Business Information Systems

OLAP Cubes in Datawarehousing

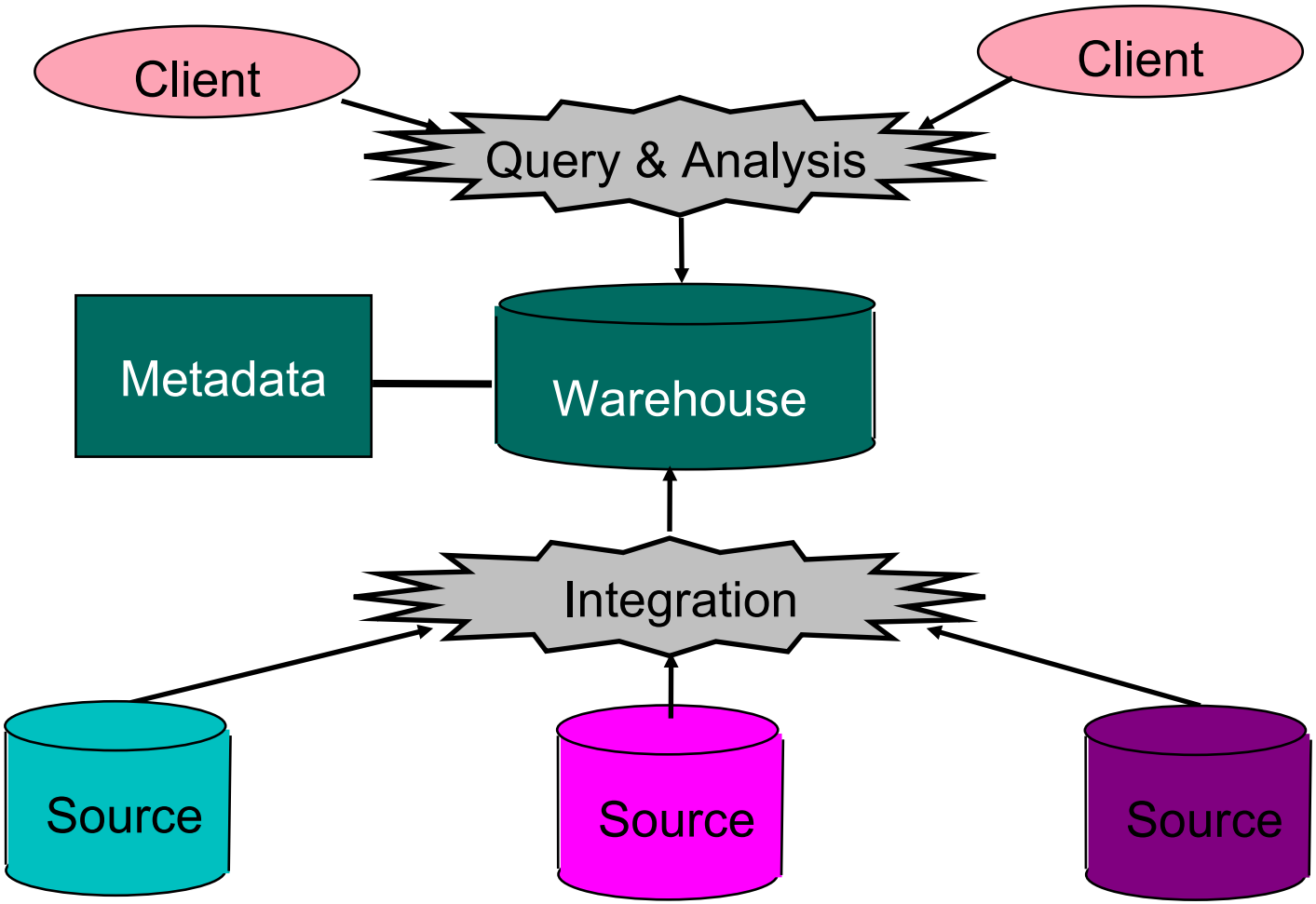
Prithwis Mukerjee, Ph.D.

•Acknowledgement
•Hector Garcia Molina – Stanford
•FORWISS - Bavarian Research Centre for Knowledge Based Systems

What is a Warehouse?

- Collection of diverse data
  - subject oriented
  - aimed at executive, decision maker
  - often a copy of operational data
  - with value-added data (e.g., summaries, history)
  - integrated
  - time-varying
  - non-volatile
- Collection of tools
  - gathering data
  - cleansing, integrating, ...
  - querying, reporting, analysis
  - data mining
  - monitoring, administering warehouse

Warehouse Architecture



OLTP vs. OLAP

- OLTP: On Line Transaction Processing
  - Describes processing at operational sites
  - Mostly updates
  - Many small transactions
  - Mb-Tb of data
  - Raw data
  - Clerical users
  - Up-to-date data
  - Consistency, recoverability critical
- OLAP: On Line Analytical Processing
  - Describes processing at warehouse
  - Mostly reads
  - Queries long, complex
  - Gb-Tb of data
  - Summarized, consolidated data
  - Decision-makers, analysts as users

Data Marts

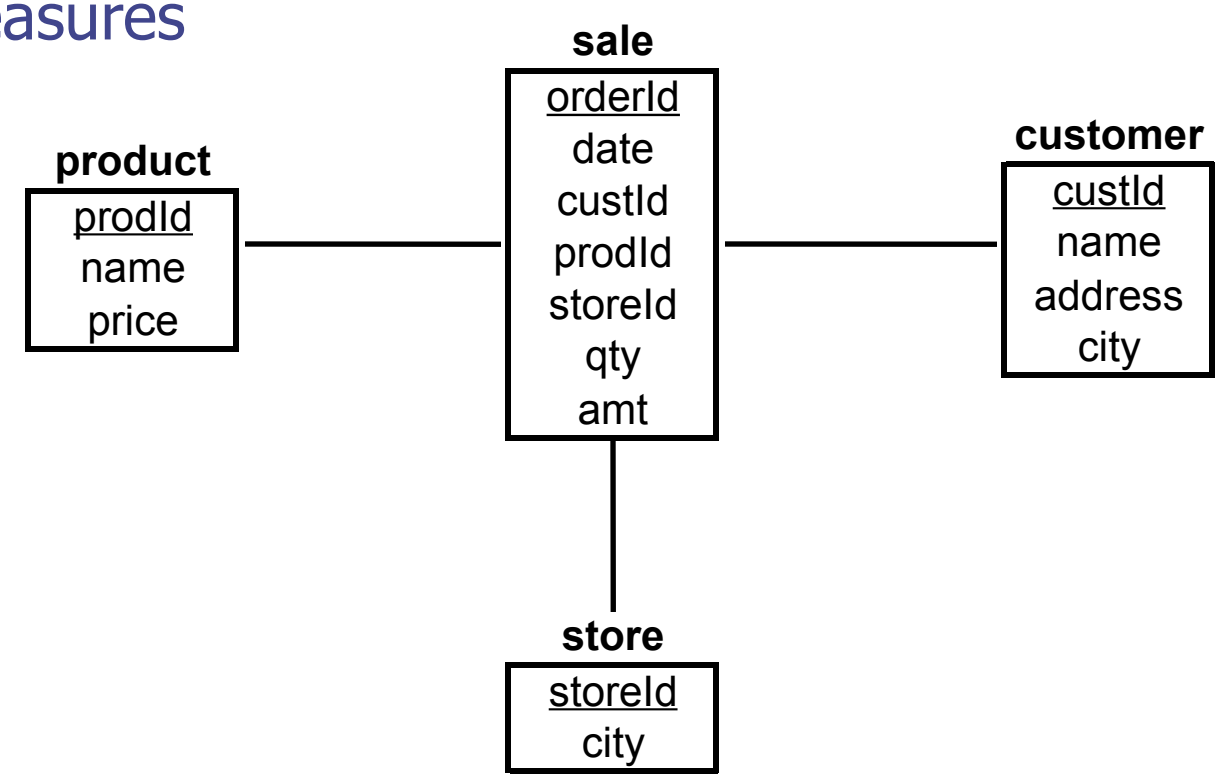
- Smaller warehouses
- Spans part of organization
  - e.g., marketing (customers, products, sales)
- Do not require enterprise-wide consensus
  - but long term integration problems?

Warehouse Models & Operators

- Data Models
  - relations
  - stars & snowflakes
  - cubes
- Operators
  - slice & dice
  - roll-up, drill down
  - pivoting
  - other

Star Schema Terms

- Fact table
- Dimension tables
- Measures



Star

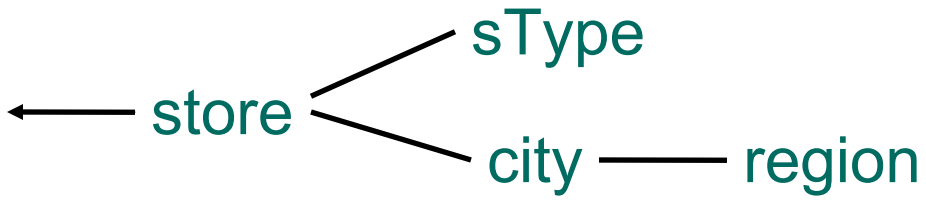
product	prodId	name	price
	p1	bolt	10
	p2	nut	5

store	storeId	city
	c1	nyc
	c2	sfo
	c3	la

sale	oderId	date	custId	prodId	storeId	qty	amt
	o100	1/7/97	53	p1	c1	1	12
	o102	2/7/97	53	p2	c1	2	11
	105	3/8/97	111	p1	c3	5	50

customer	custId	name	address	city
	53	joe	10 main	sfo
	81	fred	12 main	sfo
	111	sally	80 willow	la

Dimension Hierarchies



store	storeId	cityId	tId	mgr
	s5	sfo	t1	joe
	s7	sfo	t2	fred
	s9	la	t1	nancy

sType	tId	size	location
	t1	small	downtown
	t2	large	suburbs

city	cityId	pop	regId
	sfo	1M	north
	la	5M	south

region	regId	name
	north	cold region
	south	warm region

- ➔ snowflake schema
- ➔ constellations

Cube

Fact table view:

sale	prodId	storeId	amt
	p1	c1	12
	p2	c1	11
	p1	c3	50
	p2	c2	8

Multi-dimensional cube:

	c1	c2	c3
p1	12		50
p2	11	8	

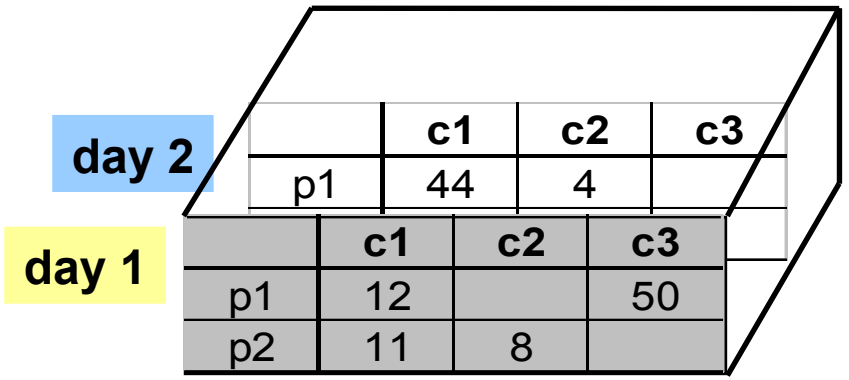
dimensions = 2

3-D Cube

Fact table view:

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

Multi-dimensional cube:



dimensions = 3

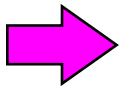
ROLAP vs. MOLAP

- ROLAP:
  Relational On-Line Analytical Processing
- MOLAP:
  Multi-Dimensional On-Line Analytical Processing

Aggregates

- Add up amounts for day 1
- In SQL: SELECT sum(amt) FROM SALE
  WHERE date = 1

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

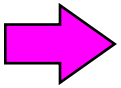


81

Aggregates

- Add up amounts by day
- In SQL: SELECT date, sum(amt) FROM SALE
  GROUP BY date

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

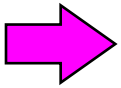


ans	date	sum
	1	81
	2	48

Another Example

- Add up amounts by day, product
- In SQL: SELECT date, sum(amt) FROM SALE
  GROUP BY date, prodId

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4



sale	prodId	date	amt
	p1	1	62
	p2	1	19
	p1	2	48

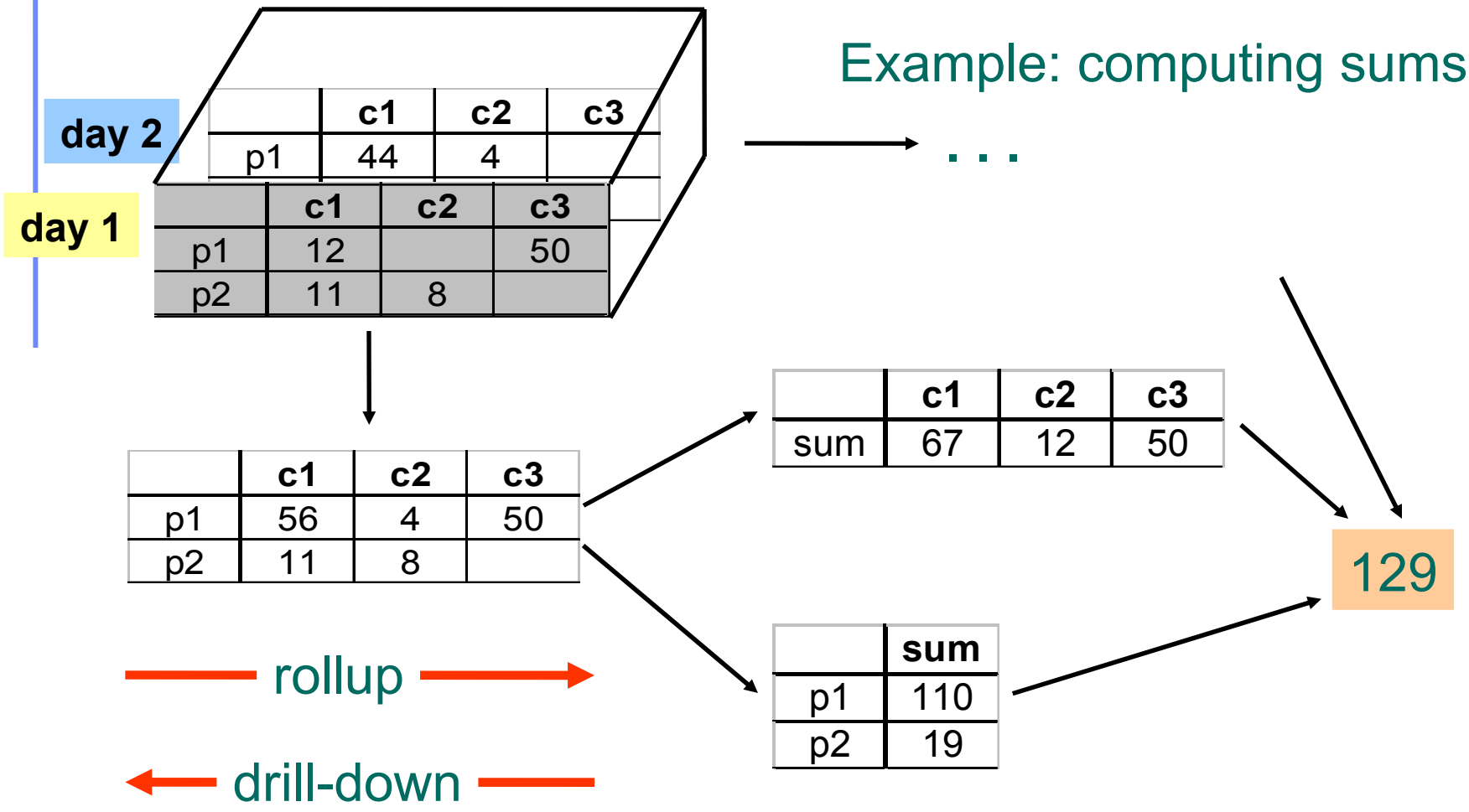
rollup →

← drill-down

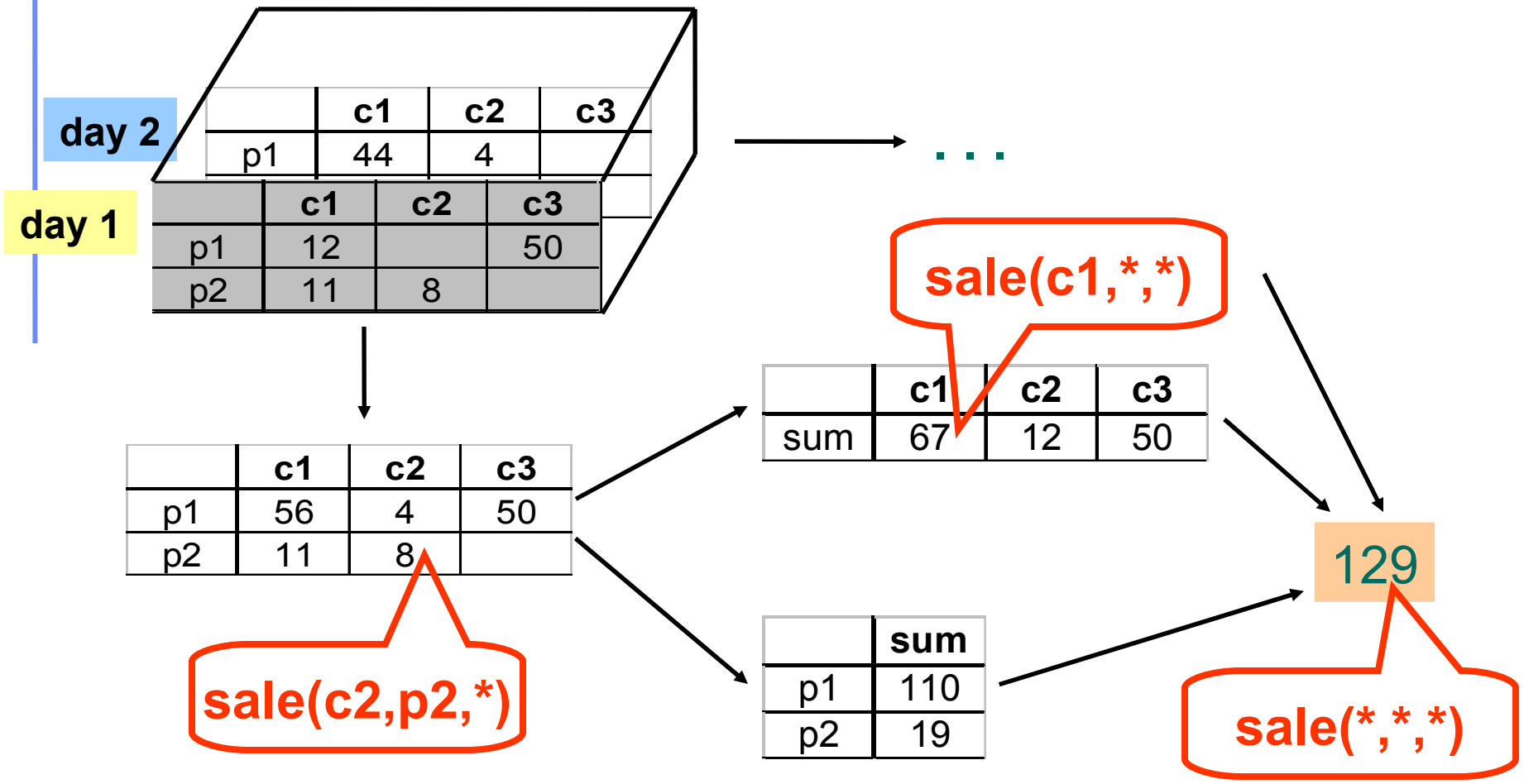
Aggregates

- Operators: sum, count, max, min, median, ave
- "Having" clause
- Using dimension hierarchy
  - average by region (within store)
  - maximum by month (within date)

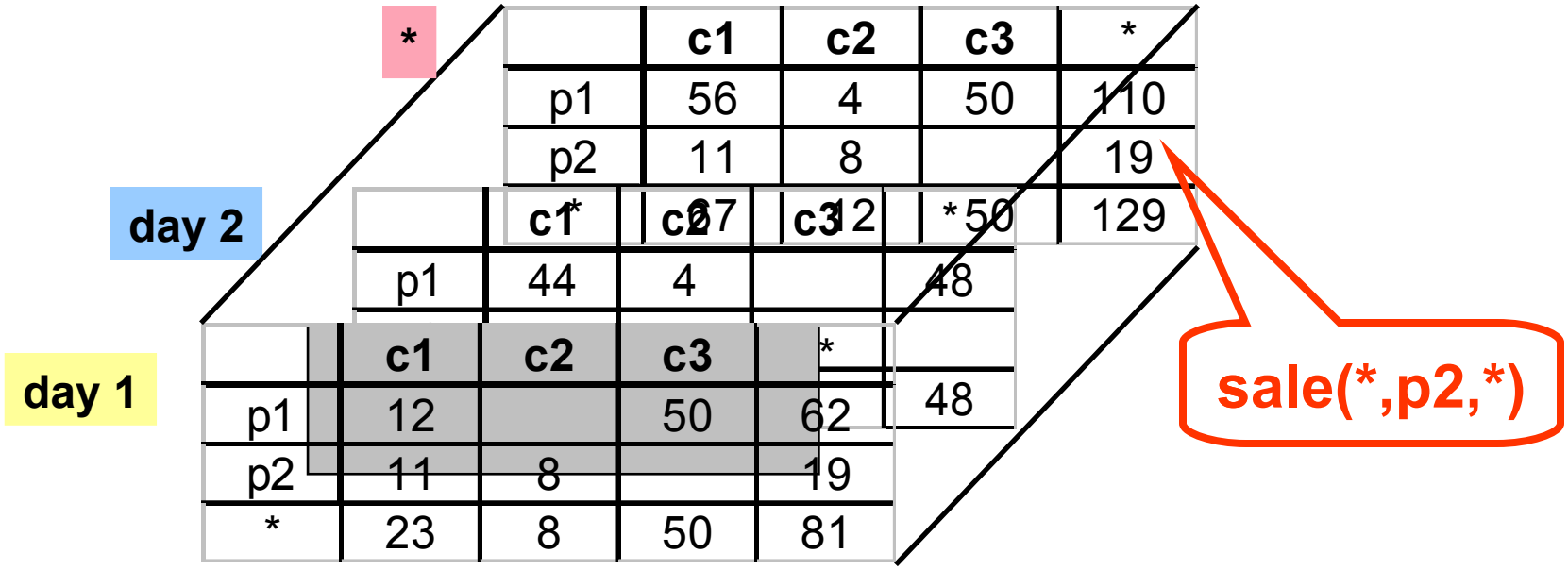
Cube Aggregation



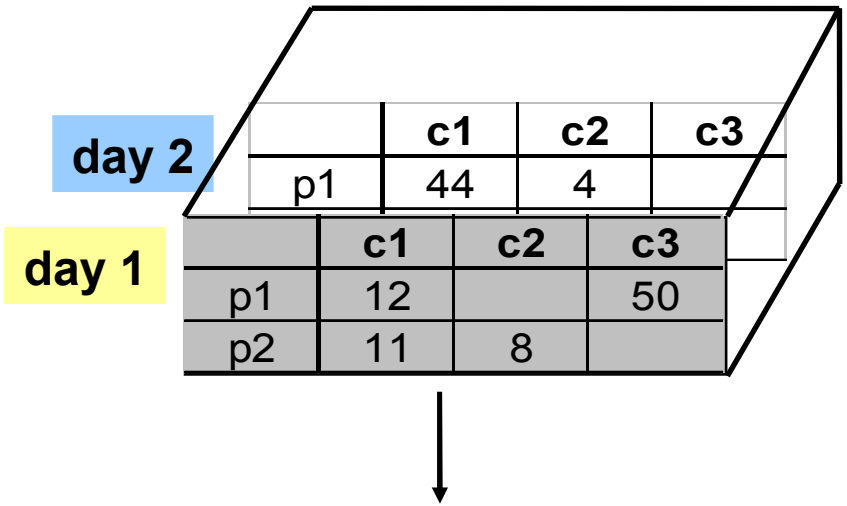
Cube Operators



Extended Cube



Aggregation Using Hierarchies



	region A	region B
p1	56	54
p2	11	8

customer

region

country

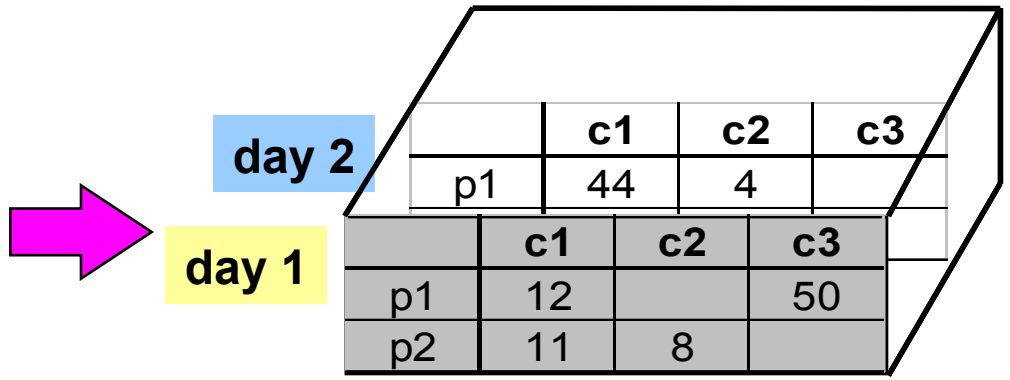
(customer c1 in Region A;
customers c2, c3 in Region B)

Pivoting

Fact table view:

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

Multi-dimensional cube:



	c1	c2	c3
p1	56	4	50
p2	11	8	

What is a Multi-Dimensional Database?

*A multi**dimensional** database (MDDB) is a computer software system designed to allow for the efficient and convenient storage and retrieval of large volumes of data that are*

- *intimately related and*
- *stored, viewed and analyzed from different **perspectives**. These perspectives are called **dimensions**.*

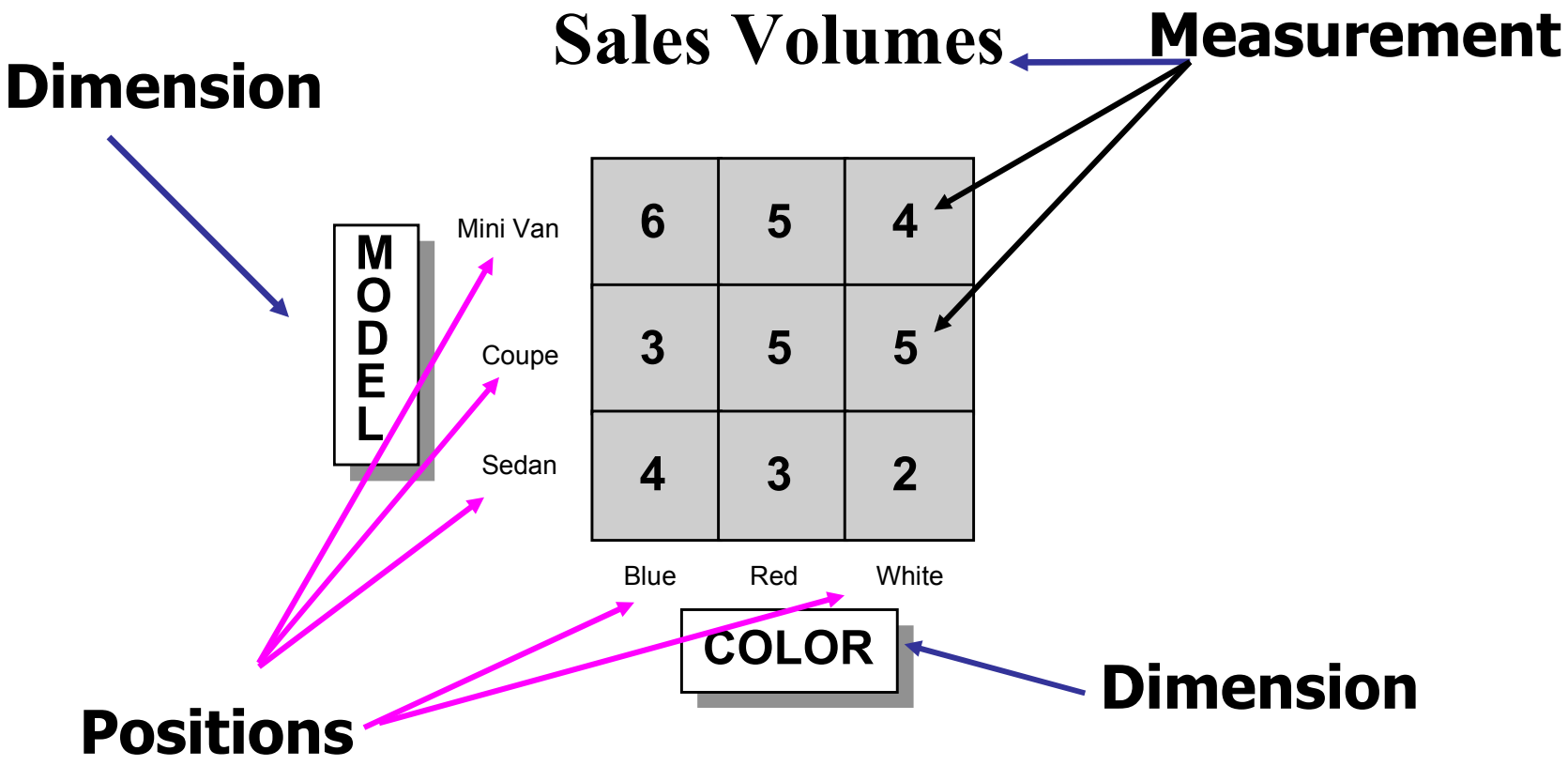
Relational and Multi-Dimensional Models: An Example

SALES VOLUMES FOR GLEASON DEALERSHIP

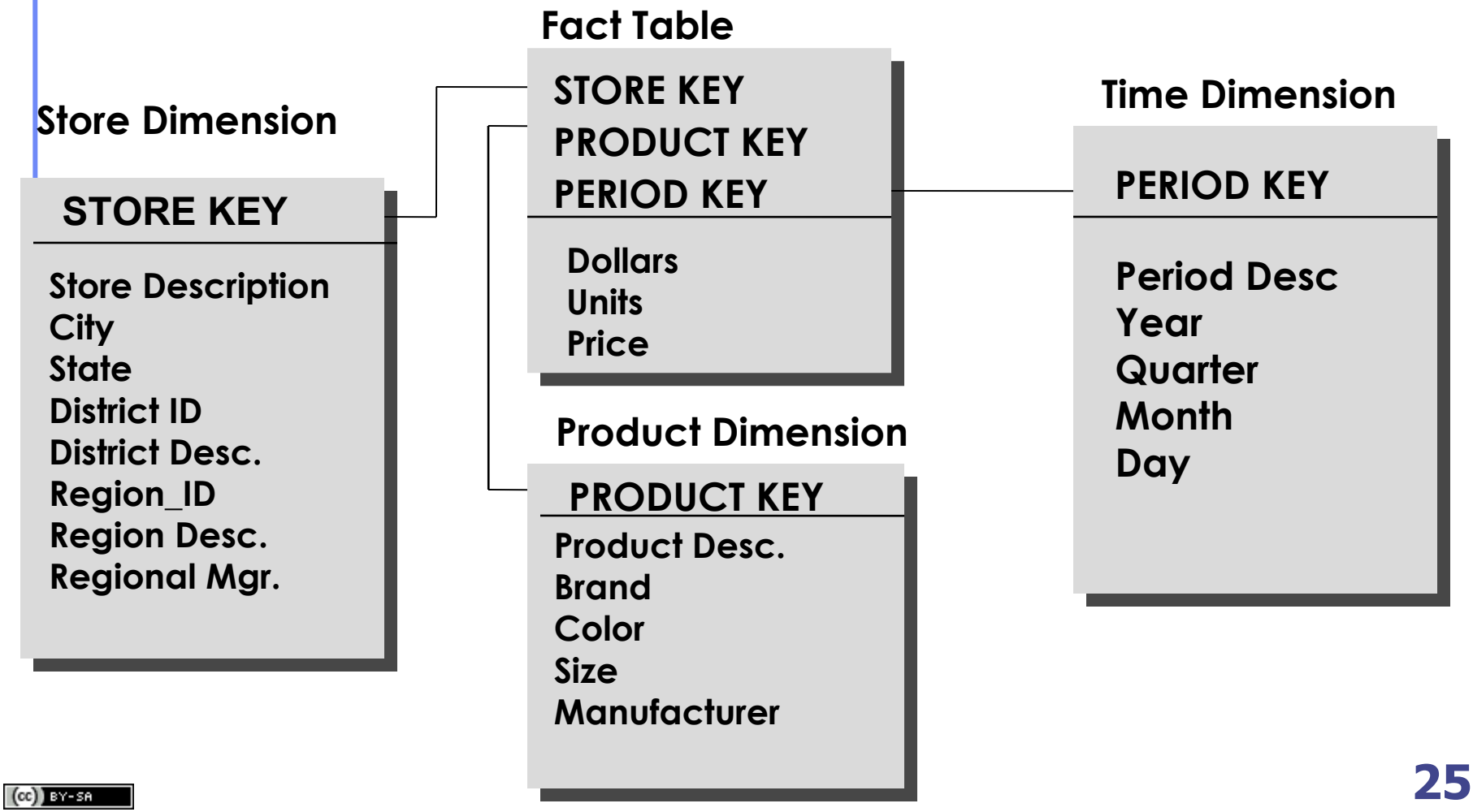
MODEL	COLOR	SALES VOLUME
MINI VAN	BLUE	6
MINI VAN	RED	5
MINI VAN	WHITE	4
SPORTS COUPE	BLUE	3
SPORTS COUPE	RED	5
SPORTS COUPE	WHITE	5
SEDAN	BLUE	4
SEDAN	RED	3
SEDAN	WHITE	2

The Relational Structure

Multidimentional Structure



The "Classic" Star Scheme



Differences between MDDB and Relational Databases

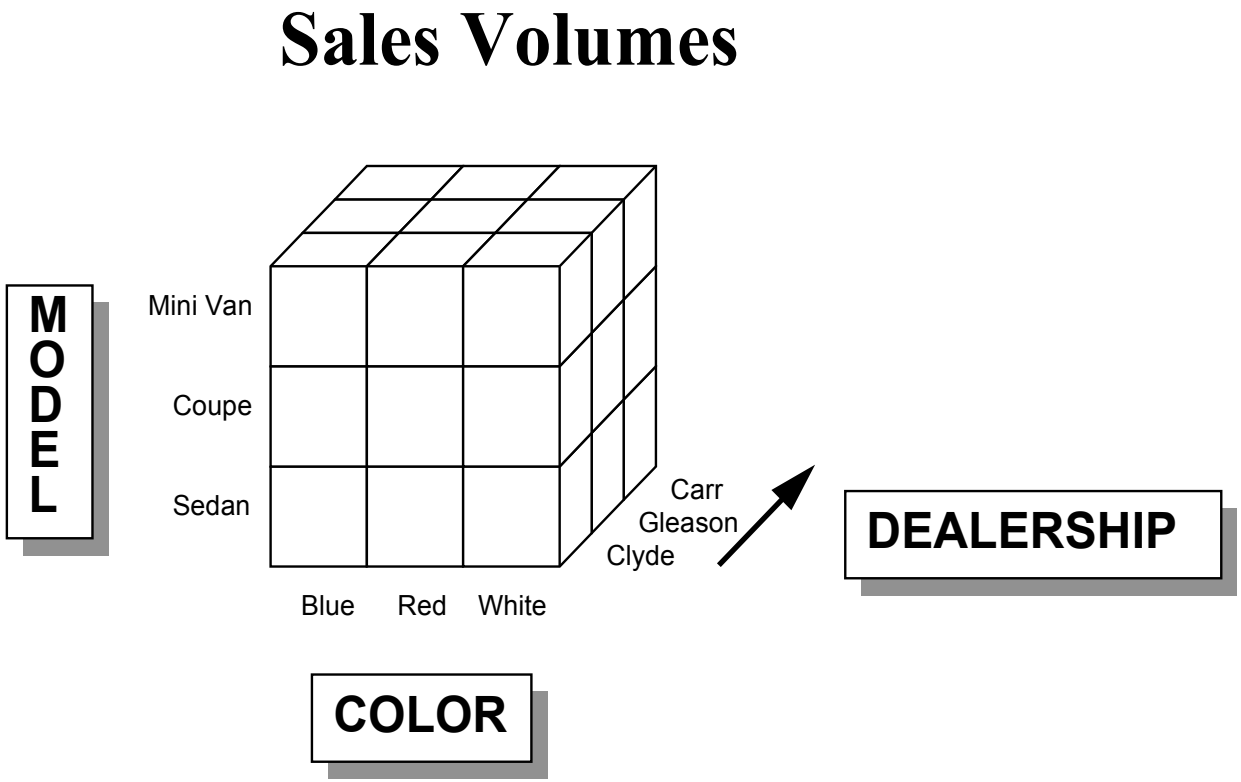
Normalized Relational	MDDB
Data reorganized based on query. Perspectives are placed in the fields – tells us nothing about the contents	Perspectives embedded directly in the structure.
Browsing and data manipulation are not intuitive to user	Data retrieval and manipulation are easy
Slows down for large datasets due to multiple JOIN operations needed.	Fast retrieval for large datasets due to predefined structure.
Flexible. Anything an MDDB can do, can be done this way.	Relatively Inflexible. Changes in perspectives necessitate reprogramming of structure.

Relational Model and Multi Dimensional Databases -Example 2

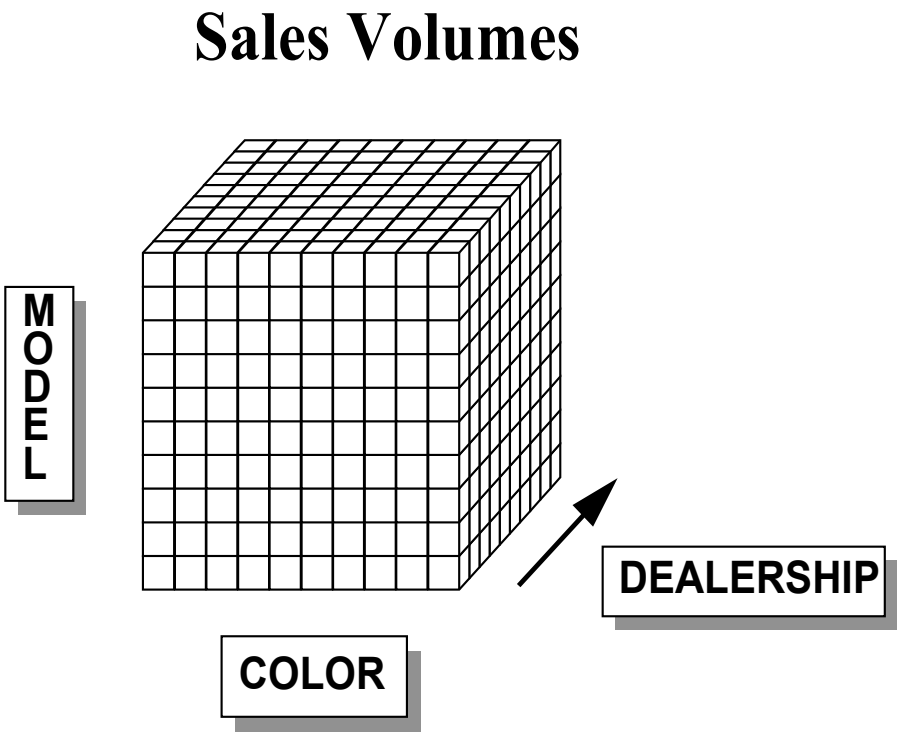
SALES VOLUMES FOR ALL DEALERSHIPS

MODEL	COLOR	DEALERSHIP	VOLUME
MINI VAN	BLUE	CLYDE	6
MINI VAN	BLUE	GLEASON	6
MINI VAN	BLUE	CARR	2
MINI VAN	RED	CLYDE	3
MINI VAN	RED	GLEASON	5
MINI VAN	RED	CARR	5
MINI VAN	WHITE	CLYDE	2
MINI VAN	WHITE	GLEASON	4
MINI VAN	WHITE	CARR	3
SPORTS COUPE	BLUE	CLYDE	2
SPORTS COUPE	BLUE	GLEASON	3
SPORTS COUPE	BLUE	CARR	2
SPORTS COUPE	RED	CLYDE	7
SPORTS COUPE	RED	GLEASON	5
SPORTS COUPE	RED	CARR	2
SPORTS COUPE	WHITE	CLYDE	4
SPORTS COUPE	WHITE	GLEASON	5
SPORTS COUPE	WHITE	CARR	1
SEDAN	BLUE	CLYDE	6
SEDAN	BLUE	GLEASON	4
SEDAN	BLUE	CARR	2
SEDAN	RED	CLYDE	1
SEDAN	RED	GLEASON	3
SEDAN	RED	CARR	4
SEDAN	WHITE	CLYDE	2
SEDAN	WHITE	GLEASON	2
SEDAN	WHITE	CARR	3

Mutlidimensional Representation

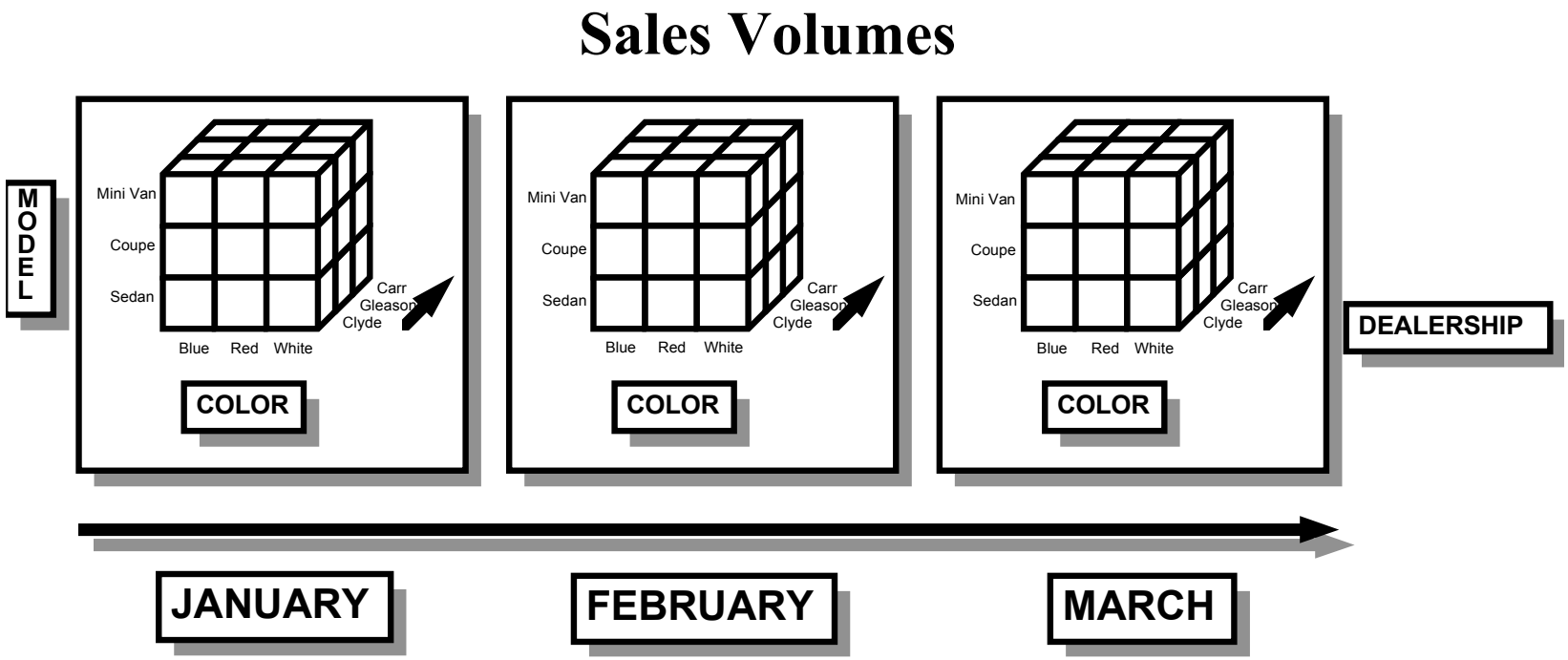


Viewing Data - An Example



- Assume that each dimension has 10 positions, as shown in the cube above
- How many records would be there in a relational table?
- Implications for viewing data from an end-user standpoint?

Adding Dimensions- An Example



When is MDD (In)appropriate?

First, consider situation 1

PERSONNEL

LAST NAME	EMPLOYEE#	EMPLOYEE AGE
SMITH	01	21
REGAN	12	19
FOX	31	63
WELD	14	31
KELLY	54	27
LINK	03	56
KRANZ	41	45
LUCUS	33	41
WEISS	23	19

When is MDD (In)appropriate?

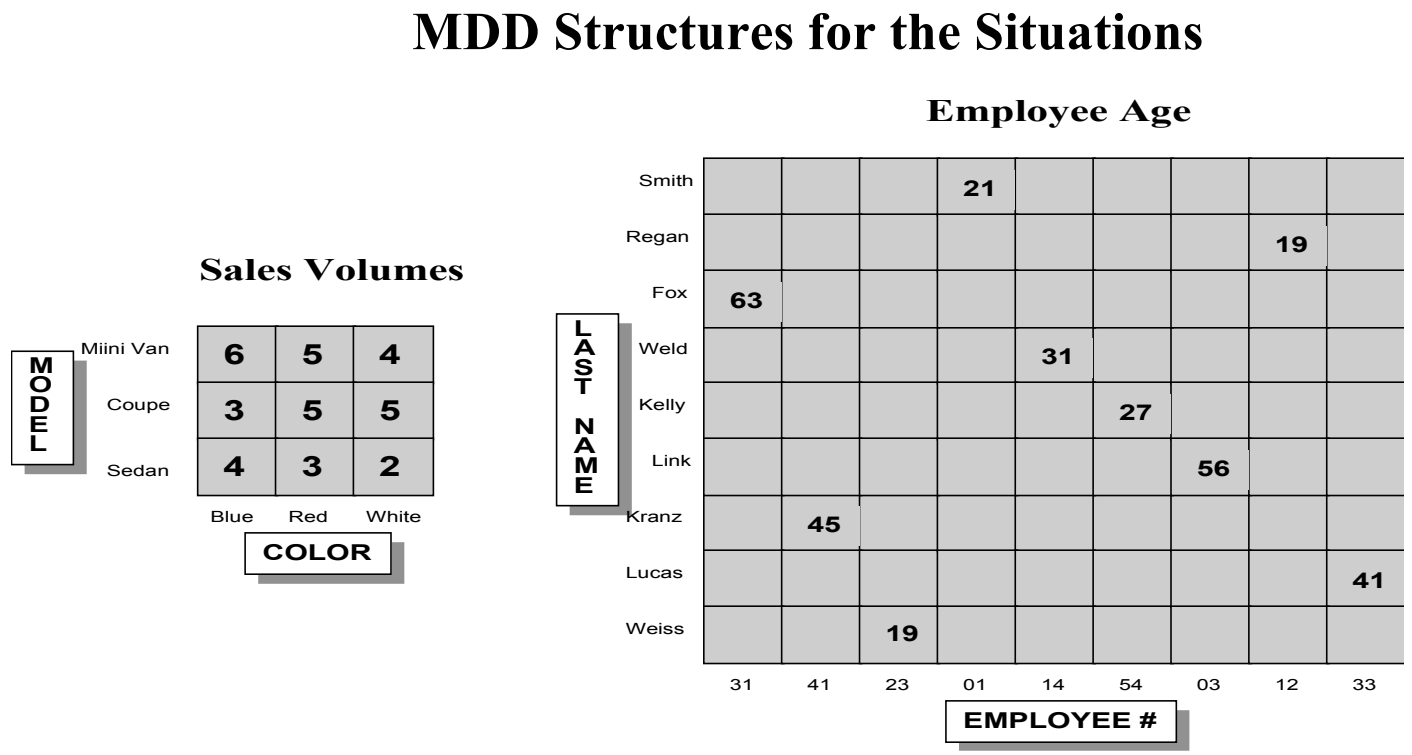
Now consider situation 2

SALES VOLUMES FOR GLEASON DEALERSHIP

MODEL	COLOR	VOLUME
MINI VAN	BLUE	6
MINI VAN	RED	5
MINI VAN	WHITE	4
SPORTS COUPE	BLUE	3
SPORTS COUPE	RED	5
SPORTS COUPE	WHITE	5
SEDAN	BLUE	4
SEDAN	RED	3
SEDAN	WHITE	2

1. Set up a MDD structure for situation 1, with LAST NAME and Employee# as dimensions, and AGE as the measurement.
2. Set up a MDD structure for situation 2, with MODEL and COLOR as dimensions, and SALES VOLUME as the measurement.

When is MDD (In)appropriate?

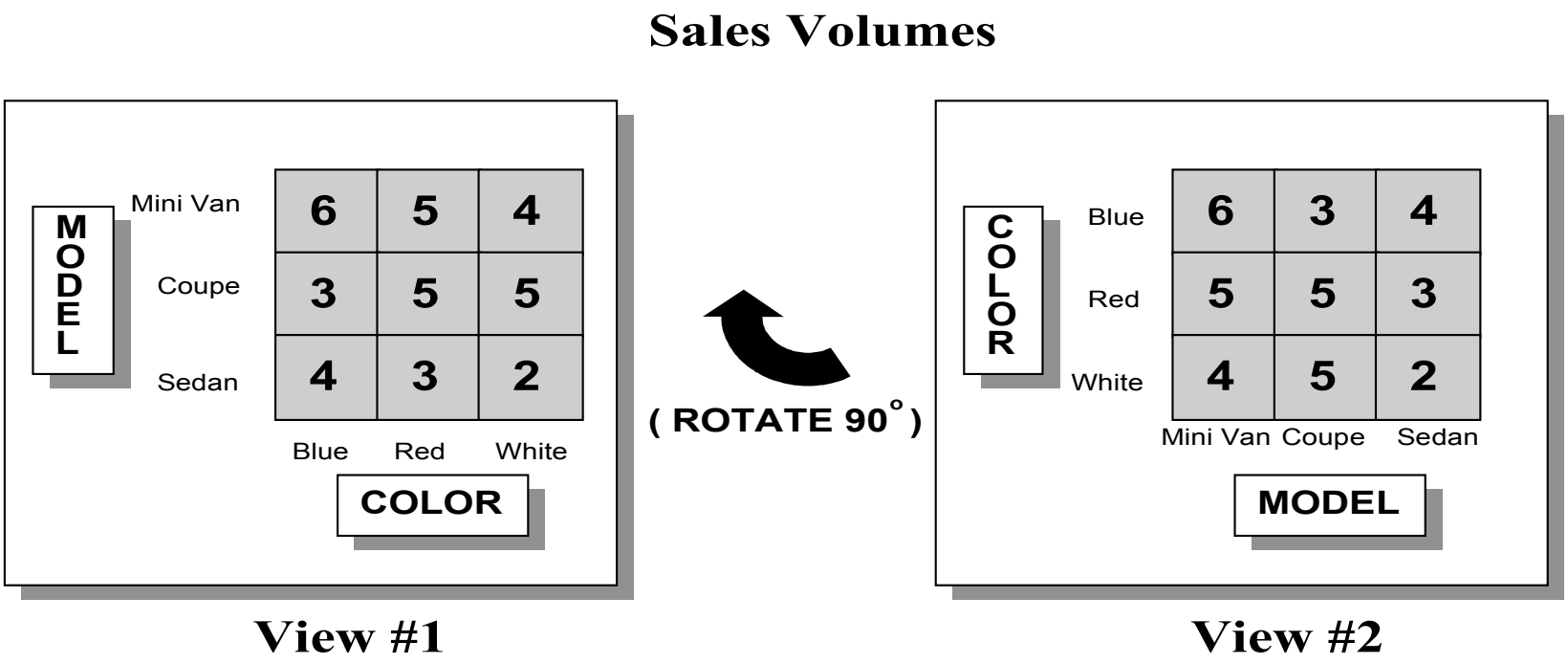


Note the sparseness in the second MDD representation

When is MDD (In)appropriate?

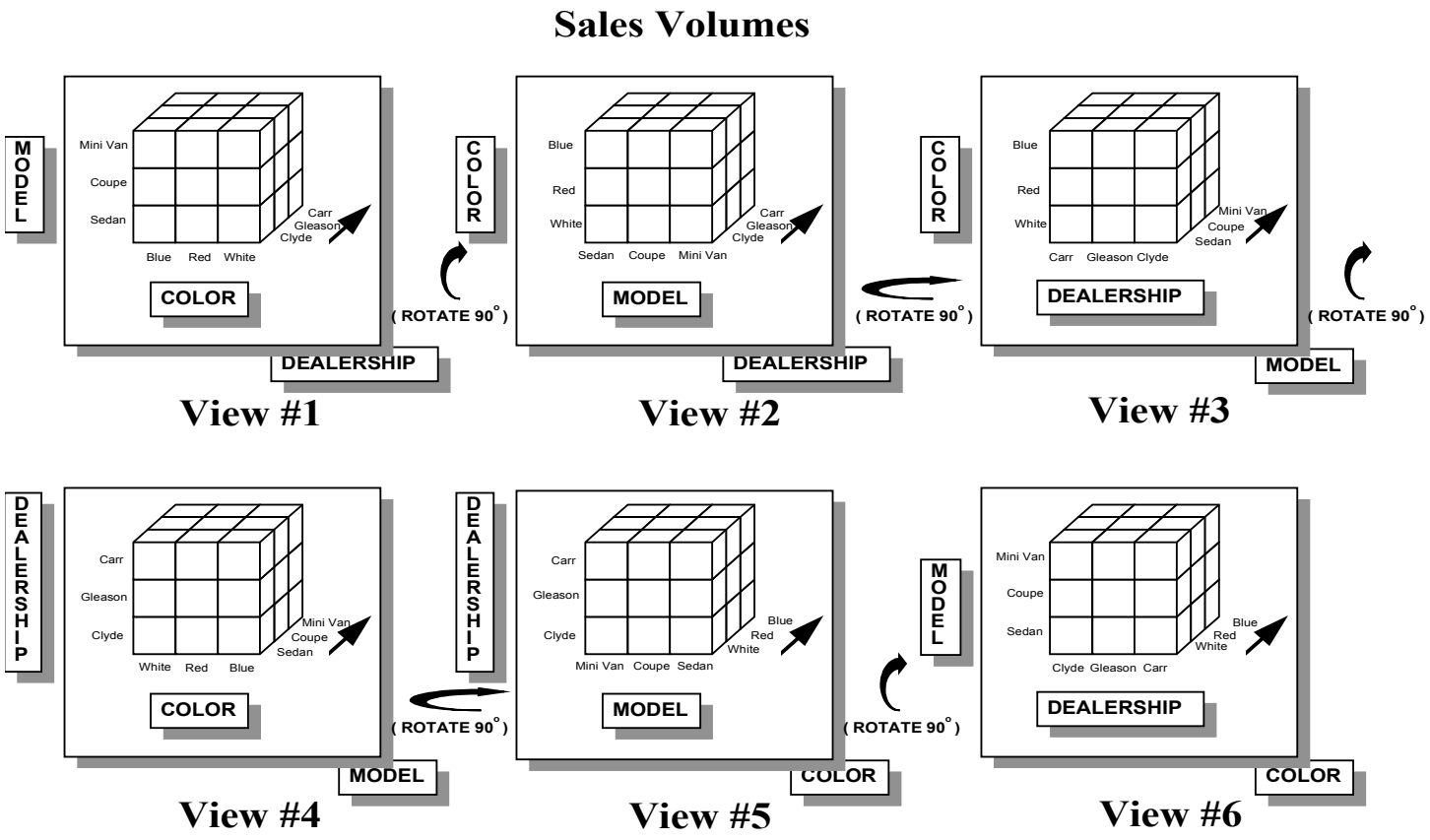
Highly interrelated dataset types be placed in a multidimensional data structure for greatest ease of access and analysis. When there are no interrelationships, the MDD structure is not appropriate.

MDD Features - *Rotation*

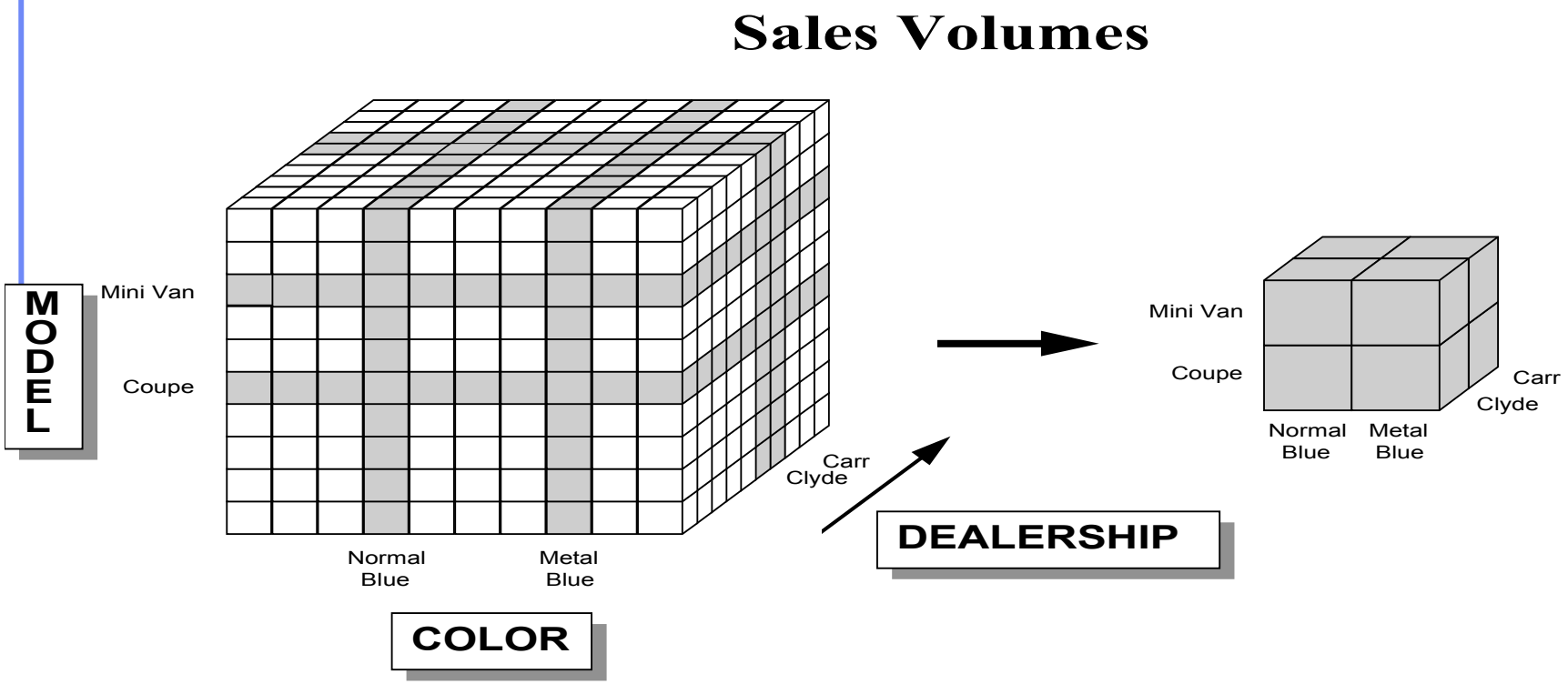


- Also referred to as "data slicing."
- Each rotation yields a different slice or two dimensional table of data – a different face of the cube.

MDD Features - *Rotation*

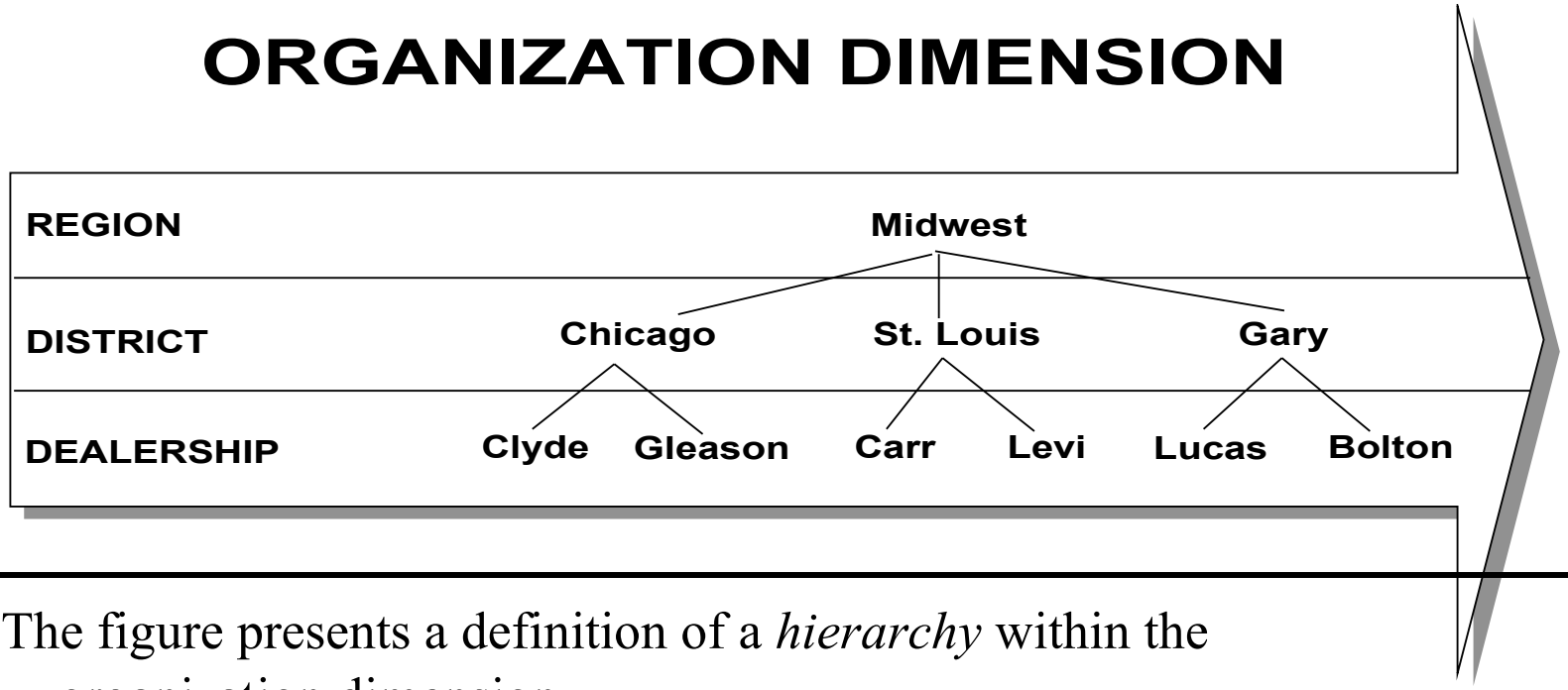


MDD Features - *Ranging*



- The end user selects the desired *positions* along *each dimension*.
- Also referred to as "data dicing."
- The data is scoped down to a subset grouping

MDD Features - *Roll-Ups & Drill Downs*



- The figure presents a definition of a *hierarchy* within the organization dimension.
- Aggregations perceived as being part of the same dimension.
- Moving up and moving down levels in a hierarchy is referred to as "roll-up" and "drill-down."

MDD Features: Multidimensional Computations

- Well equipped to handle demanding mathematical functions.
- Can treat arrays like cells in spreadsheets. For example, in a budget analysis situation, one can divide the ACTUAL array by the BUDGET array to compute the VARIANCE array.
- Applications based on multidimensional database technology typically have one dimension defined as a "business measurements" dimension.
- Integrates computational tools very tightly with the database structure.

The Time Dimension

TIME as a predefined hierarchy for rolling-up and drilling-down across days, weeks, months, years and special periods, such as fiscal years.

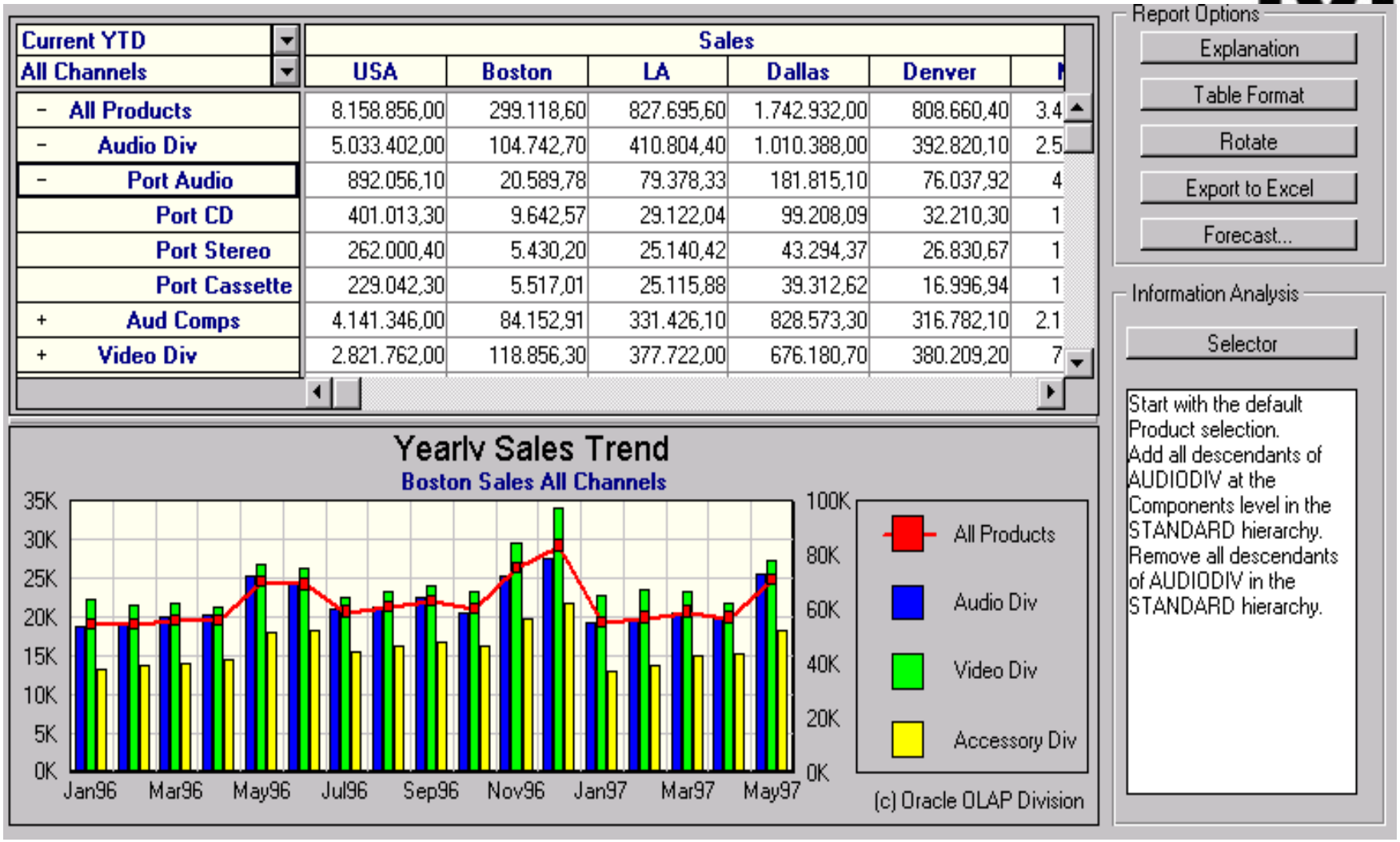
- Eliminates the effort required to build sophisticated hierarchies every time a database is set up.
- Extra performance advantages

Pros/Cons of MDD

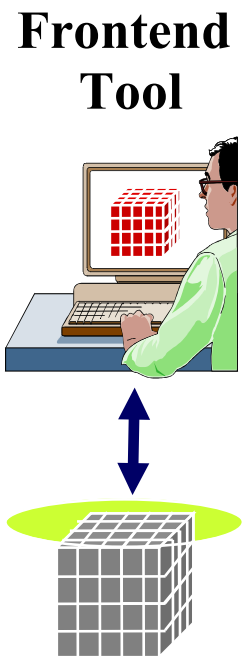
- Cognitive Advantages for the User
- Ease of Data Presentation and Navigation, Time dimension
- Performance

- Less flexible
- Requires greater initial effort

The User's view (OLAP Tool)



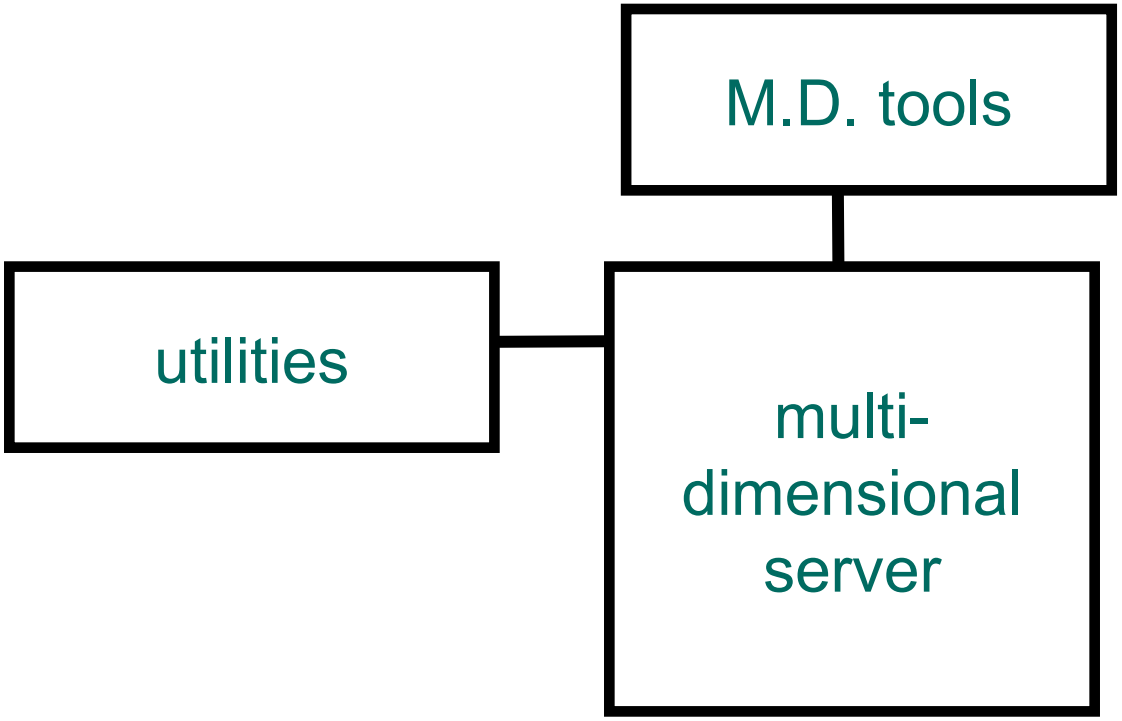
Multidimensional OLAP (MOLAP)

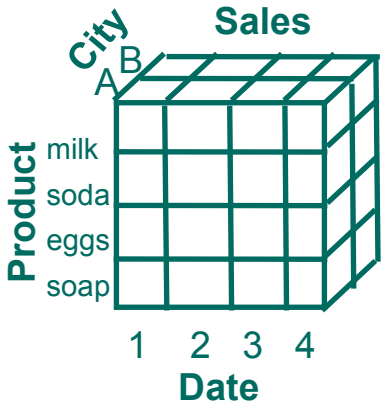


- specialized database technology
- multidimensional storage structures
- E.g. Hyperion Essbase, Oracle Express, Cognos PowerPlay (Server)
- Query Performance
- Powerful MD Model
  - write access
- Database Features
  - multiuser access/ backup and recovery
- Sparsity Handling -> DB Explosion

MOLAP Server

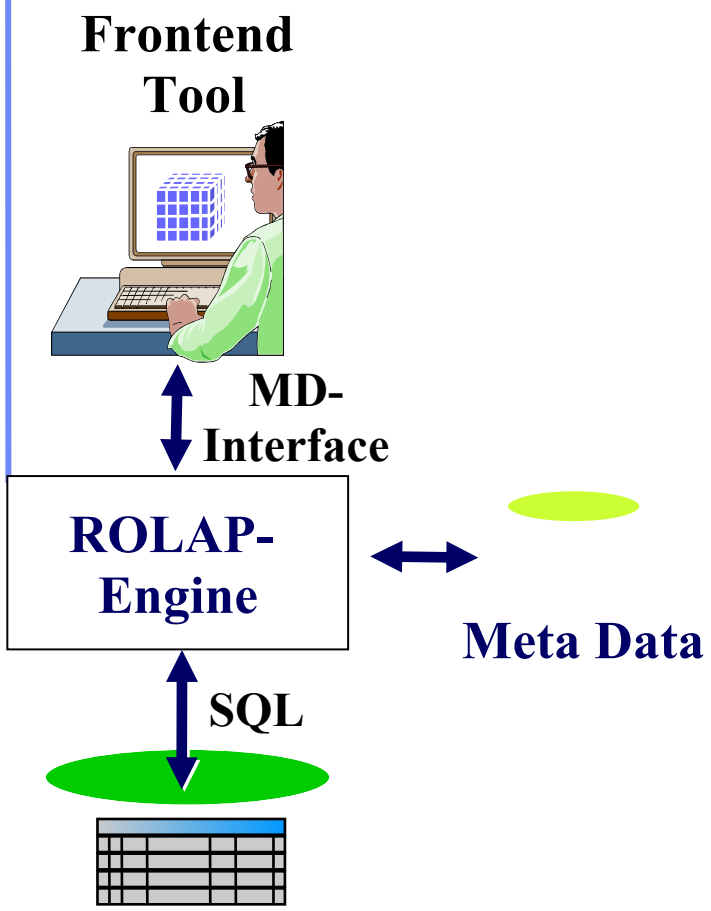
- Multi-Dimensional OLAP Server





could also sit on relational DBMS

Relational OLAP (ROLAP)

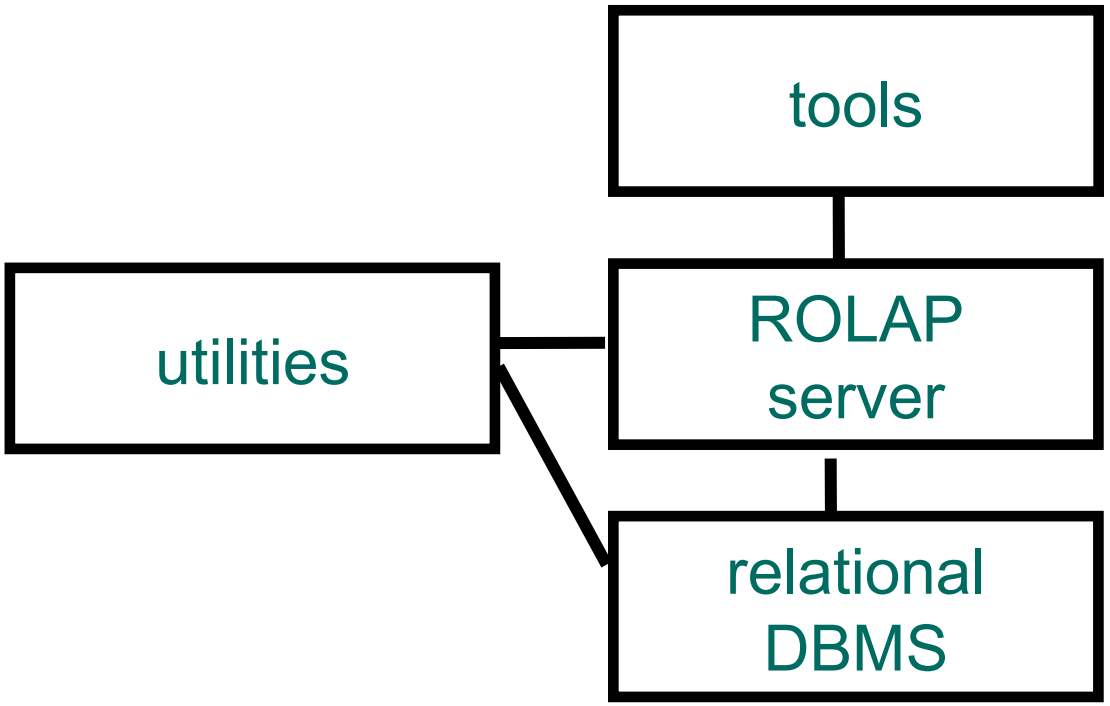


- idea: use relational data storage
- star (snowflake) schema
- E.g. Microstrategy, SAP BW
- \+ advantages of RDBMS
  - \+ scalability, reliability, security etc.
- Sparsity handling
- Query Performance
- Data Model Complexity
- no write access

ROLAP Server

- Relational OLAP Server

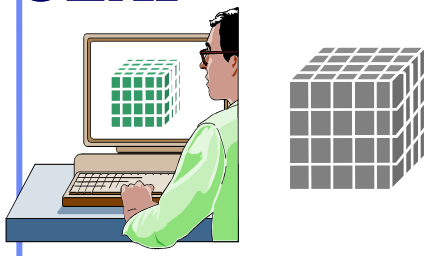
sale	prodId	date	sum
	p1	1	62
	p2	1	19
	p1	2	48



Special indices, tuning;
Schema is “denormalized”

Client (Desktop) OLAP

Client-OLAP



- proprietary data structure on the client
- data stored as file
- mostly RAM based architectures
- E.g. Business Objects, Cognos PowerPlay

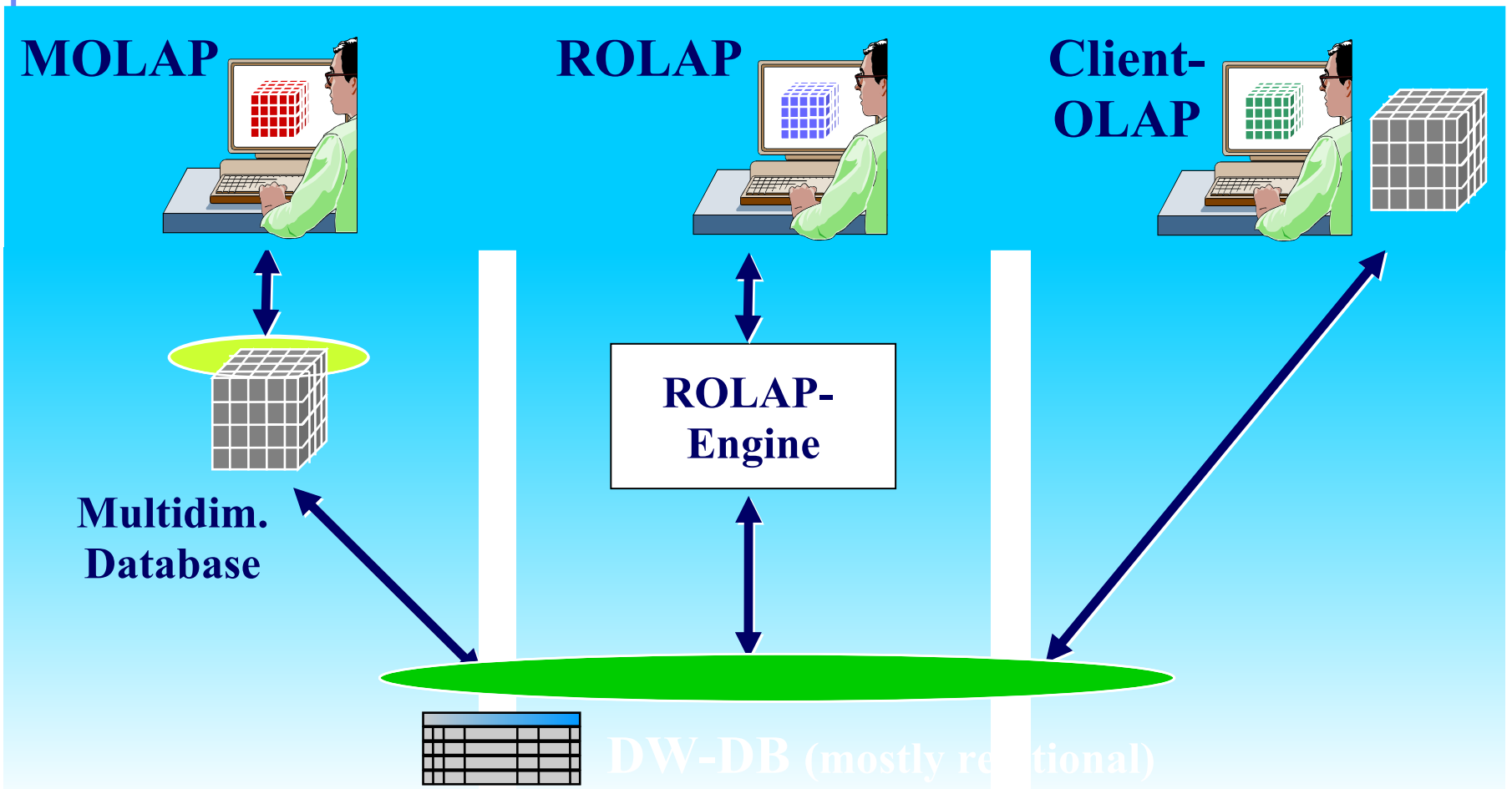
\+ mobile user

\+ ease of installation and use

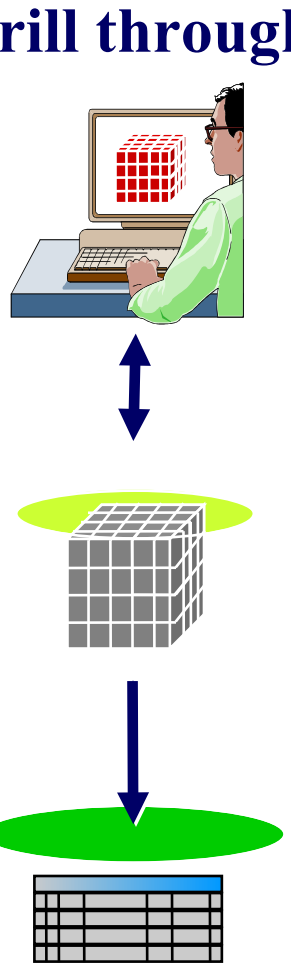
▭ data volume

▭ no multiuser capabilites

DW Integration



Combining Architectures I

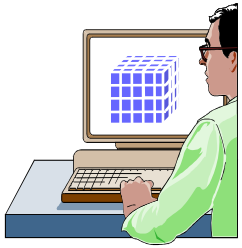


- highly aggregated data
- dense data
- 95% of the analysis requirements

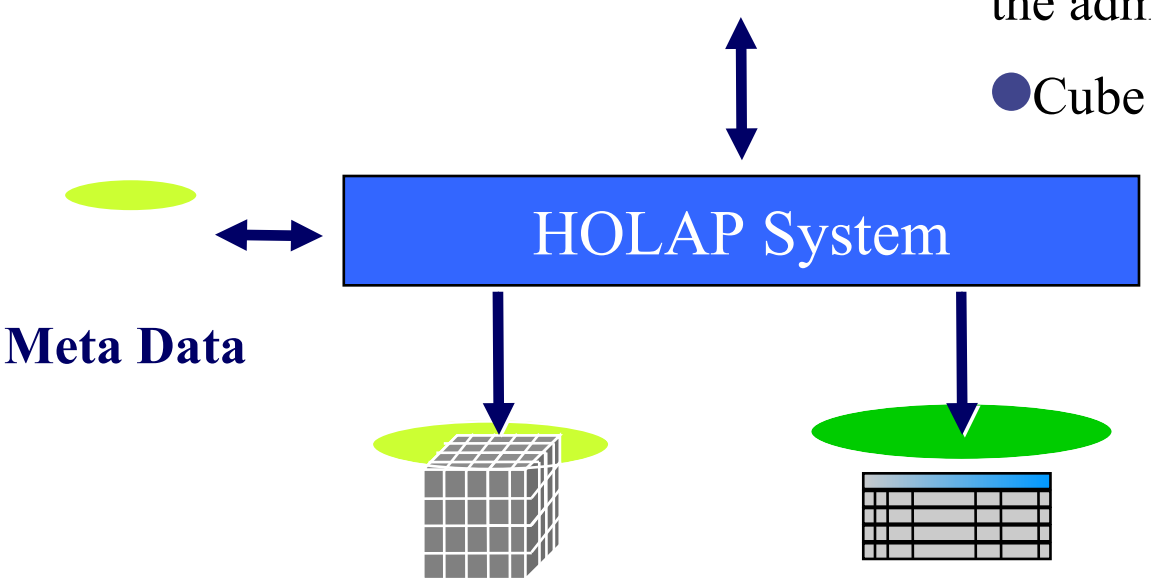
- detailed data (sparse)
- 5% of the requirements

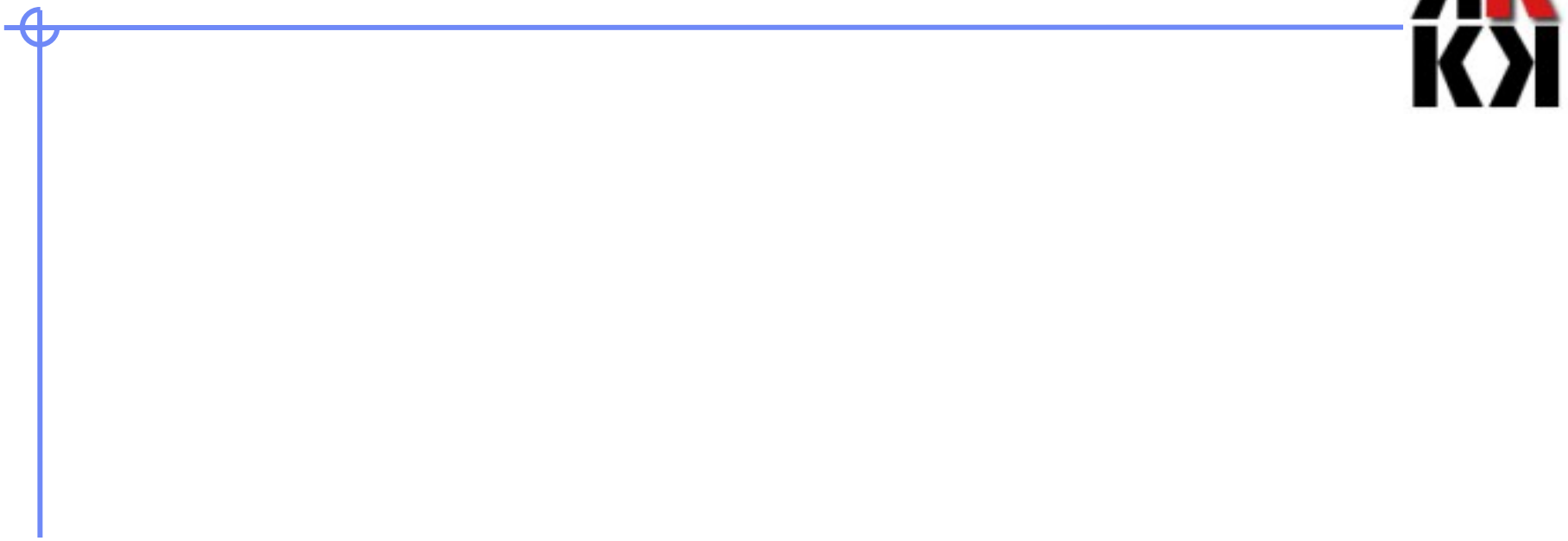
Combining Architectures II

Hybrid OLAP (HOLAP)



- equal treatment of MD and Rel Data
- Storage type at the discretion of the administrator
- Cube Partitioning



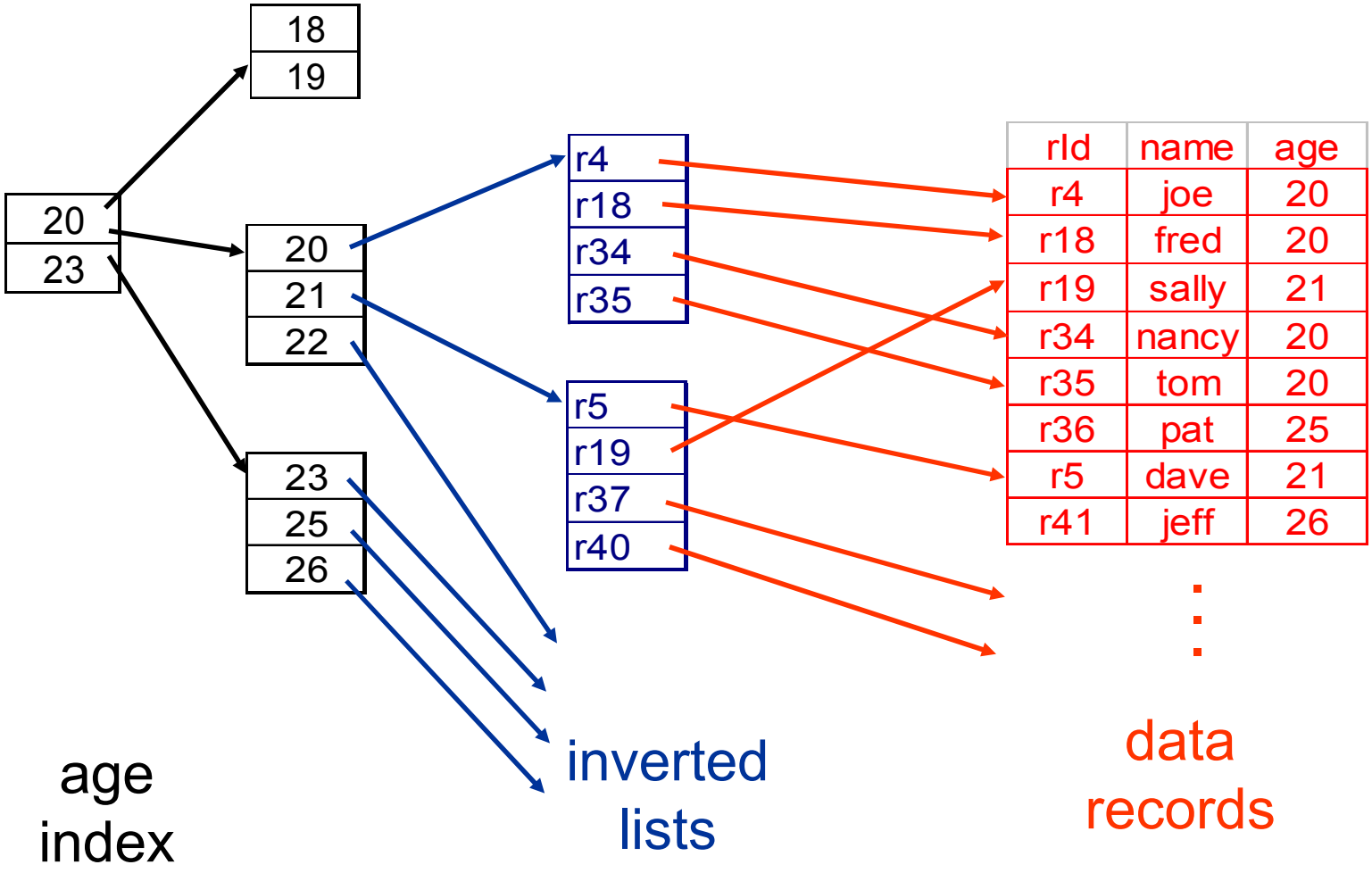


SURPLUS SLIDES

Index Structures

- Traditional Access Methods
  - B-trees, hash tables, R-trees, grids, …
- Popular in Warehouses
  - inverted lists
  - bit map indexes
  - join indexes
  - text indexes

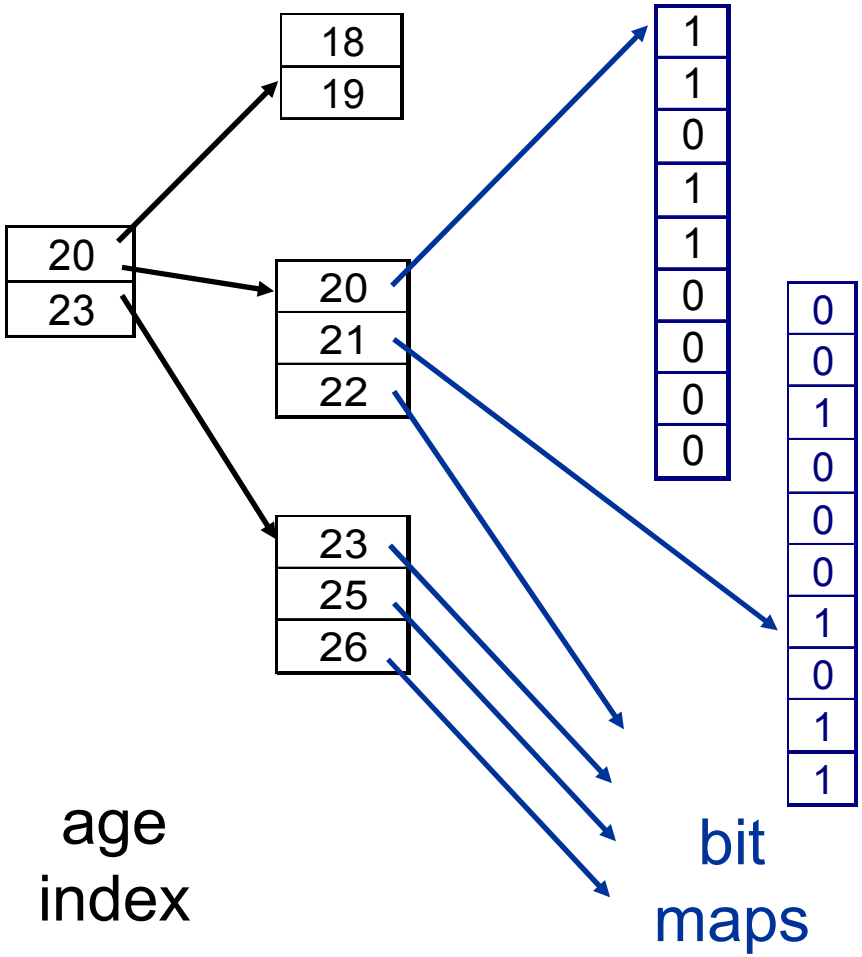
Inverted Lists



Using Inverted Lists

- Query:
  - Get people with age = 20 and name = “fred”
- List for age = 20: r4, r18, r34, r35
- List for name = “fred”: r18, r52
- Answer is intersection: r18

Bit Maps



id	name	age
1	joe	20
2	fred	20
3	sally	21
4	nancy	20
5	tom	20
6	pat	25
7	dave	21
8	jeff	26

data
records

Using Bit Maps

- Query:
  - Get people with age = 20 and name = "fred"
- List for age = 20: 1101100000
- List for name = "fred": 0100000001
- Answer is intersection: 0100000000000

- Good if domain cardinality small
- Bit vectors can be compressed

Join

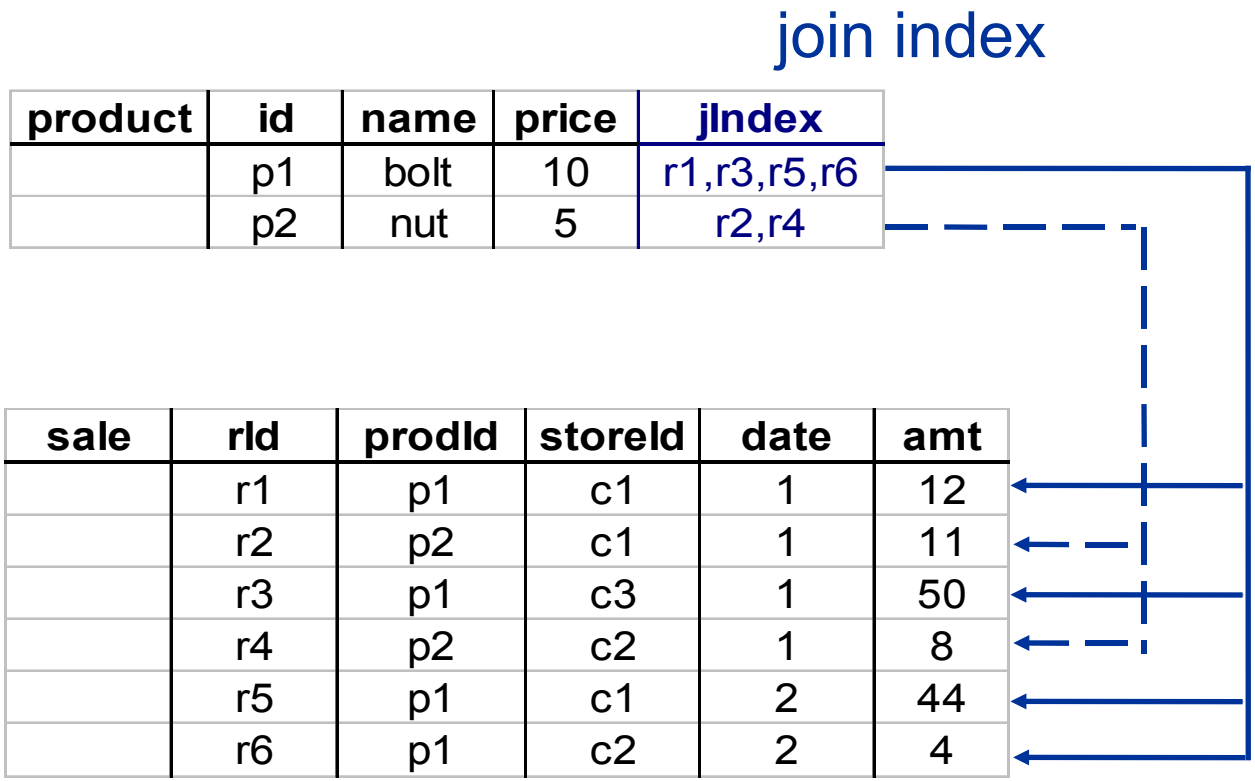
- "Combine" SALE, PRODUCT relations
- In SQL: SELECT * FROM SALE, PRODUCT

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

product	id	name	price
	p1	bolt	10
	p2	nut	5

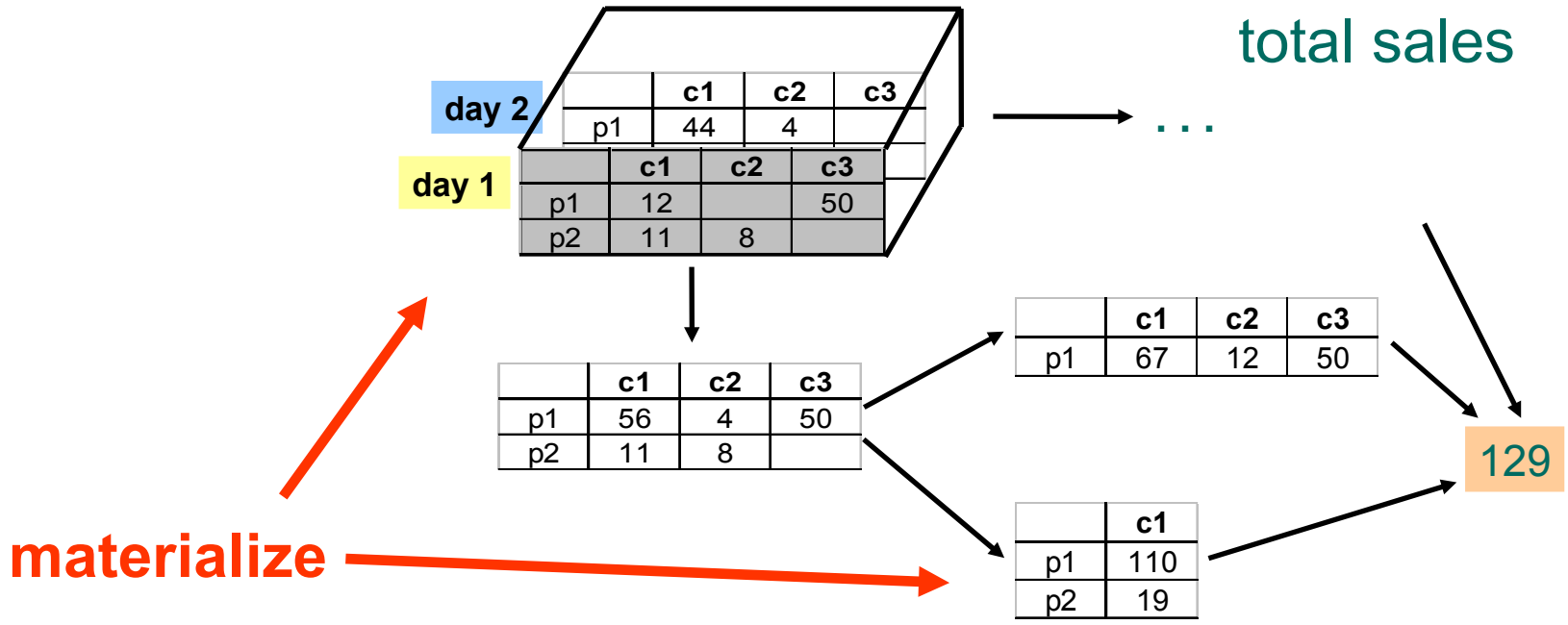
joinTb	prodId	name	price	storeId	date	amt
	p1	bolt	10	c1	1	12
	p2	nut	5	c1	1	11
	p1	bolt	10	c3	1	50
	p2	nut	5	c2	1	8
	p1	bolt	10	c1	2	44
	p1	bolt	10	c2	2	4

Join Indexes



What to Materialize?

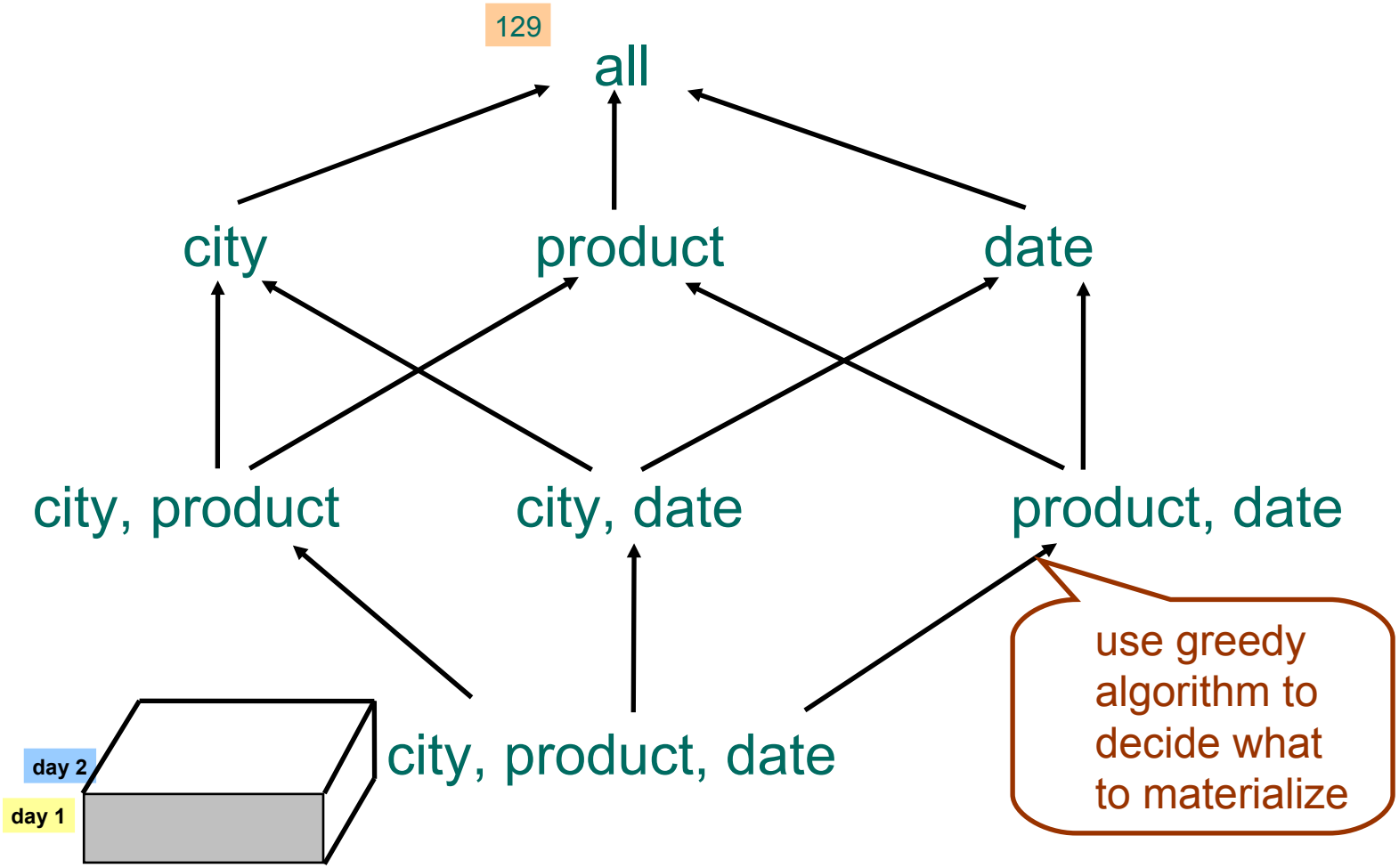
- Store in warehouse results useful for common queries
- Example:



Materialization Factors

- Type/frequency of queries
- Query response time
- Storage cost
- Update cost

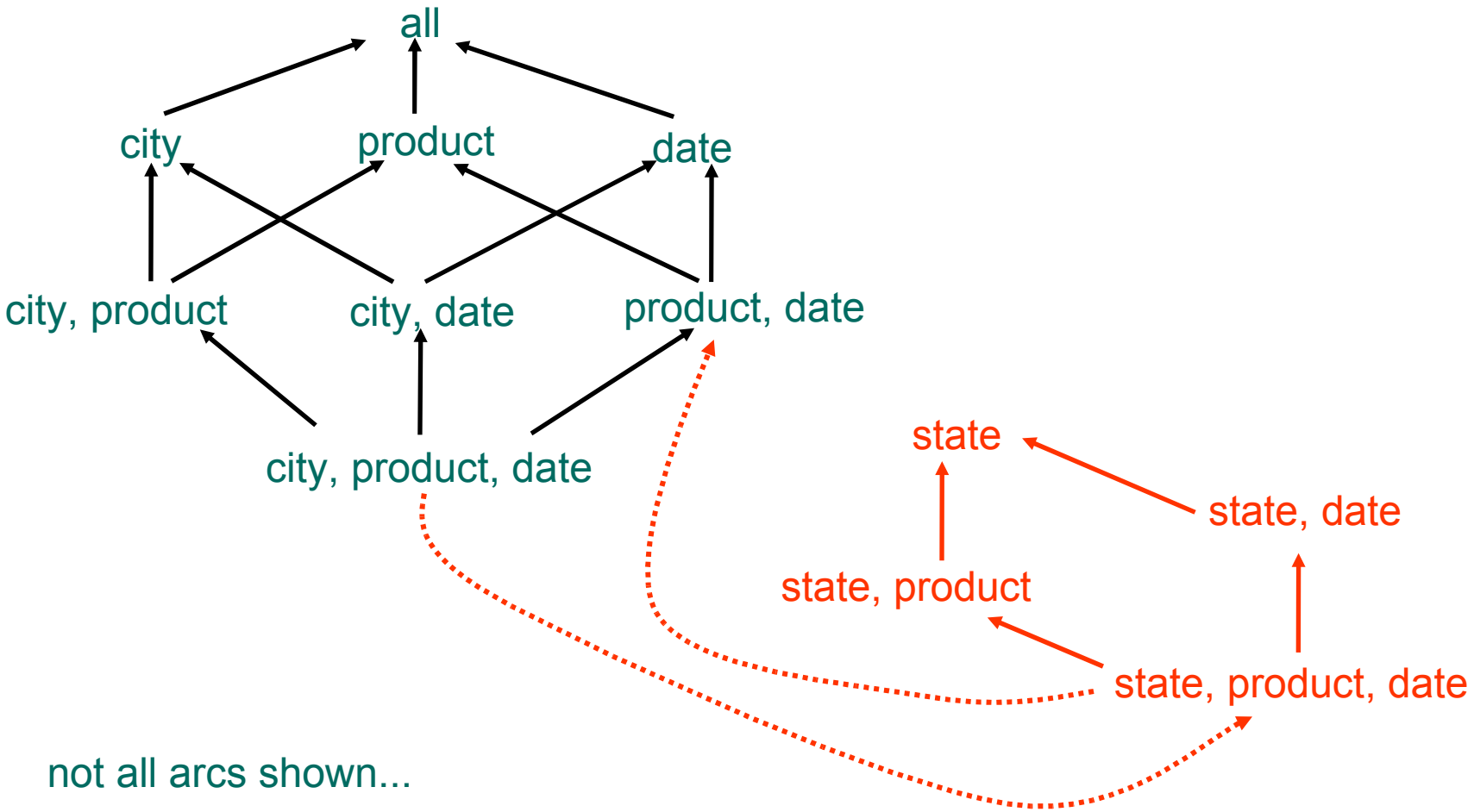
Cube Aggregates Lattice



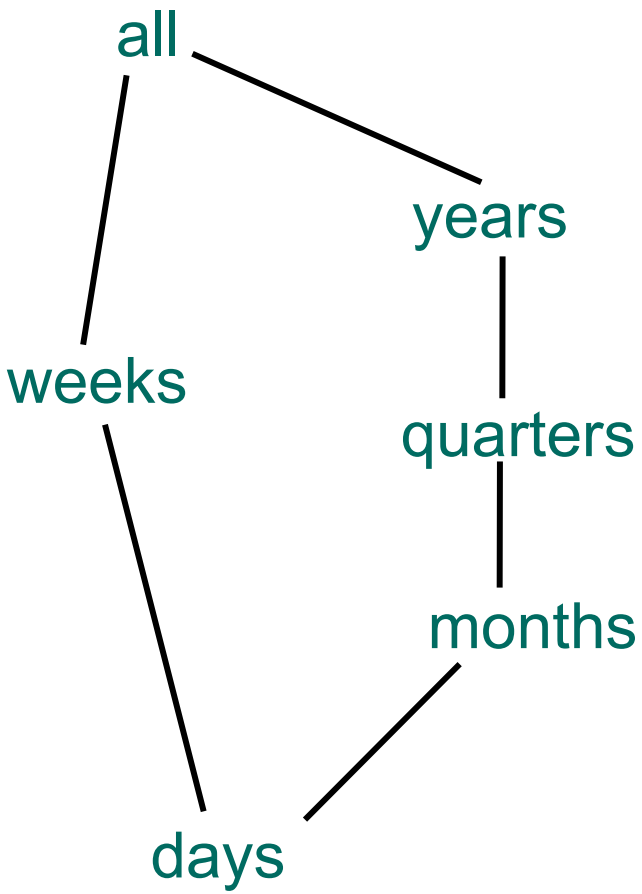
Dimension Hierarchies

cities	city	state
	c1	CA
	c2	NY

Dimension Hierarchies



Interesting Hierarchy



time	day	week	month	quarter	year
	1	1	1	1	2000
	2	1	1	1	2000
	3	1	1	1	2000
	4	1	1	1	2000
	5	1	1	1	2000
	6	1	1	1	2000
	7	1	1	1	2000
	8	2	1	1	2000

conceptual dimension table

Design

- What data is needed?
- Where does it come from?
- How to clean data?
- How to represent in warehouse (schema)?
- What to summarize?
- What to materialize?
- What to index?

Tools

- Development
  - design & edit: schemas, views, scripts, rules, queries, reports
- Planning & Analysis
  - what-if scenarios (schema changes, refresh rates), capacity planning
- Warehouse Management
  - performance monitoring, usage patterns, exception reporting
- System & Network Management
  - measure traffic (sources, warehouse, clients)
- Workflow Management
  - "reliable scripts" for cleaning & analyzing data

Current State of Industry

- Extraction and integration done off-line
  - Usually in large, time-consuming, batches
- Everything copied at warehouse
  - Not selective about what is stored
  - Query benefit vs storage & update cost
- Query optimization aimed at OLTP
  - High throughput instead of fast response
  - Process whole query before displaying anything