



# HCR Technology



**RDBMS**  
(Relational Database  
Management System)

- Secure, reliable, powerful
- Need an expert to drive
- Bound to its track and timetable
- Often late on arrival ...



**SPREADSHEET**

- Easy to use (without a License!)
- Fast, nimble, flexible in traffic
- Goes anywhere, but carries only one
- Can be dangerous (especially if you don't wear a helmet!)
- Not suitable for long trips



**HCR DB**  
(Hierarchical Cartesian  
Relational Database)

- Powerful, but all can drive it (with a License!)
- Flexible like a scooter, but with airbags
- Fast, for the whole family
- Comfortable, for long trips
- And it files, too! How can it be?

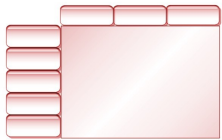
## LET'S CHOOSE OUR SOFTWARE TOOL

### RDBMS



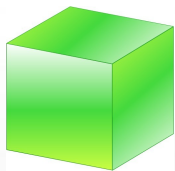
- Handles huge data sets (teraBYTES)
- Maximum reliability and protection (transactional)
- Optimized within its scope (one-dimensional)
- Can't get out of its own schema (SQL)
- Becomes inefficient if used in areas outside its scope (e.g. multidimensional queries)
- It often requires long implementation time

### SPREADSHEET



- Can be used straightaway, even without knowing about informatics
- It comes close to multidimensional ("2 ½" dimensions)
- It's a "personal" tool
- It has little semantics (no clearcut separation between data and their description)
- Can't handle great complexity (spreadsheet entropy)
- Can't handle large datasets (e.g. no more than 64000 rows)
- It isn't easy to share data and results

### HCR DB



- All can use it, with little training
- Safe and robust, but still flexible
- Merge relational and multidimensional for improved optimization
- It has semantics, avoids entropy ("crystal lattice")
- Deals with high complexity ("brain leverage" effect)
- Shortens development time
- Shares data, semantics and reporting

## WHY 'MULTIDIMENSIONAL' IS LIKE FLYING?

HCR DB



- Nearly all decisions are made on the basis of data comparisons against:
  - competitors
  - budget
  - previous year
  - different scenarios
  - different products
  - different markets
  
- Even the most simple data item (e.g. the price) is a data “**hypercube**”
  - Own price and competitors'
  - The previous year's price
  - The price in a different market
  - The previous year's competitors' price
  - The competitors' price in a different market
  - The previous year's competitors' price in a different market
  - And so on and so forth (and this for price alone!)

- Like the Copernican versus the Galilean system, method makes the difference. Choose the wrong method, and storing and managing data may become extremely difficult
- Since the storing and managing of data is the starting point for complex analyses, a poor approach to this issue will affect negatively all further developments
- What looks intricate when seen from the ground can be made easier by “flying up along the 3<sup>rd</sup> dimension “





- Relational databases can only rely on a “**fact table**”: a 3-dimensional cube is therefore stored as a table with 4 columns

YEAR	MAKER	MARKET	PRICE
2005	BMW	ITALIA	€ 30.000
2005	AUDI	ITALIA	€ 28.000
2005	BMW	GERMANIA	€ 32.000
2005	AUDI	GERMANIA	€ 30.000
2004	MERCEDES	ITALIA	€ 26.000
...	...	...	...

- This is a **costly** method. If the three dimensions have

[10 (years) × 15 (**makers**) × 20 (markets)] items

There will be 3000 rows, or 12000 cells ( 9000 alphanumeric and 3000 numeric), plus the indexes needed for searches

- It is **inefficient**. In order to retrieve data about “BMW in Spain in 2003” each and every line must be read (even if indexes are used) until the corresponding one is found. Handling of hierarchies ([see later](#)) increases inefficiency.
- It is **difficult**. SQL can't express complex queries. It must be supplemented with external tools (OLAP), putting up with longer development times and less flexibility.

### SPREADSHEET

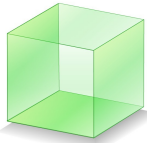


- A spreadsheet is **two-dimensional** in nature.
- Its dimensions (rows and columns) don't have semantics (A38, K25, etc.). This contributes to the entropy of any model
- The third dimension is implemented by sheets. This is nevertheless an “unlucky” dimension because relating cells across different sheets is unwieldy, and “transverse” analyses impossible.
- No provision is made for further dimensions
- Pivot tables are an attempt multidimensionality, but
  - It is complicated and knotty for the user
  - The basic entropy is inherited
  - It doesn't exploit the “sheet dimension”
  - Hierarchies are not managed ([see ahead](#))
- Limitations to roughly 65000 rows and 256 columns and to a few tens of Mb make it suitable only for small “hypercubes”

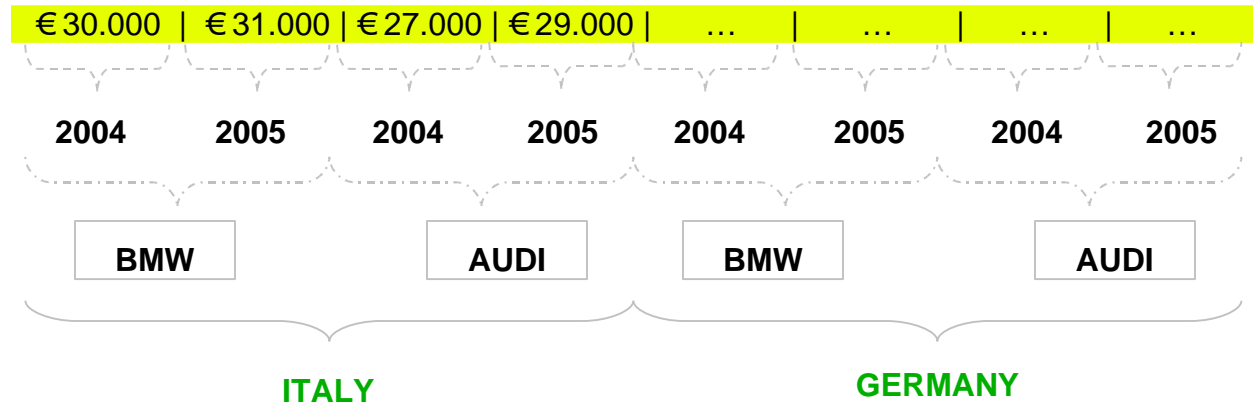
In Excel 2007, the worksheet size is 16,384 columns by 1,048,576 rows

# HOW TO ACHIEVE MULTIDIMENSIONALITY

HCR DB

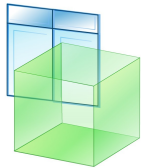


- Storage is **optimized**. A 3 dimensional cube, for instance, is stored “positionally”

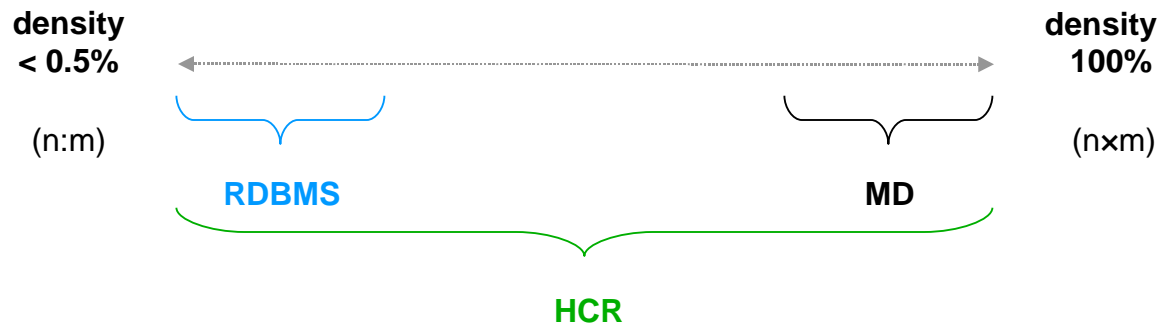


	2004	2005
GERMANY		
ITALY		
BMW	€ 30.000	€ 31.000
AUDI	€ 27.000	€ 29.000
...	€ ...	€ ...

- Any datum is **immediately singled out (direct access)**. The position (offset) of a cell can be obtained by a simple algebraic calculation (independent of the size of the cube)
- The **high data access efficiency** allows for the fast management of hierarchies
- Users can really think multidimensionally, concentrating on high level matters, because HCR handles multidimensionality (“brain leverage”).



- Very often data are in a “**many to many**” relation (A:B), which differs from the “**all to all**” relation called “**cartesian product**” (AxB), that characterizes traditional multidimensional databases. A typical example: products vs. markets.
- In reality, there is a continuum between “many to many” and “cartesian product”. The parameter that drives the transition is “**sparsity**” or its reciprocal, the **density**.



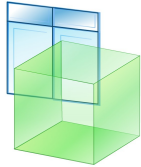
- Besides, in the same cube some dimensions may be in a “many to many” relation, while others are “cartesian”.

e.g.: A:BxCxD

- a **RDBMS** stores them as **A:B:C:D**, creating very long and inefficient **fact tables**
- a purely multidimensional system (**MD**) stores them as **A x B x C x D**, creating a huge, mostly empty (sparse) cube.

HCR DB

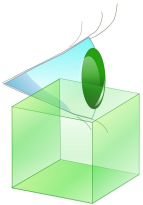
- The **HCR database** uses its **cartesian features (C)** for the multidimensional part and the **relational features (R)** for the “many to many” part, thus creating a **mixed structure, dense and efficient (A:B)**



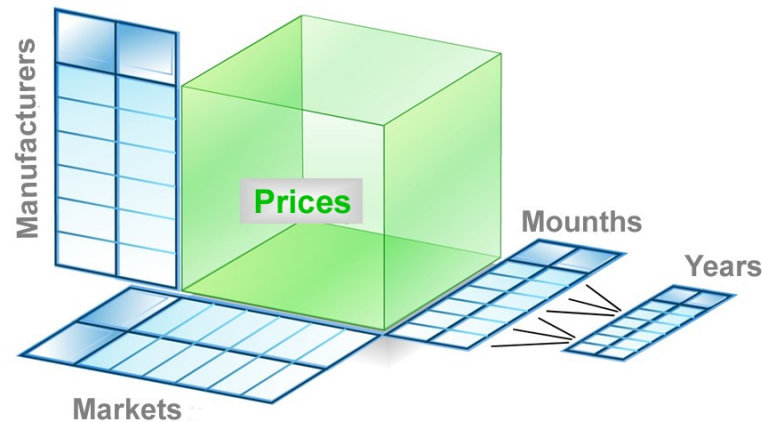
Multidimensional “Records”

(A:B)		
A	B	
A1	B1	 (C×D)
A1	B2	 (C×D)
A1	B3	...
A2	B1	...
...	...	...

(A:B)×C×D

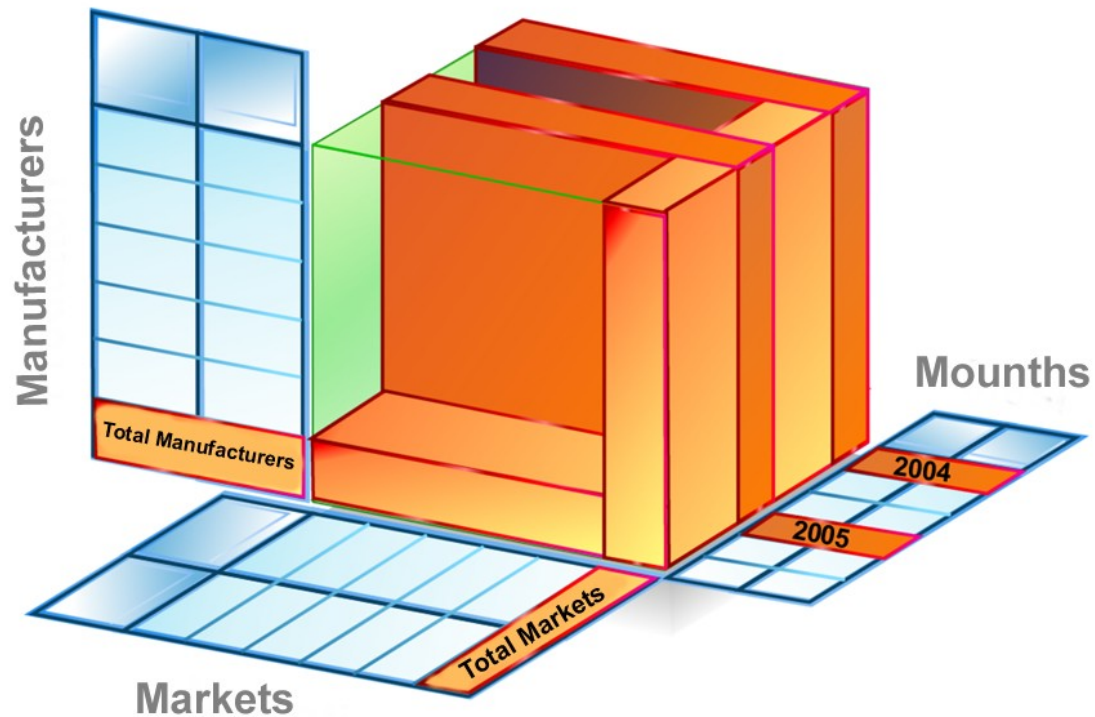
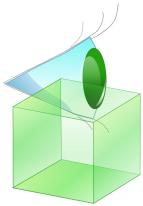


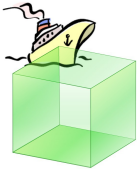
- Anche se il criterio principe per l'analisi dei dati è come visto in precedenza, il confronto e, spesso, la numerosità dei dati stessi (milioni, spesso miliardi di celle), rende impossibile considerarli tutti.
- Data are therefore **aggregated** to provide a summary view that can be used as a starting point for detailed analyses
- In **relational DBs** this technique is implemented with “**one to many**” relations (1:N).
- The **HCR DB** implements it with the **hierarchical feature**, that **goes beyond and exploits multidimensionality**.



HCR DB

- The **summary views** (colored in orange) are indeed many and **only a hypercubic structure can contain them.**





- It is the way to “flatten” the many cube's dimension to to two-dimensional displays and printers.
- It is also a new way of posing a “**query**”.Rather than asking for rows and apply filters one “moves” along data,starting from summary all the way down to details (**H**), compares them with “similar” data, that because of multidimensionality are close to each other in the cube (**C**).
- These techniques are called:
  - “**drill down**” (climbing up and down hierarchies);
  - “**dicing**” (rotatele dimensions);
  - “**slicing**” (flip through pages).
- The same browsing methods can be used also for all graphics

### HCR DB



- Once data are stored into a HCR db it becomes possible to think at a higher abstraction level, while the HCR db takes care of the complexities .
- Each cube becomes part of a higher level design that may encompass, for instance
  - Creating scenarios and simulations (**what if analysis**)
  - **“Merging” data** originating from different “worlds”
  - **“Disaggregating” summary data** according to statistical rules (drivers)
  - **integrate data** that, by their nature, have different detail levels
  - etc.
- The **“brain leverage”** is thus achieved, that makes it possible to use technology as an **“amplifier”**, not a substitute, **of our intellectual strengths.**