Dimensional Modelling

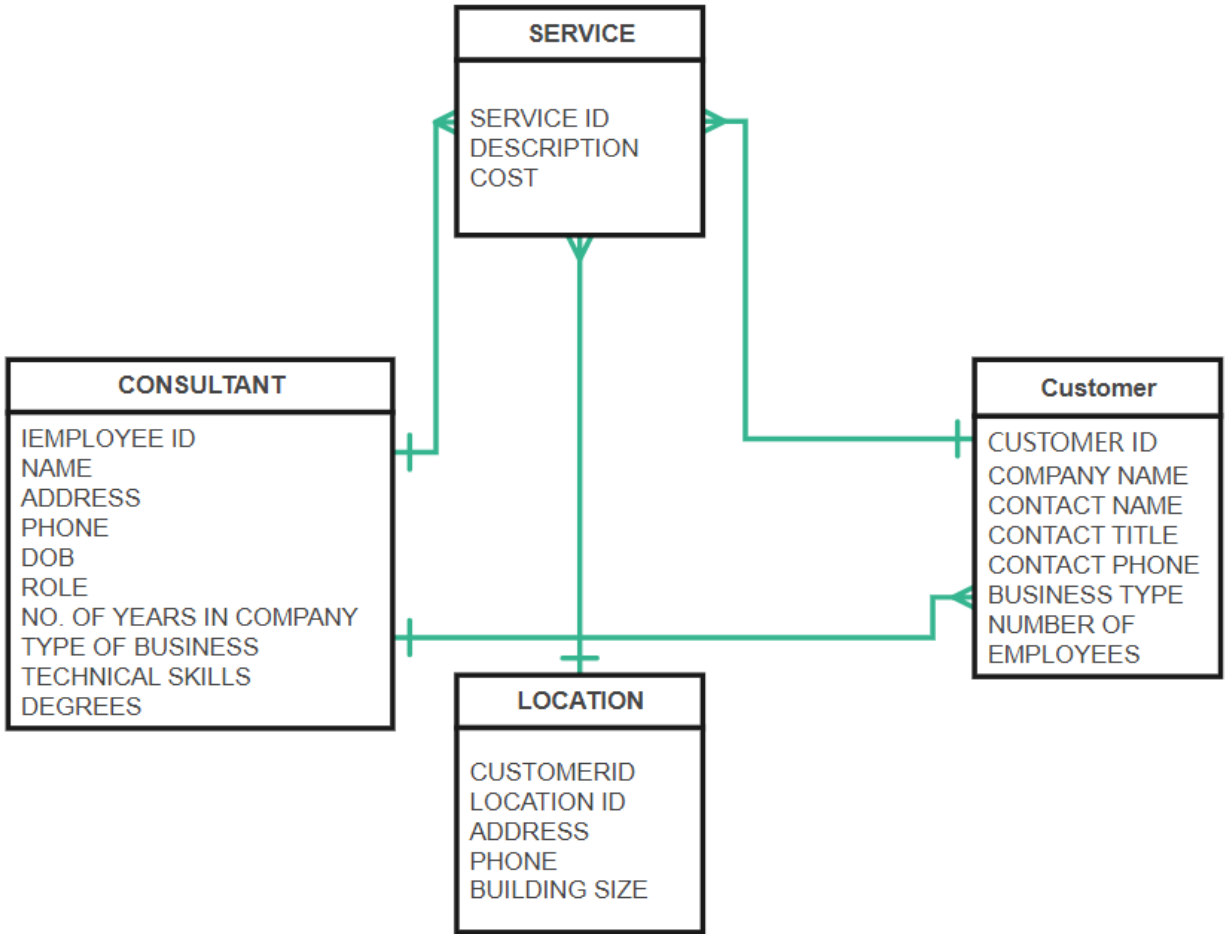
Dimensional modeling is widely accepted as the preferred technique for presenting analytic data.

It addresses two simultaneous requirements:

- Deliver data that's understandable to the business users.
- Deliver fast query performance.

Dimensional modeling is a technique for making databases simple.

Remember E-R Diagrams?



Warehouse Models & Operators

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

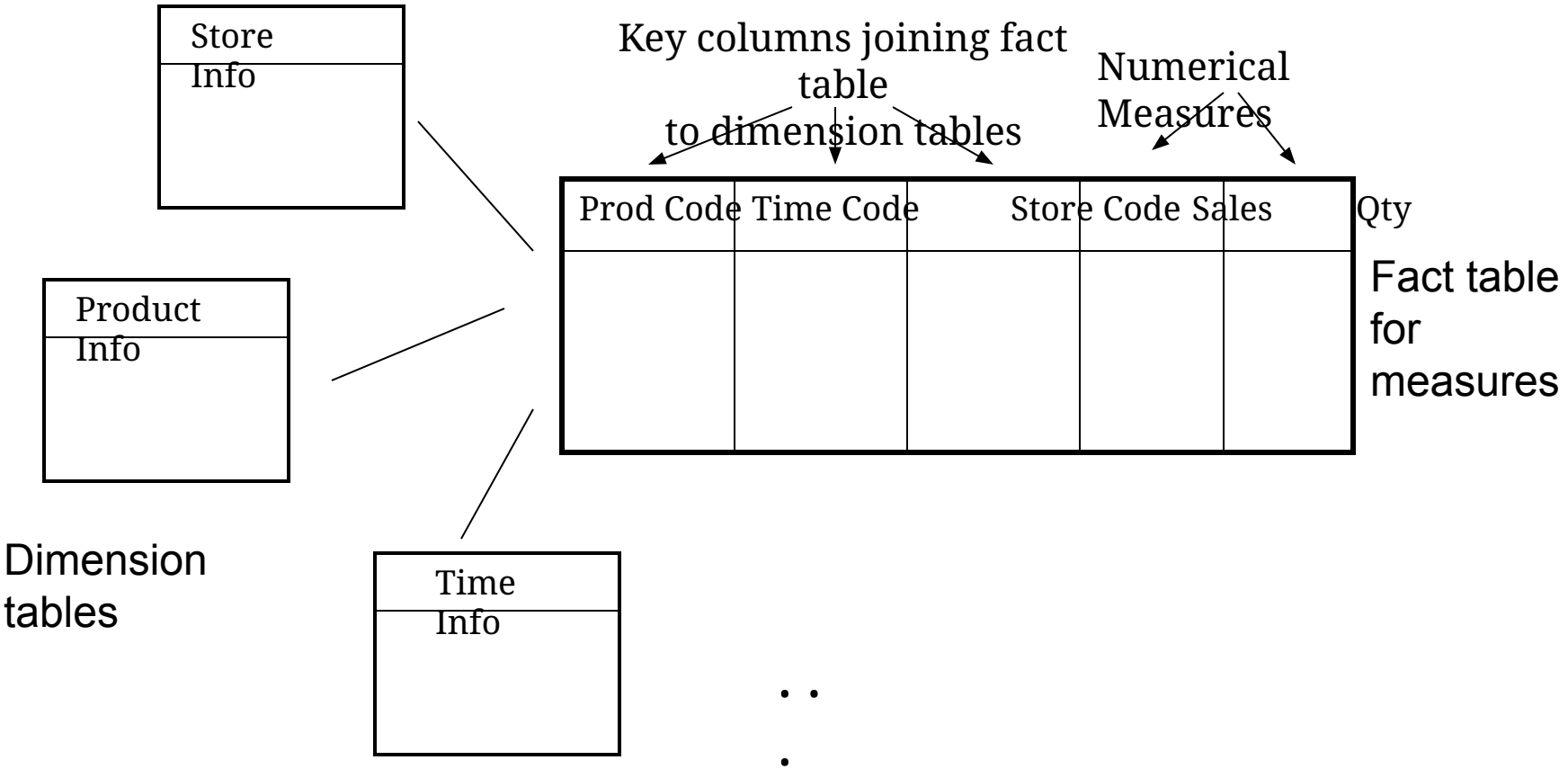
Multi-Dimensional Data

- Facts or Measures - numerical (and additive) data being tracked in business, can be analyzed and examined.
- Attributes or Dimensions - business parameters that define a transaction, relatively static data such as lookup or reference tables
- Example: Analyst may want to view ***sales*** data (measure) by *geography*, by *time*, and by *product* (dimensions)

The Multi-Dimensional Model

"Sales by product line over the past six months"

"Sales by store between 2020 and 2024"

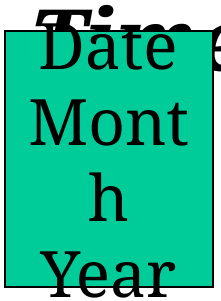


Multidimensional Modeling

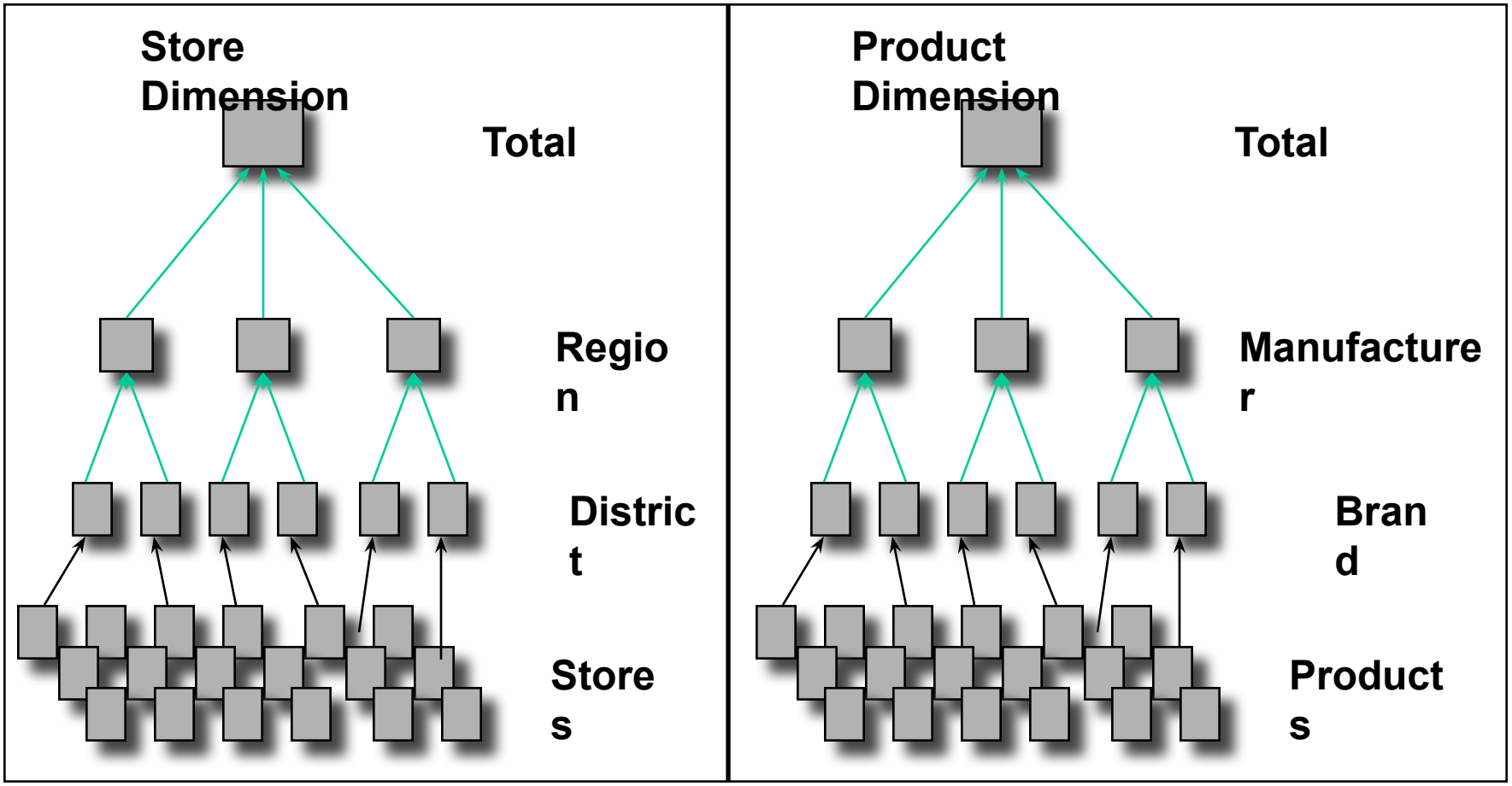
- Multidimensional modeling is a technique for structuring data around the business concepts
- ER models describe “entities” and “relationships”
- Multidimensional models describe “measures” and “dimensions”

Dimensional Modeling

- Dimensions are organized into hierarchies
  - E.g., Time dimension: days → weeks → quarters
  - E.g., Product dimension: product → product line → brand
  - E.g. Geography dimension: Region → Country → State → City
- Dimensions have attributes



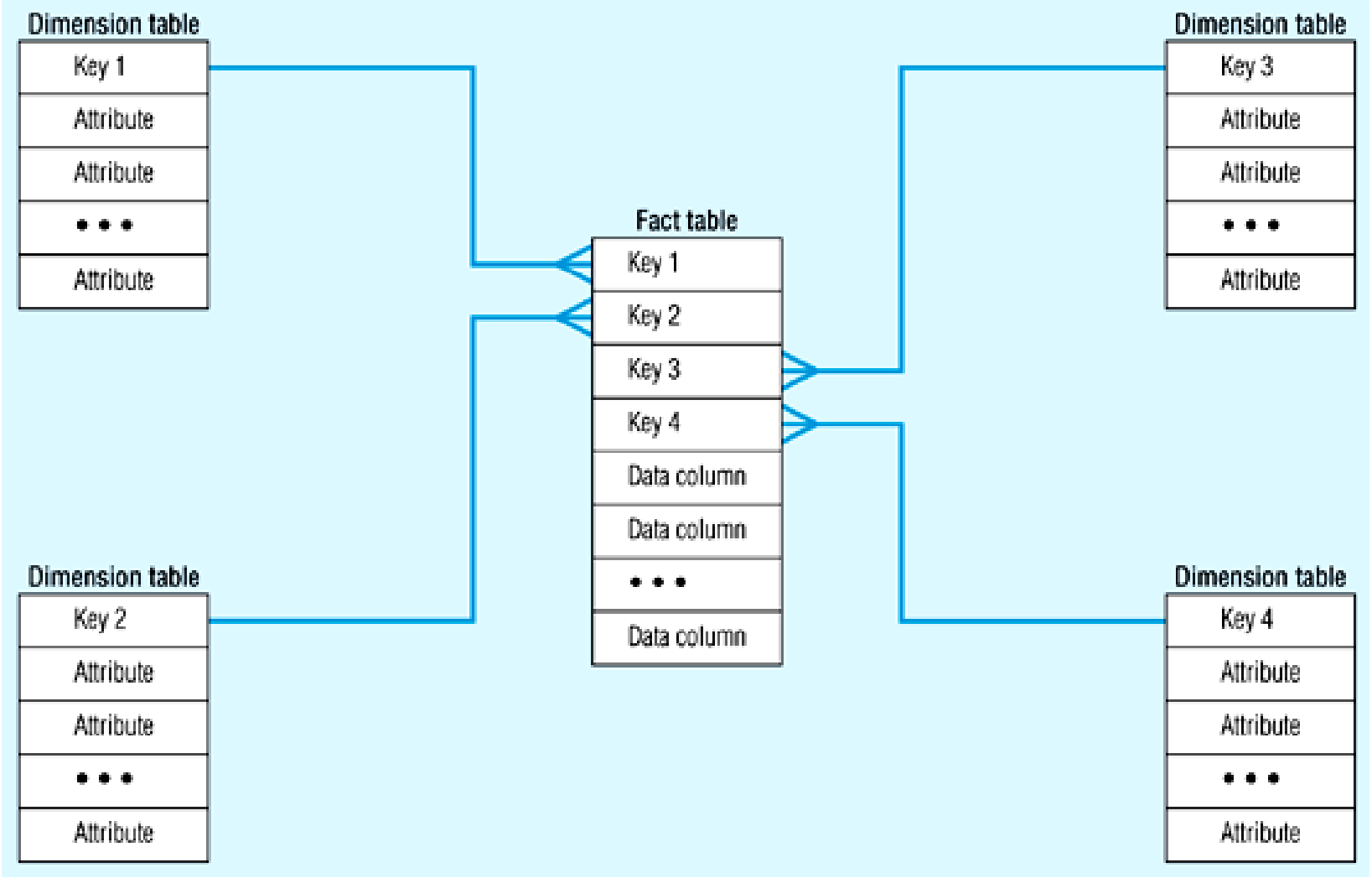
Dimension Hierarchies



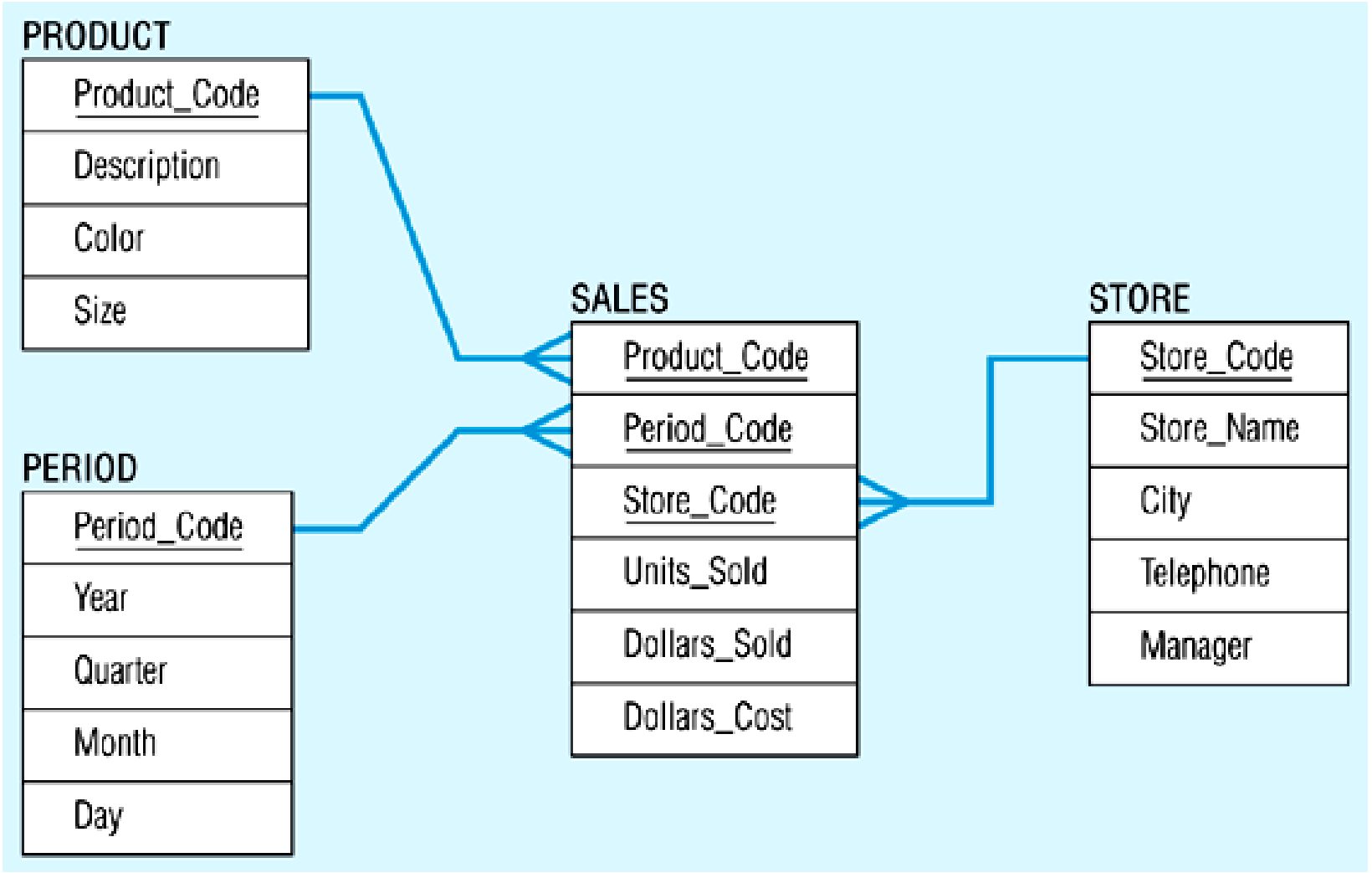
Schema Design

- Most data warehouses use a star schema to represent multi-dimensional model.
- Each dimension is represented by a **dimension table** that describes it.
- A **fact table** connects to all dimension tables with a multiple join. Each tuple in the fact table consists of a pointer to each of the dimension tables that provide its multi-dimensional coordinates and stores measures for those coordinates.
- The links between the fact table in the center and the dimension tables in the extremities form a shape like a star.

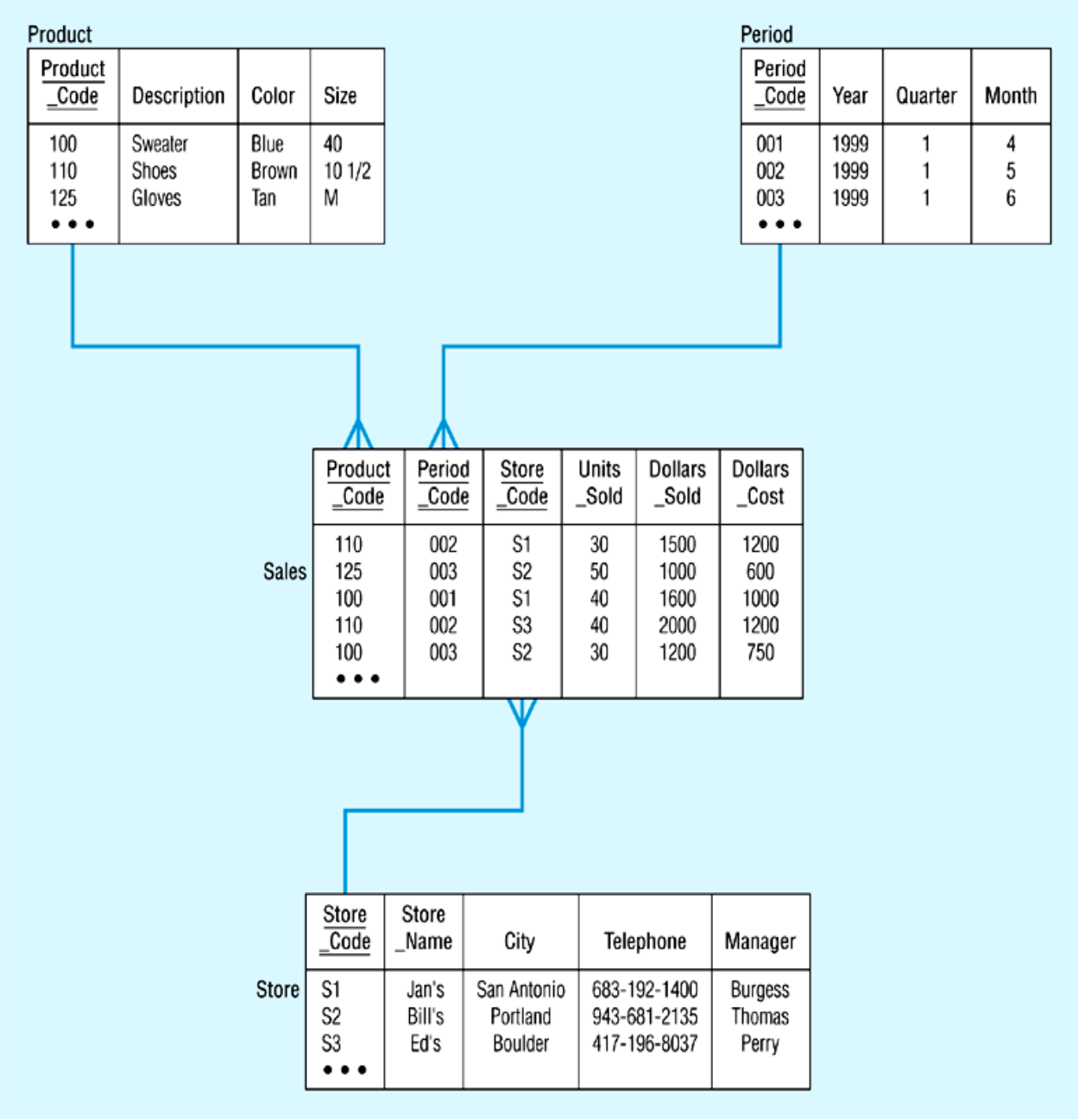
Star Schema (in RDBMS)



Star Schema Example



Star Schema with Sample Data



The “Classic” Star Schema

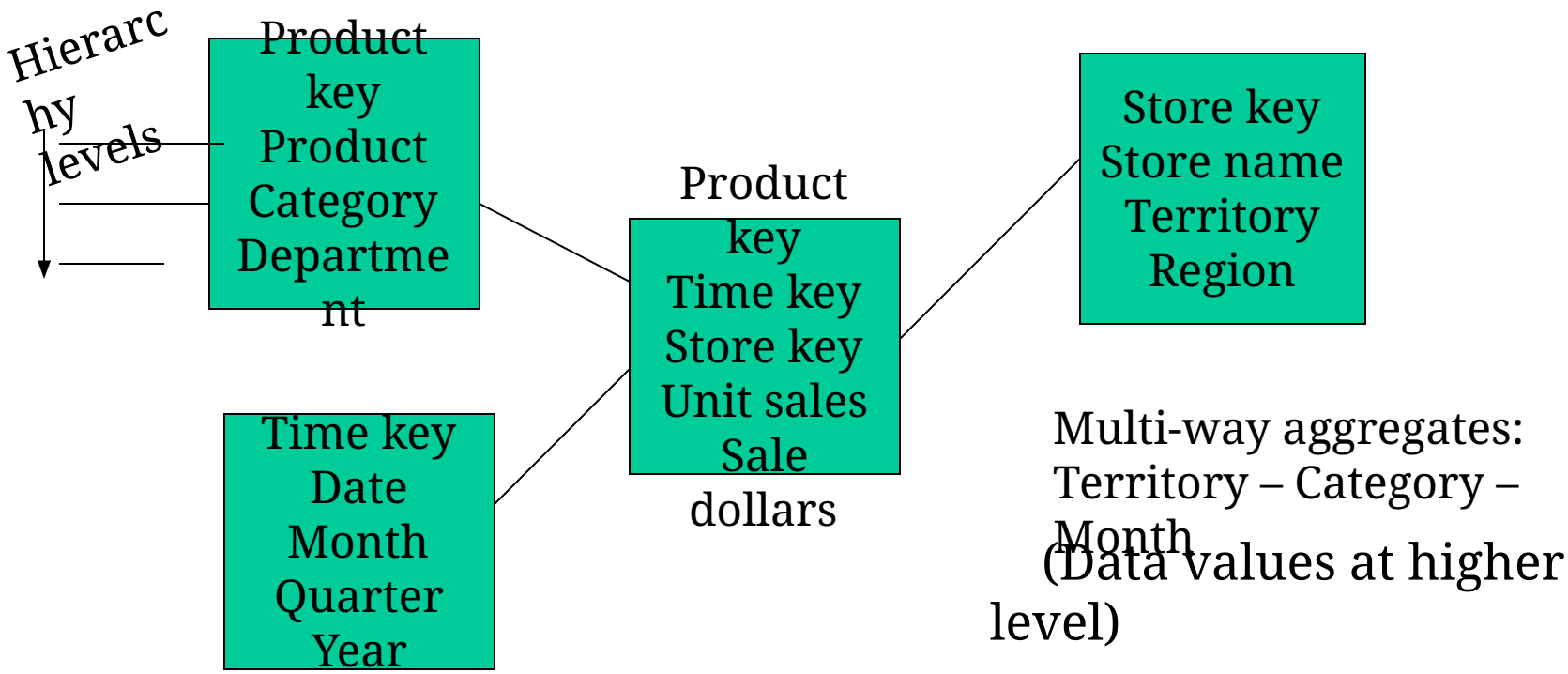
- A relational model with a one-to-many relationship between dimension table and fact table.
- A single fact table, with detail and summary data
- Fact table primary key has only one key column per dimension
- Each dimension is a single table, highly denormalized
- **Benefits**: Easy to understand, intuitive mapping between the business entities, easy to define hierarchies, reduces # of physical joins, low maintenance, very simple metadata
- **Drawbacks**: Summary data in the fact table yields

Need for Aggregates

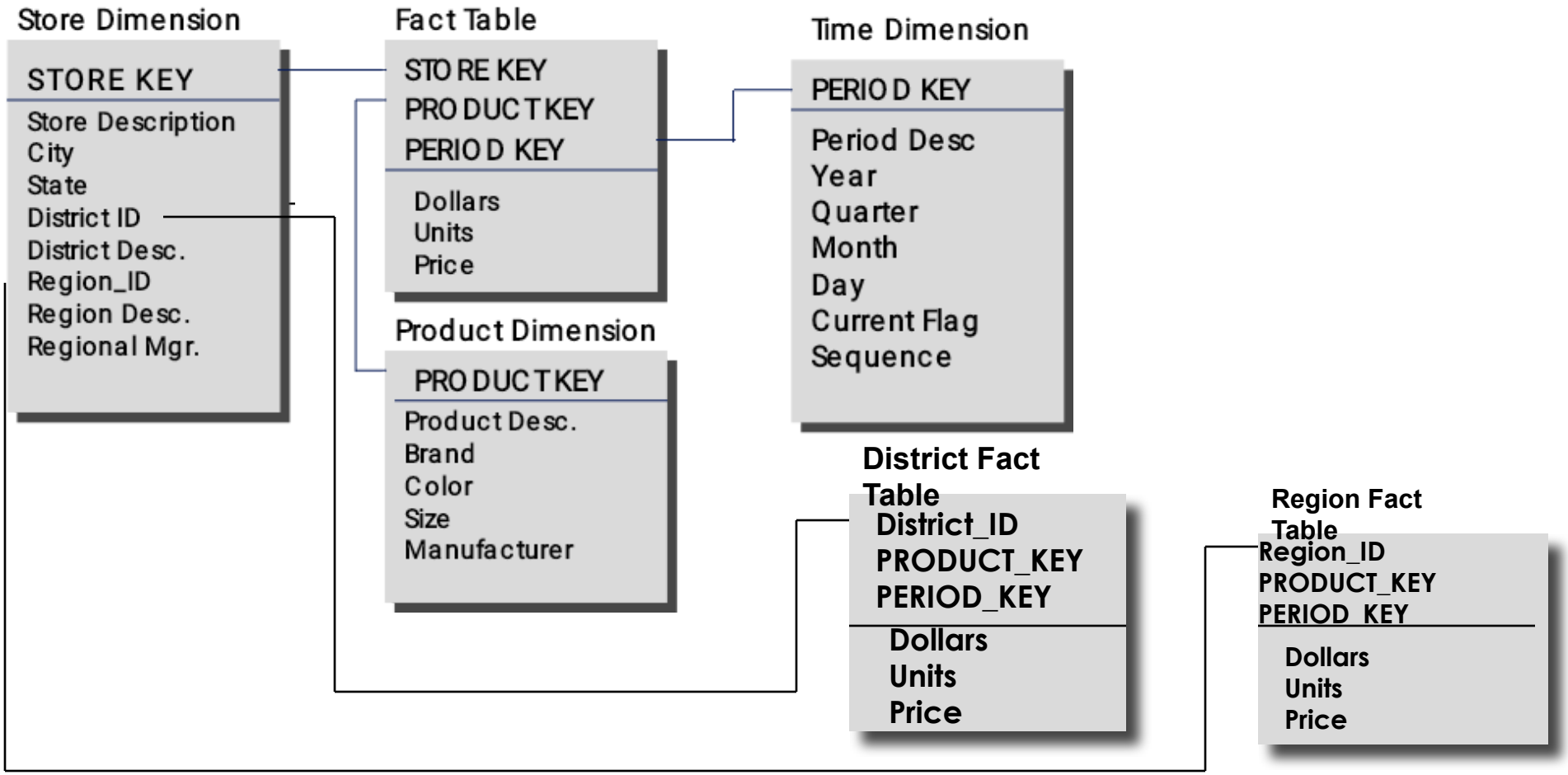
- Sizes of typical tables:
  - Time dimension: 5 years x 365 days = 1825
  - Store dimension: 300 stores reporting daily sales
  - Production dimension: 40,000 products in each store (about 4000 sell in each store daily)
  - Maximum number of base fact table records: 2 billion (lowest level of detail)
- A query involving 1 brand, all store, 1 year: retrieve/summarize over 7 million fact table rows.

Aggregating Fact Tables

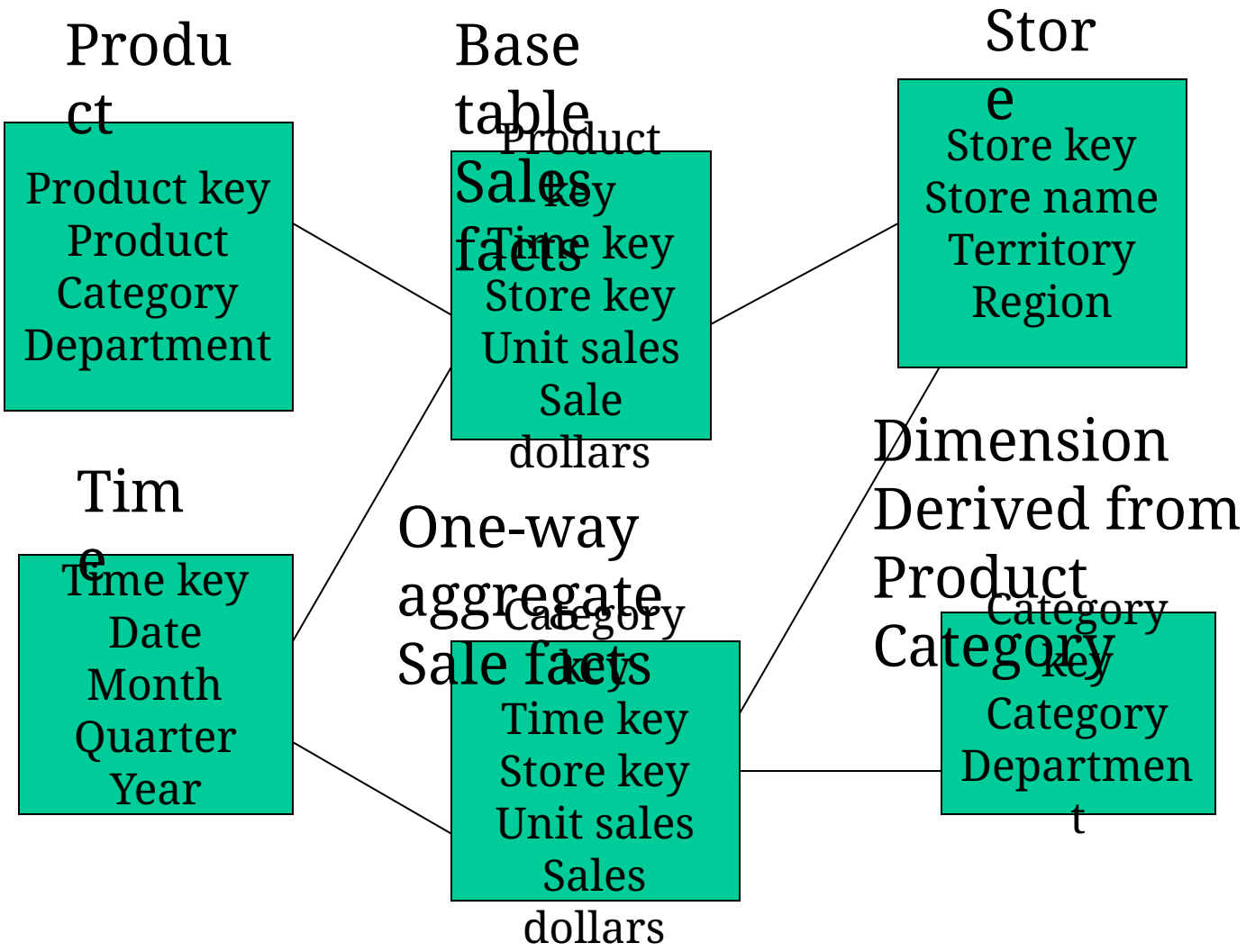
- Aggregate fact tables are summaries of the most granular data at higher levels along the dimension hierarchies.



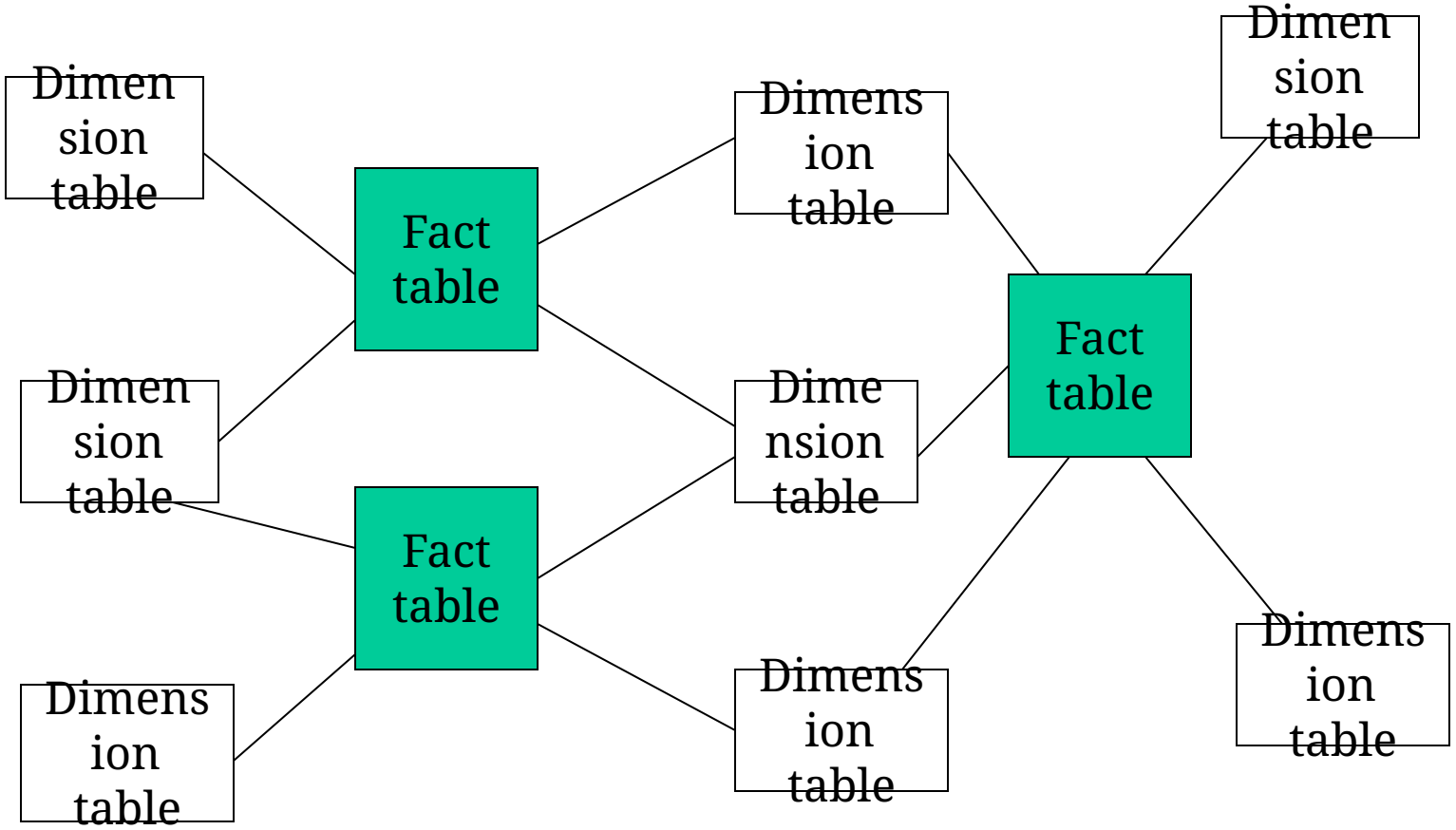
The "Fact Constellation" Schema



Aggregate Fact Tables



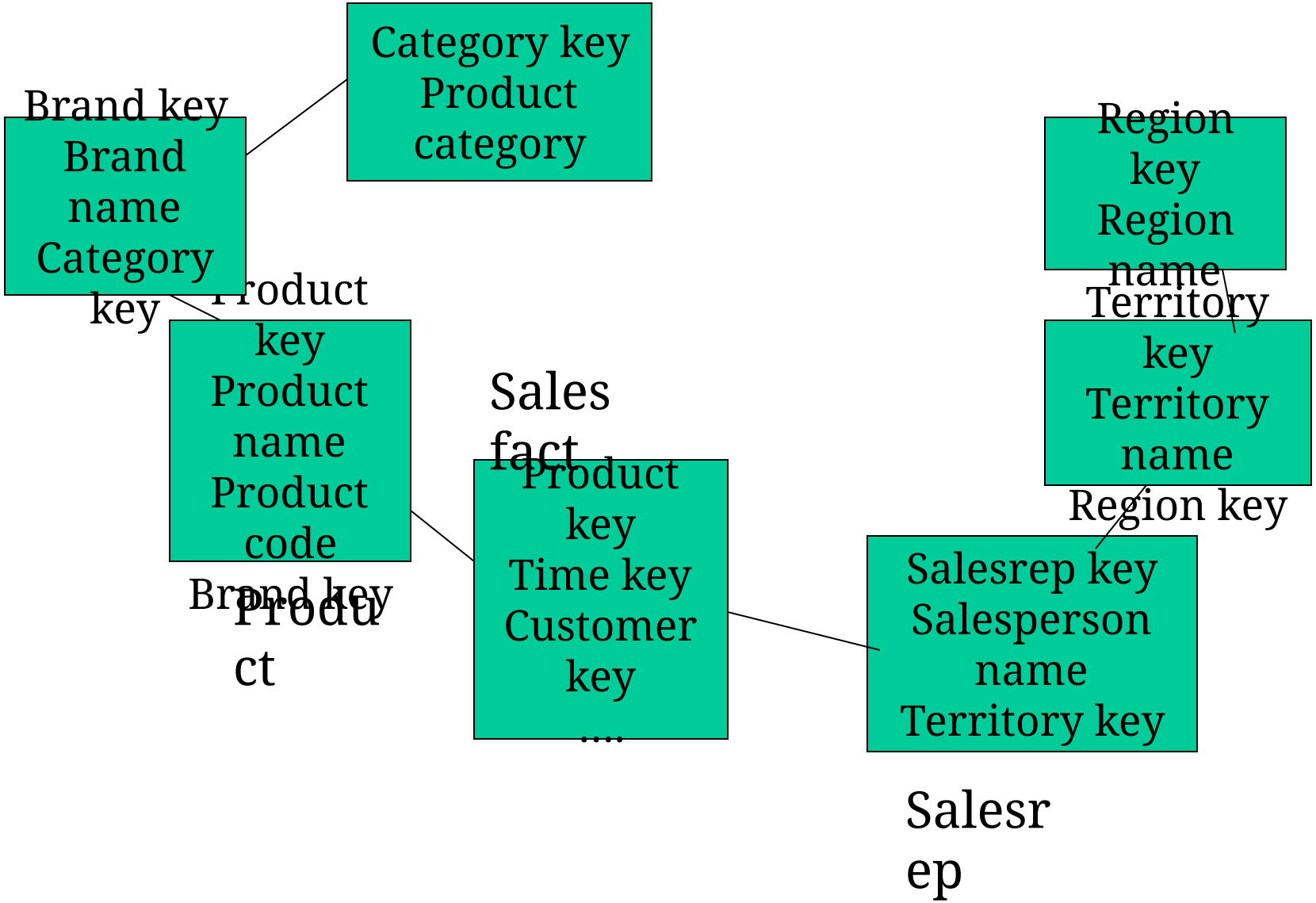
Families of Stars



Snowflake Schema

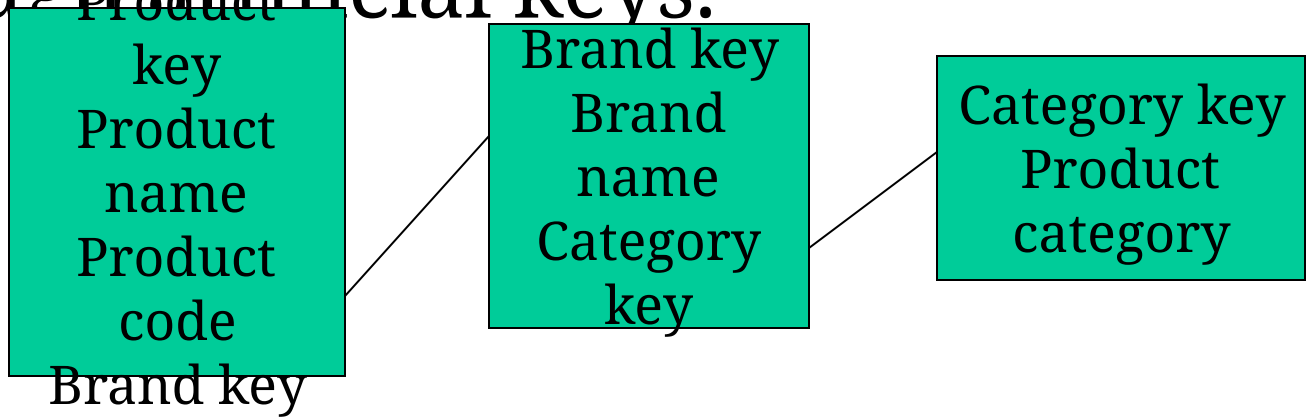
- Snowflake schema is a type of star schema but a more complex model.
- “Snowflaking” is a method of normalizing the dimension tables in a star schema.
- The normalization eliminates redundancy.
- The result is more complex queries and reduced query performance.

Sales: Snowflake Schema



Snowflaking

- The attributes with low cardinality in each original dimension table are removed to form separate tables. These new tables are linked back to the original dimension table through artificial keys.



Snowflake Schema

- Advantages:
  - Small saving in storage space
  - Normalized structures are easier to update and maintain
- Disadvantages:
  - Schema less intuitive and end-users are put off by the complexity
  - Ability to browse through the contents difficult
  - Degrade query performance because of additional joins

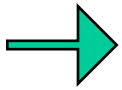
What is the Best Design?

- Performance benchmarking can be used to determine what is the best design.
- Snowflake schema: easier to maintain dimension tables when dimension tables are very large (reduce overall space). It is not generally recommended in a data warehouse environment.
- Star schema: more effective for data cube browsing (less joins): can affect performance.

Aggregates

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

sale	prodId	storeId	date	amt
	p1	s1	1	12
	p2	s1	1	11
	p1	s3	1	50
	p2	s2	1	8
	p1	s1	2	44
	p1	s2	2	4

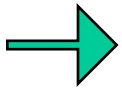


8
1

Aggregates

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

sale	prodId	storeId	date	amt
	p1	s1	1	12
	p2	s1	1	11
	p1	s3	1	50
	p2	s2	1	8
	p1	s1	2	44
	p1	s2	2	4

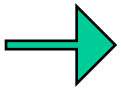


ans	date	sum
	1	81
	2	48

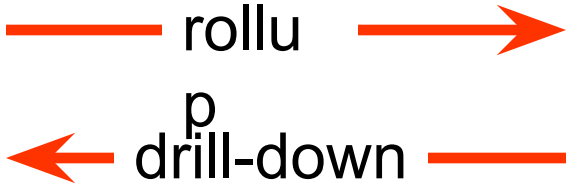
Another Example

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

sale	prodId	storeId	date	amt
	p1	s1	1	12
	p2	s1	1	11
	p1	s3	1	50
	p2	s2	1	8
	p1	s1	2	44
	p1	s2	2	4



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



Aggregates

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

Data Cube

Fact table view:

sale	prodId	storeId	amt
	p1	s1	12
	p2	s1	11
	p1	s3	50
	p2	s2	8

Multi-dimensional cube:

	s1	s2	s3
p1	12		50
p2	11	8	

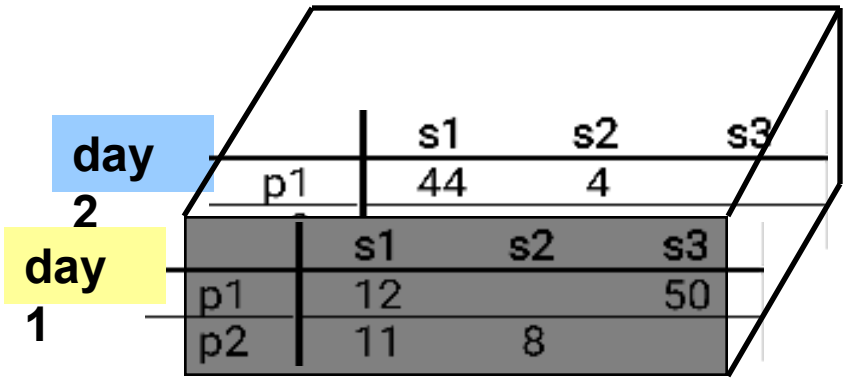
dimensions = 2

3-D Cube

Fact table view:

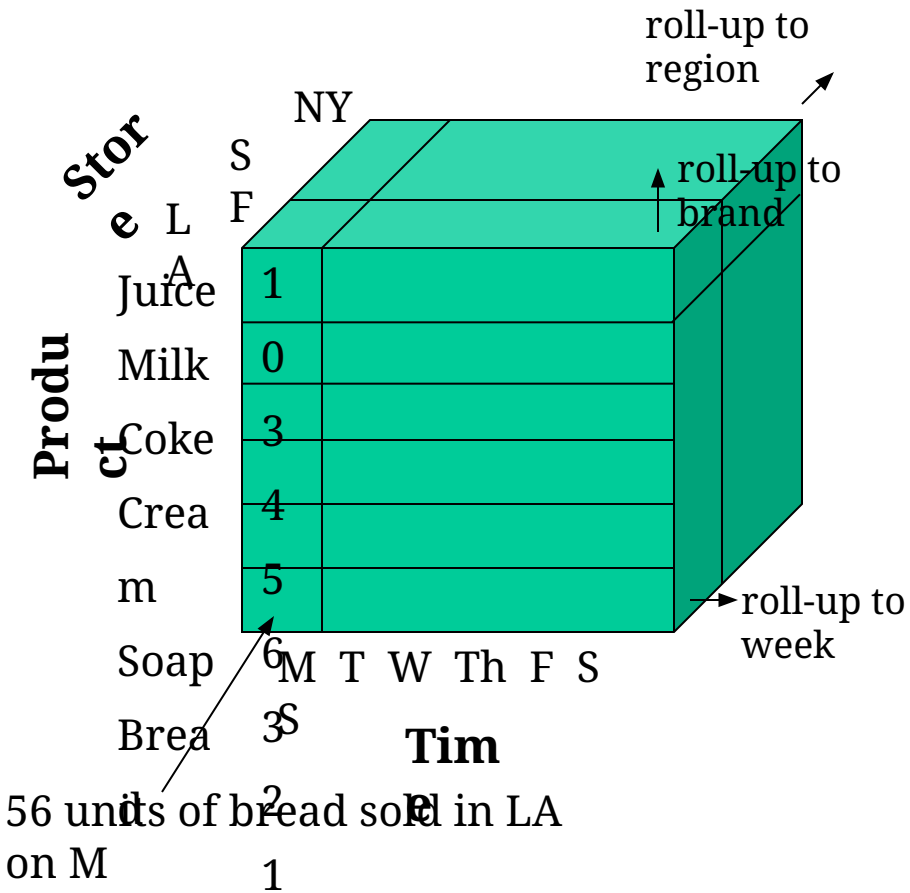
sale	prodId	storeId	date	amt
	p1	s1	1	12
	p2	s1	1	11
	p1	s3	1	50
	p2	s2	1	8
	p1	s1	2	44
	p1	s2	2	4

Multi-dimensional cube:



dimensions = 3

Example



Dimensions:

Time, Product, Store

Attributes:

Product (upc, price, …)

Store …

…

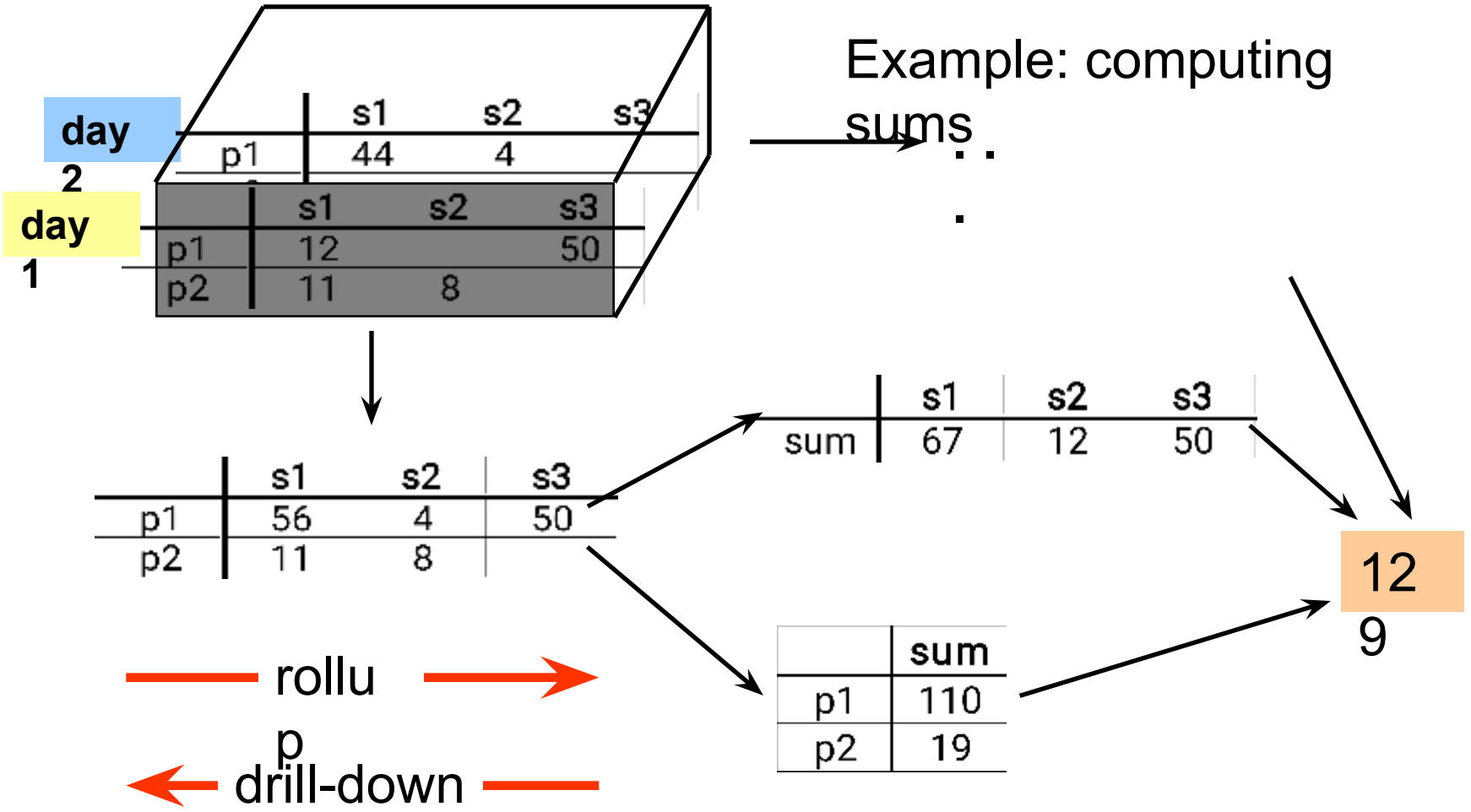
Hierarchies:

Product → Brand → …

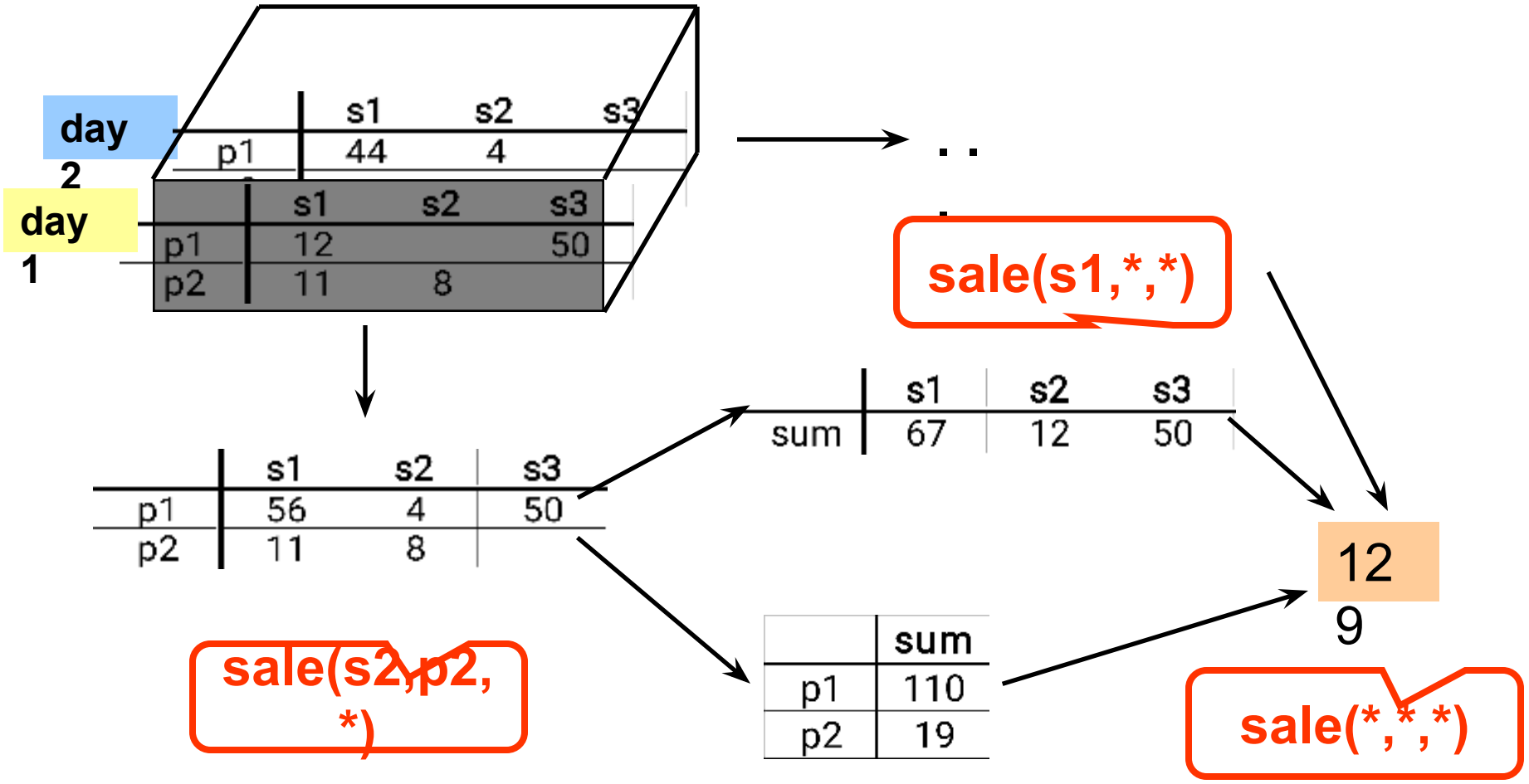
Day → Week → Quarter

Store → Region → Country

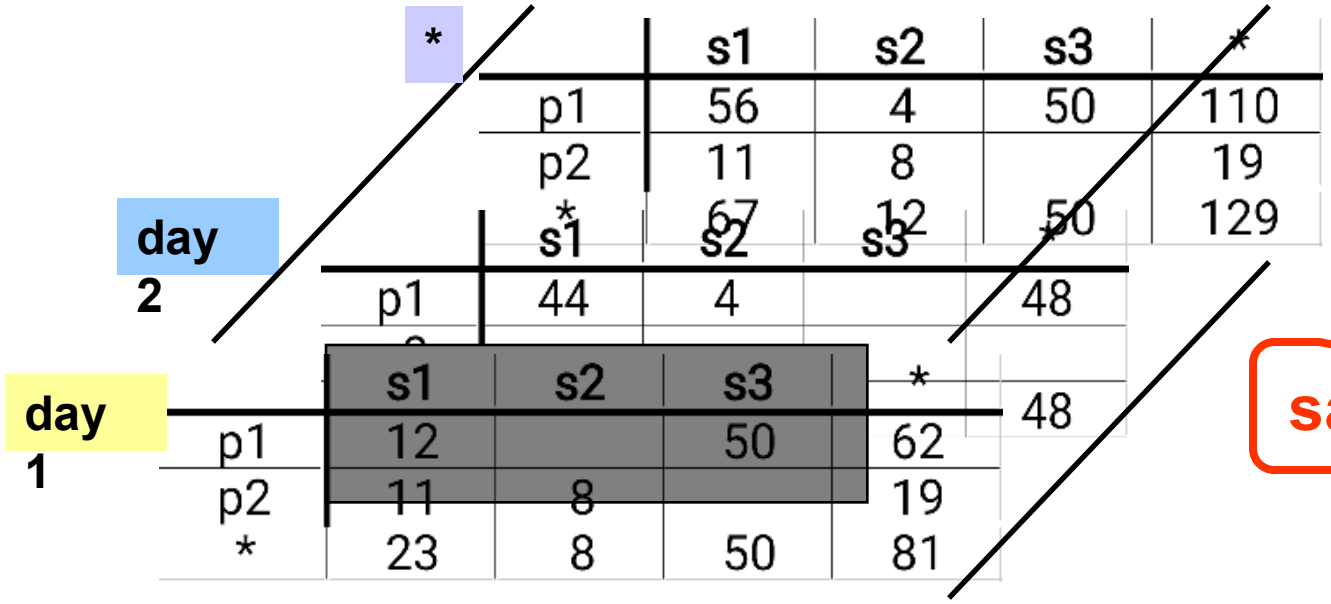
Cube Aggregation: Roll-up



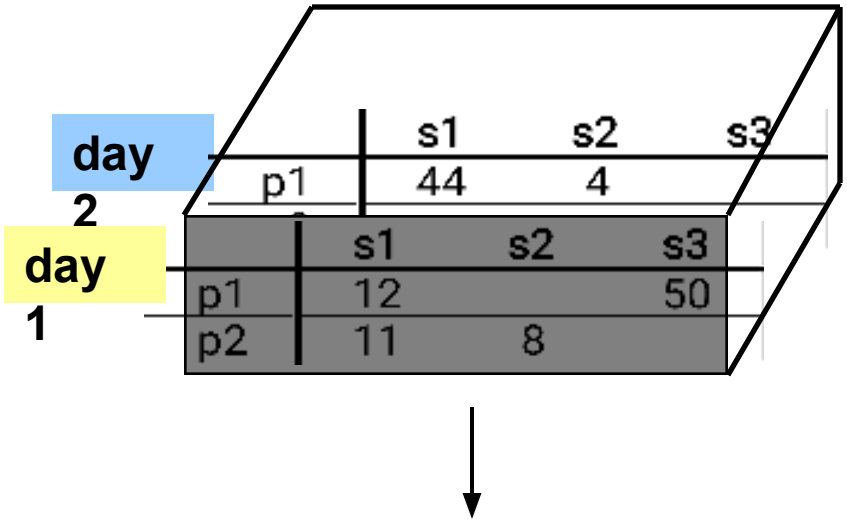
Cube Operators for Roll-up



Extended Cube



Aggregation Using Hierarchies

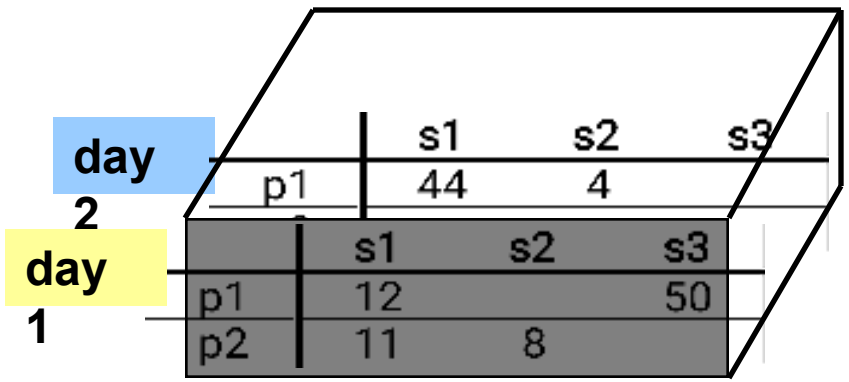


	region A	region B
p1	56	54
p2	11	8

stor
e
region
country

(store s1 in Region A; stores s2, s3 in Region B)

Slicing



TIME = day 1

	s1	s2	s3
p1	12		50
p2	11	8	

Summary of Operations

- Aggregation (roll-up)
  - aggregate (summarize) data to the next higher dimension element
  - e.g., total sales by city, year → total sales by region, year
- Navigation to detailed data (drill-down)
- Selection (slice) defines a subcube
  - e.g., sales where city ='Gainesville' and date = '1/15/90'
- Calculation and ranking
  - e.g., top 3% of cities by average income
- Visualization operations (e.g., Pivot)
- Time functions
  - e.g., time average

Query & Analysis Tools

- Query Building
- Report Writers (comparisons, growth, graphs,...)
- Spreadsheet Systems
- Web Interfaces
- Data Mining