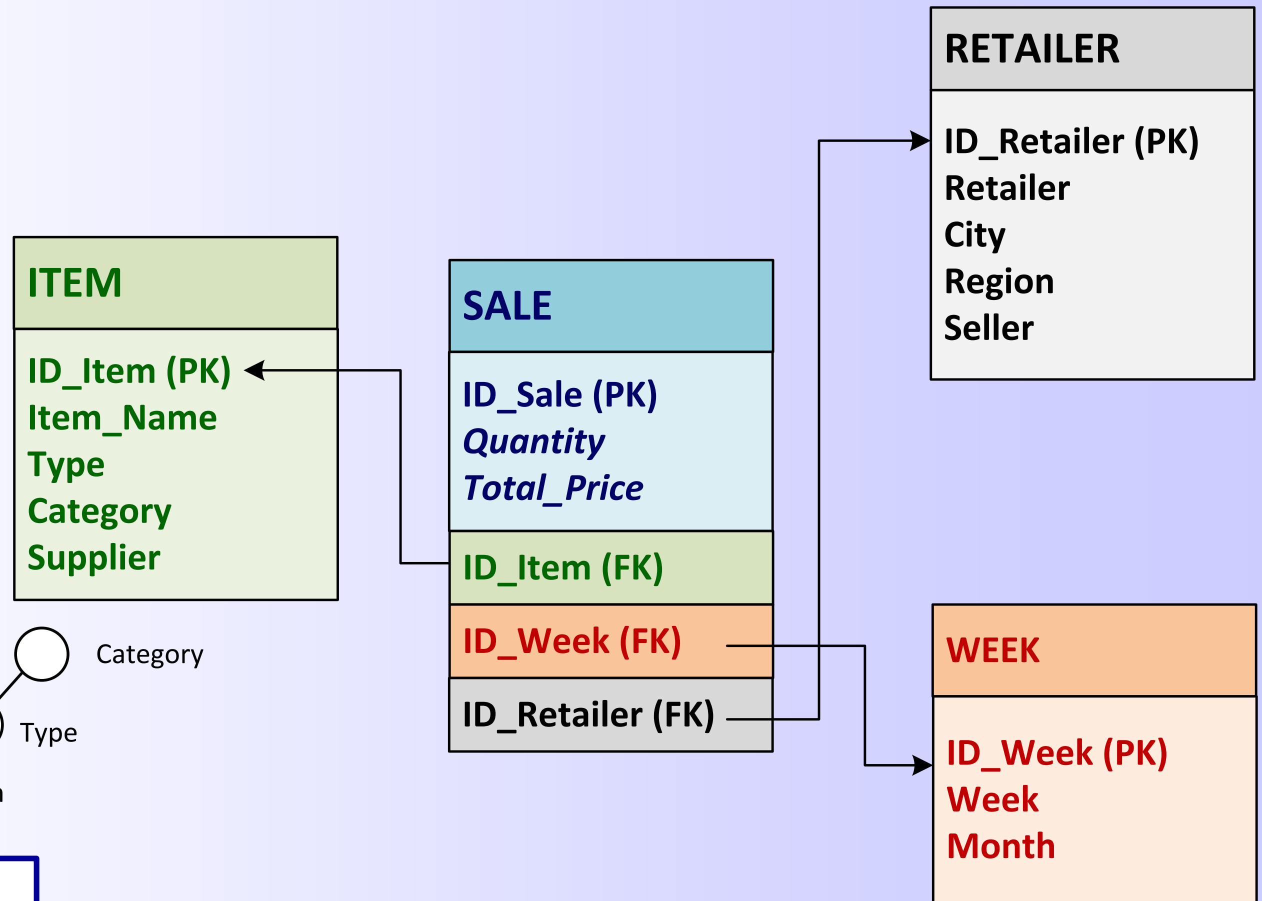
STAR SCHEMA

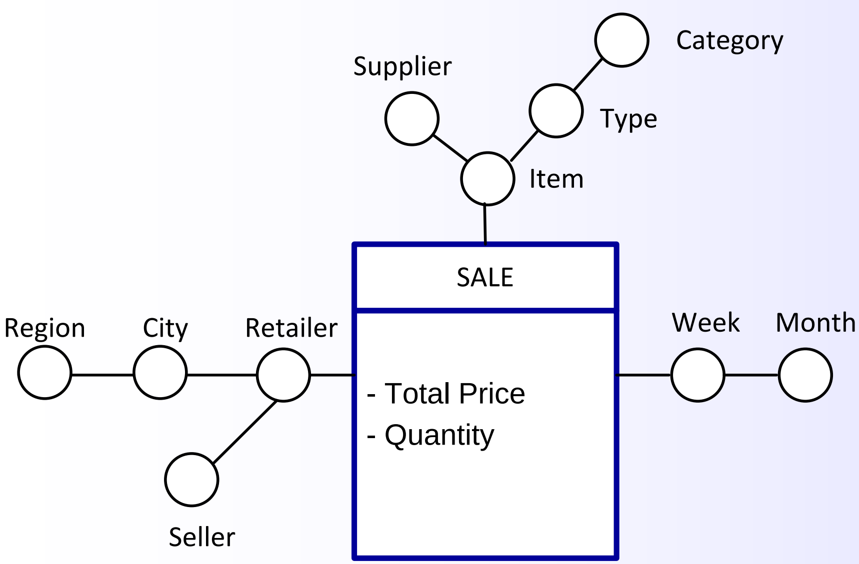
- **PROS**
  - Maximum speed to retrieve information i.e., Due to a high level of de-normalization, queries are based on single (mono level) joins

- **CONS**
  - Redundancy
  - Data structure (and representation) may not be clear
  - Data uploading is time consuming and complex
  - Long execution times due to redundancy

SNOW FLAKE SCHEMA – Example (I)

- A Sale Fact
- Its DFM model
- Its Snow Flake Schema



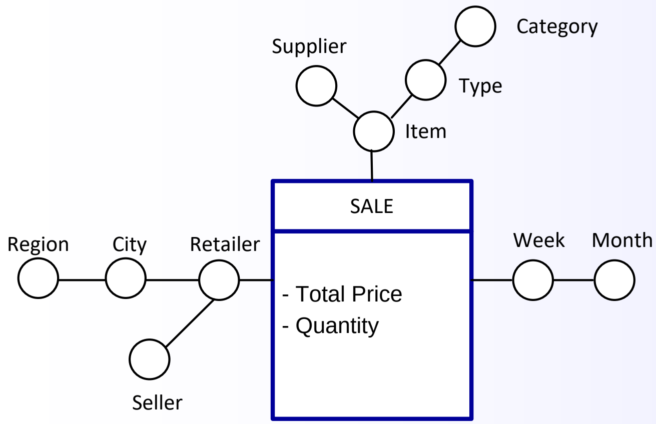


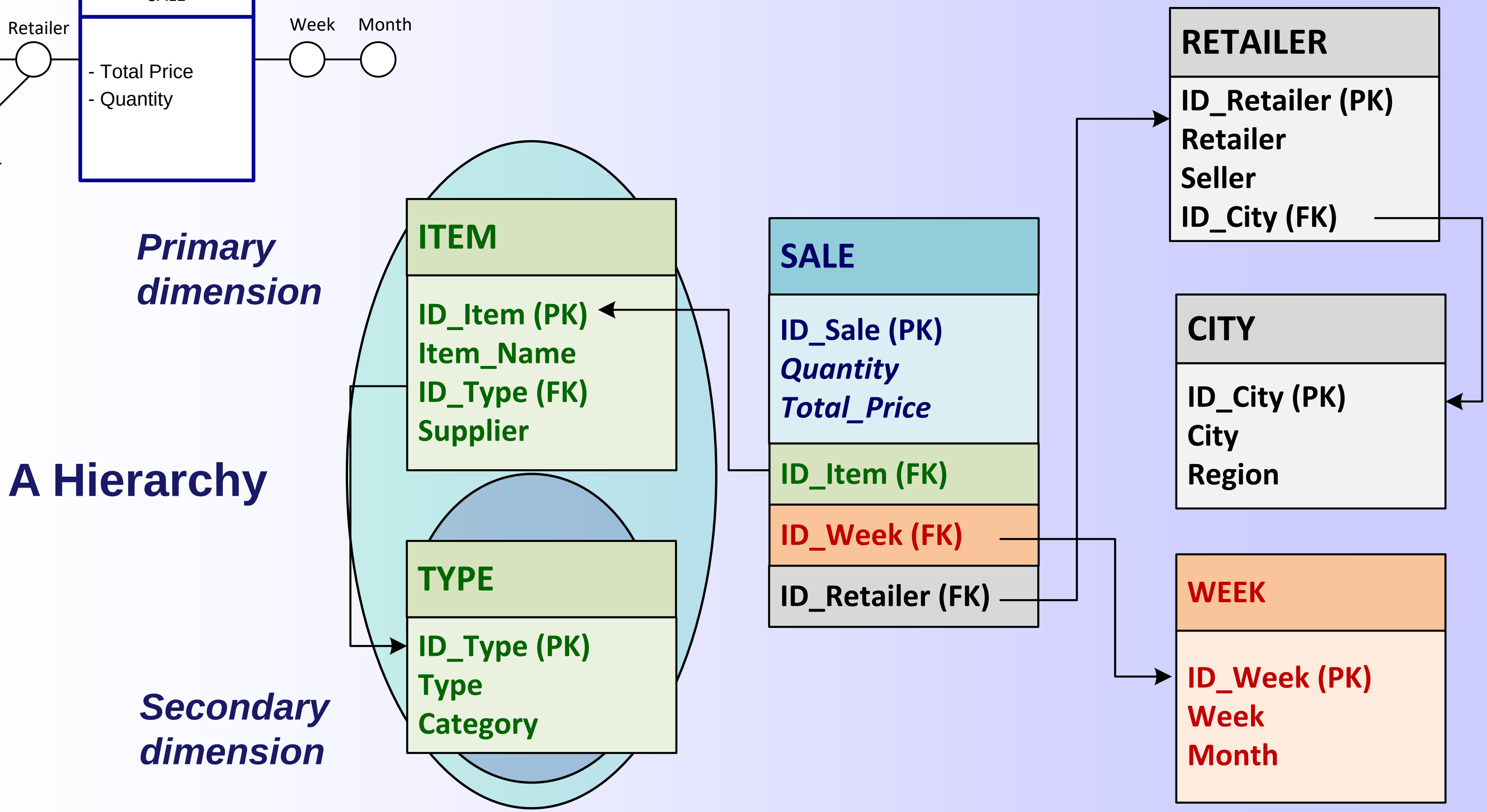
SNOW FLAKE SCHEME

- It reduces the de-normalization level
- Hierarchies are made explicit, i.e., instead being the field of a Table they are transformed in additional Tables
- In addition to the Facts' tables we have:
  - *Primary dimensions (hierarchy's root) Tables*
  - *Secondary dimensions Tables*
- Primary Dimension Tables are in OTM relation with the Fact Table
- Secondary Dimensions Tables are in OTM relation with the Primary Dimension Table, root of the hierarchy to which they belong to

SNOW FLAKE SCHEME – Example (I)

- A Sale Fact
- Its DFM model
- Its Snow Flake Schema





SNOW FLAKE SCHEME – Example (II)

ID_Ciy	City	Region
1	Rome	Lazio
2	Milan	Lombardia

ID_Retailer	Retailer	ID_City	Seller
1	R1	1	S1
2	R2	1	S1
3	R3	2	S2
4	R3	2	S2

FACT TABLE

ID_Sale	ID_Retailer	ID_Week	ID_Item	Quantity	Total Price
1	1	1	1	100	100 €
2	1	2	1	150	150 €
3	3	3	4	350	350 €
4	4	4	4	200	200 €
5	2	2	2	50	120 €
6	1	3	2	50	130 €

ID_Week	Week	Month
1	Jan #3	January
2	Jan #4	January
3	Feb #1	February
4	Feb #2	February

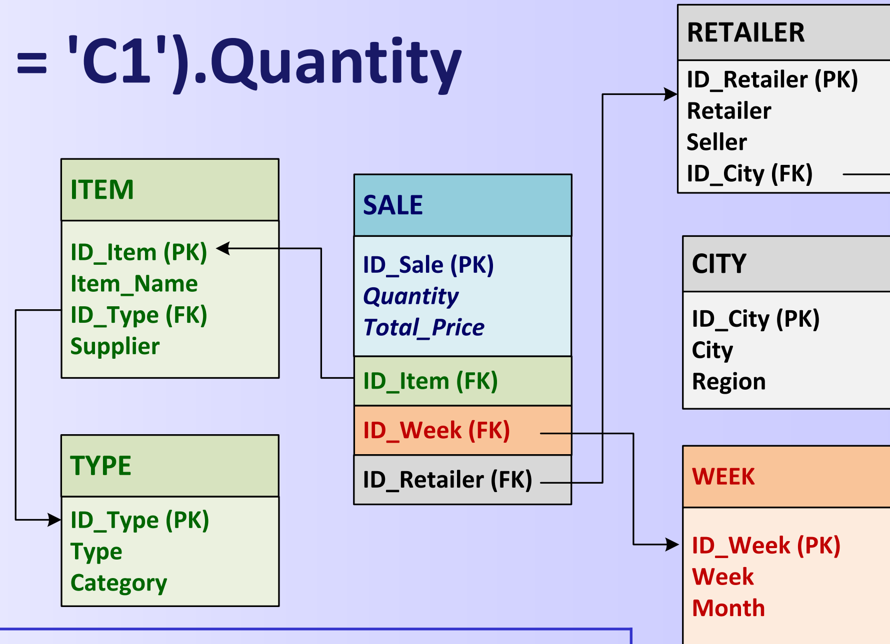
ID_Item	Item	Supplier	ID_Type
1	I1	S1	1
2	I2	S1	1
3	I3	S1	2
4	I4	S2	3

ID_Type	Type	Category
1	A	C1
2	B	C2
3	C	C1

SNOW FLAKE SCHEME – Example (III)

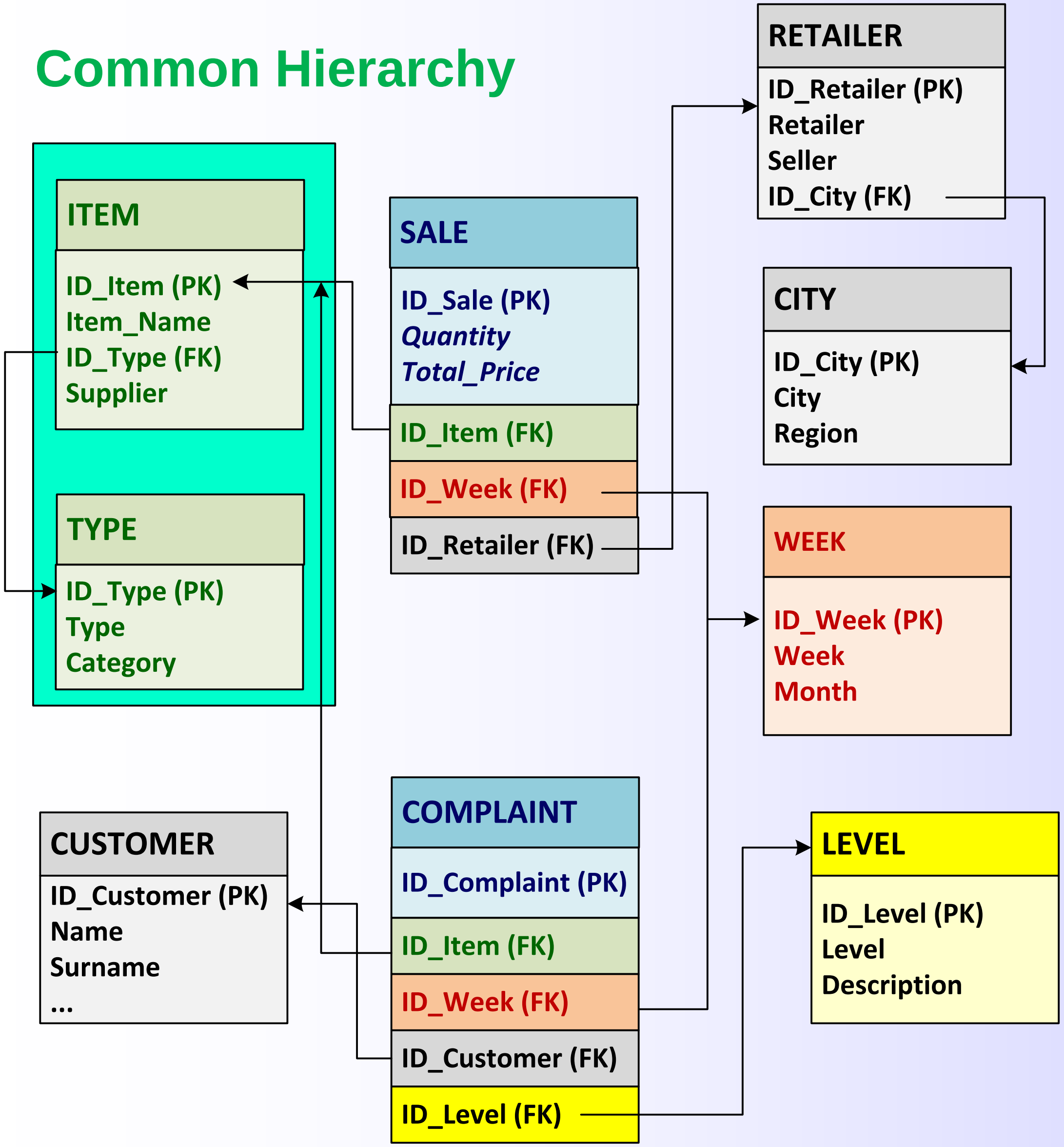
Sale(City, Week, Type, Category = 'C1').Quantity

How to write this query?!?



SELECT	City, Week, Type, Sum(Quantity)
FROM	WEEK, RETAILER, ITEM, SALE, TYPE, CITY
WHERE	WEEK.ID_Week = SALE.IDC_Week ITEM.ID_Item = SALE.ID_Item ITEM.ID_Type = TYPE.ID_Type RETAILER.ID_Retailer = SALE.ID_Retailer RETAILER.ID_City = CITY.ID_City ITEM.Type = 'C1'
GROUP BY	City, Week, Type

FACTS' CONSTELLATION



SALE and COMPLAINT facts share some hierarchies

Some hierarchies are shared by two or more facts

This strategy should be followed anytime some hierarchies are shared among facts

SNOW FLAKE SCHEMA

Useful when:

- The ratio between the cardinality of the Primary and Secondary Dimension Tables is High. In this case a considerable amount of space can be saved
- If there are shared hierarchies

PROS

- ✓ Subjects are clearly subdivided
- ✓ Better performance in case of aggregated data
- ✓ Lower sensibility with respect to variation of hierarchies over time

CONS

- Keys duplication
- Lower speed in queries execution, if secondary dimensions are needed

LOGICAL MODELS (II)

- ROLAP **– MOLAP –** HOLAP
- **MOLAP →MULTI DIMENSIONAL OLAP**
  - Facts are implemented using a real multi-dimensional DB, with positional access;
  - Queries are made using proprietary methods (MDX di Microsoft)

PRO

✓ High efficiency in queries execution

✓ Highly adherent to the conceptual model

✓ There is not the need to use SQL to create multi-dimensional queries

CONS

× Matrices are sparse; a lot of space is needed to store data

× Standards are not available, this is a hurdle for the acceptance of MOLAP

× Programmers are not very familiar with MOLAP

LOGICAL MODELS (III)

- ROLAP – MOLAP **– HOLAP**
- **HOLAP → HIBRID OLAP**
  - Intermediate solution between MOLAP and ROLAP
  - Data Warehouse is based on ROLAP
    - Ease of development
    - System Scalability
  - Data Marts are based on MOLAP
    - Queries' efficiency
    - Contained dimensions
    - Easiness of development (populating multi dimensional matrices with data coming from a warehouse that already implements a multi-dimensional approach is easier)
  - It requires a three levels architecture with a staging area