Database Management Systems - I, CS 157A

OLTP vs. Data Warehouse

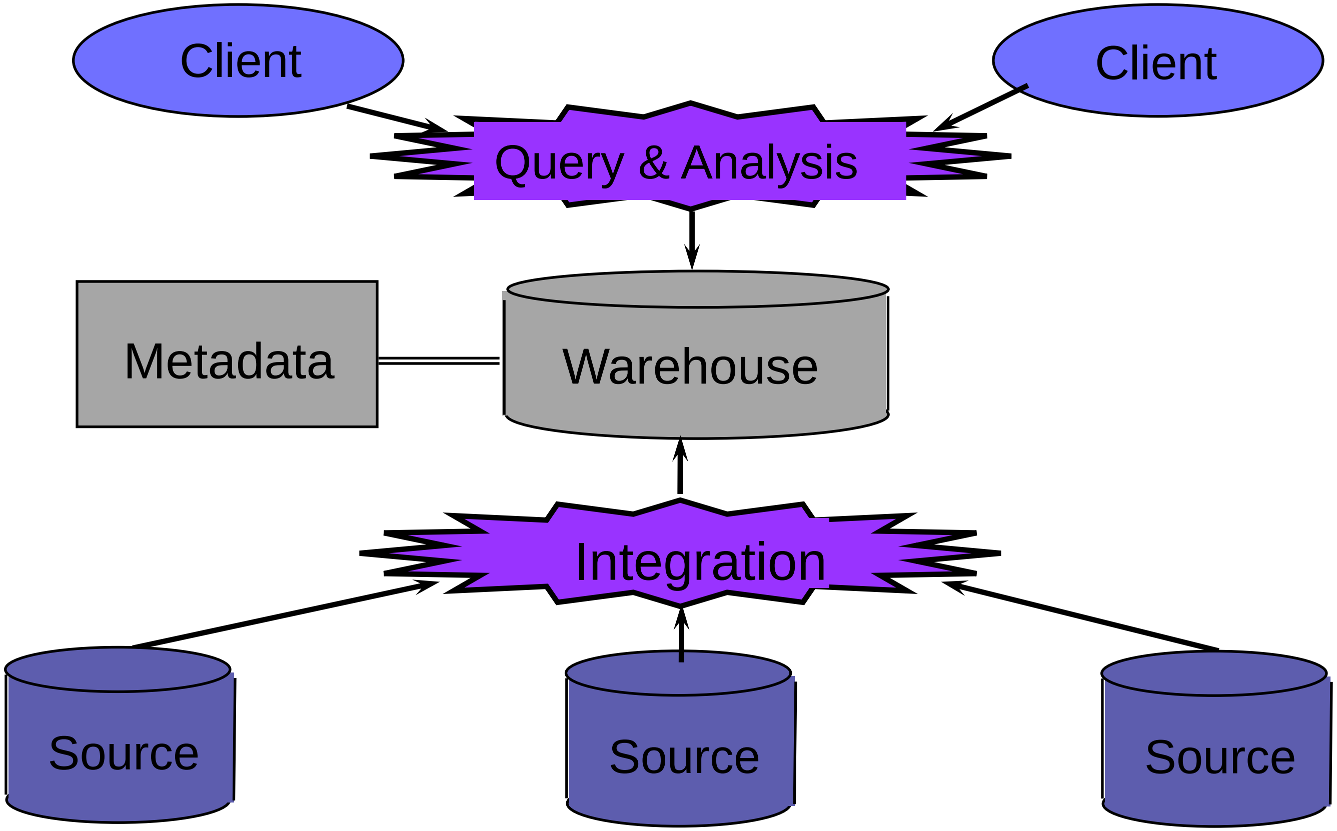
Agenda

- What is Data Warehouse
- OLTP vs. DW Characteristics
- DW Common Schemas
- Summary

What is Data Warehouse

What is a Data Warehouse

- Data Warehouse (DW) Architecture:



DW Definition and Concepts

- **Data warehouse:**
  - A physical repository where relational data are specially organized to provide enterprise-wide, cleansed data in a query-able format.
  - Some data is de-normalized for simplification and to improve performance.
  - Large amount of historical data.
  - Both planned and ad-hoc queries are common.
  - The data load is controlled.

DW Definition and Concepts

- **Characteristics of data warehousing:**
  - □ Subject oriented.
  - □ Integrated.
  - □ Time variant (time series).
  - □ Nonvolatile.
  - □ Web based.
  - □ Relational/multidimensional.
  - □ Client/server.
  - □ Near Real-time.
  - □ Include metadata.

Data Warehouse Overview

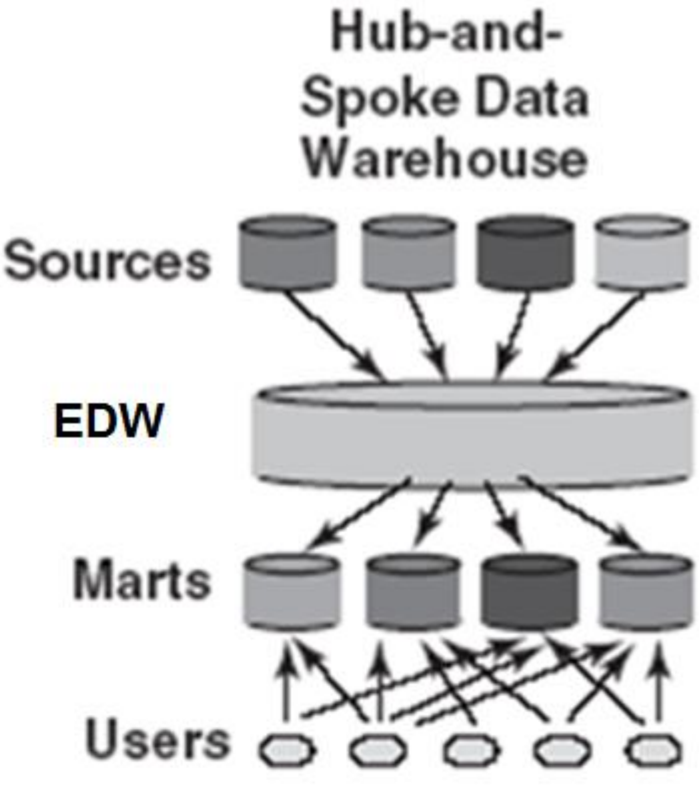
- **DW** presents (collects) a unified view of data from all relevant data sources in the enterprise to be used to help improving its operation.
- **DW** is typically implemented using an extended version of traditional RDBMS: (1) Typically connected to many transactional data sources (operational servers). (2) Bulk and efficient load and extract data to/from DW is a must. (3) Supports massive amount of data including historical data. (4) Tables are typically optimized (special schema - Star schema) for efficient complex query execution. (5) Needs to support complex queries with large size joins (15-20 way joins).

Data Warehouse Overview

- **DW comes in two flavors:**
  - □ **Enterprise Data Warehouse (EDW):** Enterprise-wide server that enable global optimizations.
  - □ **Data Mart:** Department-level warehouse:
    - ➢ **Dependent data mart**
      A subset that is created directly from a data warehouse.
    - ➢ **Independent data mart**
      A small data warehouse designed for a strategic business unit or a department.

DW Architecture: Hub-and-Spoke Data Mart

Hub-and-Spoke is an architecture with a centralized EDW (Hub) to set & enforce common standards and answering queries at the enterprise level that cuts across department, and set of data marts (Spokes) that allow departments (BU) to meet their needs, and exchange data with the Hub according to their own schema.



DW vs. DSS vs. BI

- **DSS (Decision Support System)** is A conceptual framework for a process of supporting managerial decision-making, usually by modeling problems and employing quantitative models for solution analysis; DSS do not necessarily require the use of DW as a source of data, e.g., spreadsheet is used in IT as a popular decision support tool.
- **BI** is a framework for decision support. It combines DW, analytics & data mining tools and other applications such as visualization. In summary, use data to advance your business.
- **Historically BI is used for:** (1) Confirm a suspicion. (2) Identify out of the ordinary. (3) Generate reports , e.g., compare information about customers, product, profit over different time periods. (4) Confirm or discover trends and relationships.

BI Framework

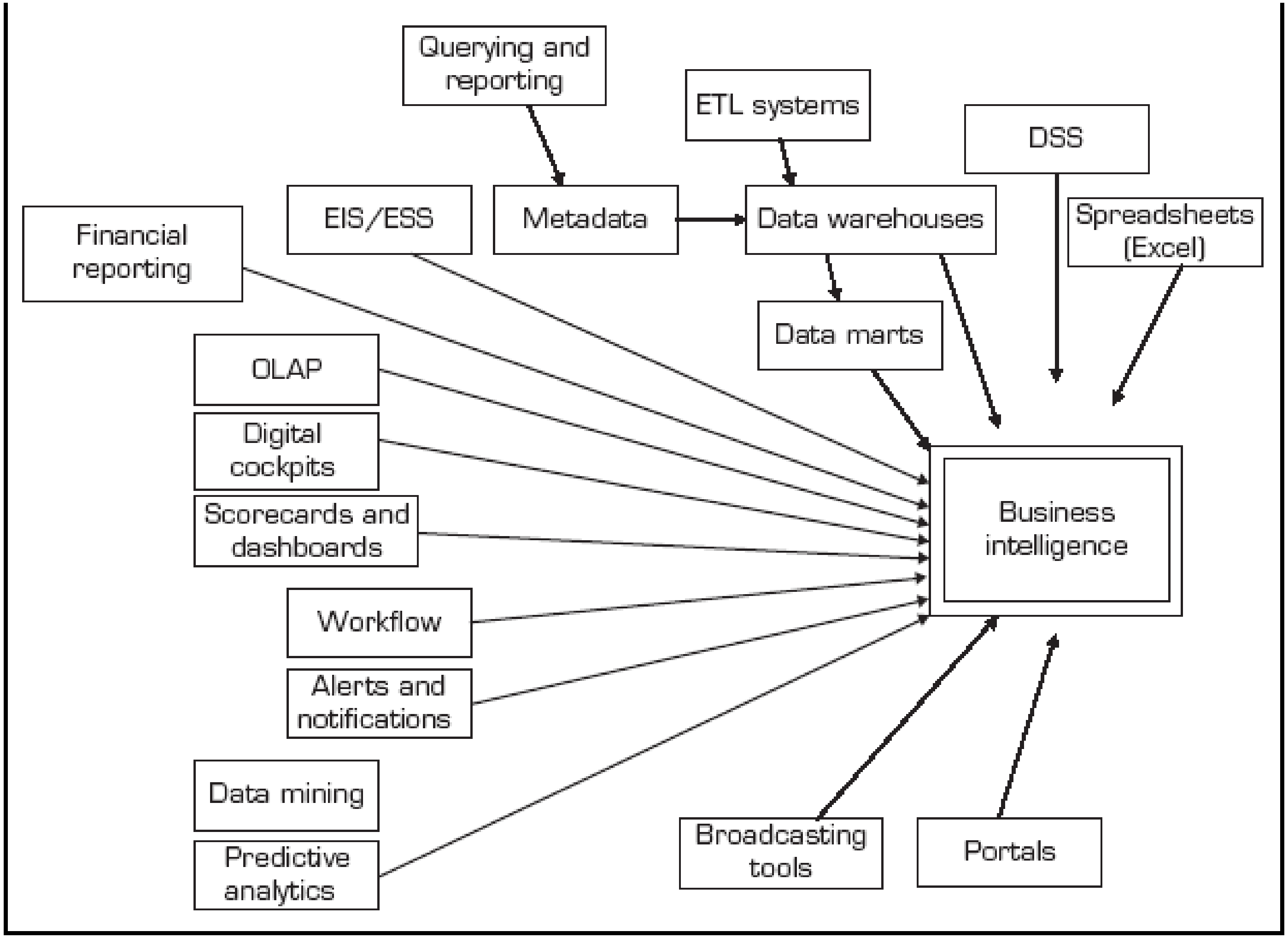


FIGURE 1.2 Evolution of BI

BI Components

- **BI** typically collects most of its data from On-line Transaction Processing systems (OLTP).
- **OLAP (On-line Analytics Processing):** OLAP allows business users to slice and dice the data at will. Answers multi-dimensional analytical queries.
- **Data Mining**: A class of BI information analysis based on databases that looks for hidden patterns in a collection of data which can be used to predict future behavior.
- **BPM (Business Performance Mgmt):** A component of BI based on the *balanced scorecard* methodology. Defines/implement business strategy by linking objectives with factual measures.
- **Dashboard:** A visual presentation of critical data for executives to view.

OLTP vs. DW Characteristics

OLTP vs. OLAP

OLTP

- Mostly updates
- Many small transactions
- Mb-Gb of data
- Raw data
- Clerical users
- Up-to-date data
- Consistency, recoverability critical

OLAP

- Mostly reads
- Queries long, complex
- GB-TB of data
- Summarized, consolidated data
- Decision-makers, analysts as users

DW vs. RDBMS/OLTP

- **A fundamental difference is** that most database instance will place an emphasis on one subject area/application and is typically transactional, while DW deals with multiple subject areas simultaneously.
- **DW** has the ability to support analysis of trends.
- **DW data do not change** as much as in transactional systems.
- **DW has to be built to support very complex queries** (e.g., 20-way join) while traditional databases would not be capable of executing such complex queries.

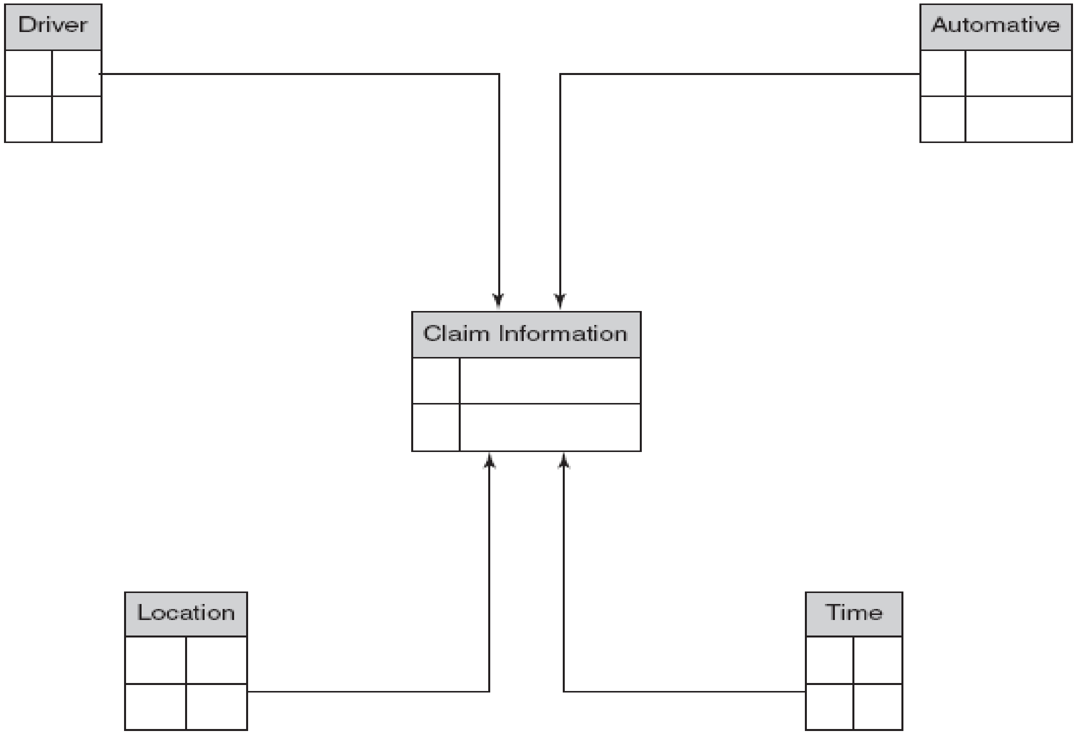
DW vs. RDBMS/OLTP

- **Some DW engines support OLAP operations** inside the engine (ROLAP) vs. traditional MOLAP support outside the RDBMS.
- **In databases,** it is important to normalize tables, while with data warehouse we might sacrifice normalization for the sake of efficiency.
- **Additional differences are:** dimensionality (data is connected in different ways with DW), timespan (short time vs. long time frame), and granularity (managers needs information that is summarized at various degrees).

DW Common Schemas

Data Warehouse – Star Schema

Star Schema Example
Automobile Insurance Data Warehouse



Dimension:
How data will be accessed (e.g. by location, time period, type of automobile or driver)

Facts:
Central table that contains summarized (usually) information; also contains foreign keys to access each dimension table

FIGURE 2.9 Star Schema

Data Warehouse – Star Schema

- **Data warehouse structure**: The **Star Schema**
  - □ **Dimensional modeling**
    - A retrieval-based system that supports high-volume query access.
  - □ **Fact Table(s)**
    - One or more tables with clustered primary key (aggregate of dimensions primary keys); 3rd order form – data depends on one dimension or all of them.
    - Contain the detail data.
    - Simple queries – join between fact and one dimension tables.
  - □ **Dimension tables**
    - A table with simple primary key that address *how* data will be analyzed; 2nd order form.
    - Contain the look up information.
  - □ **Summary tables**
    - **Pre-summarized results** – goal is to pre-summarize data to meet 90% of user queries.
    - **Most queries are for month-to-date** totals and balances therefore summary tables are month-to-date.
    - **Use triggers and stored procedures** to maintain Financial summary tables current.; when a new record is loaded, the summary table is automatically updated.
  - □ Allow for unexpected **ad-hoc** queries.

Data Warehouse – Star Schema

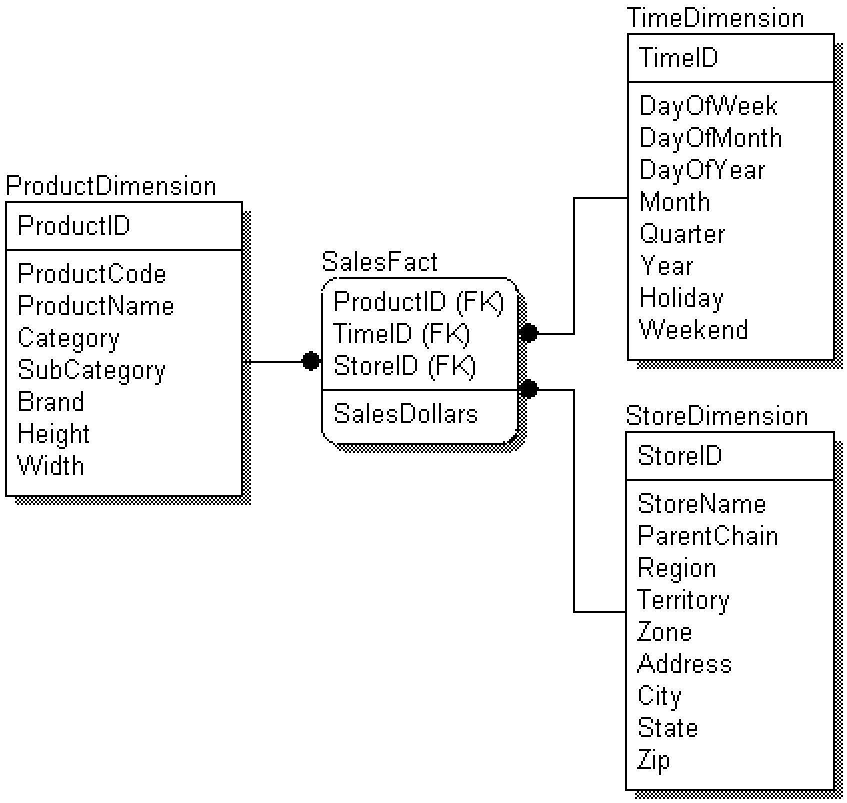
- Data warehouse structure - The Star Schema:
  - People cares about aggregates, called measures.
  - Measure is not enough but rather using the “by” condition, e.g., sales by day, sales by quarter, etc.
  - These “by” conditions are called dimensions. Time, geography, product are common dimensions.
  - First step in designing a star schema is to find out what (measures) people are interested in, and how they want to see it (dimensions).
  - Sometimes you need to build dimension hierarchy! Hierarchy in OLAP is different as the hierarchy for a given dimension is stored in a single dimension table unless use snowflaked schema. An example is “product” may be structured/grouped into categories and subcategories.

Data Warehouse – Star Schema

- **Data warehouse structure**: The Star Schema
  - With snowflakes schema, a dimension table have the hierarchy broken into set of tables – more complicated and slower queries.
  - The Fact table(s) holds the measures, or facts. Measures are numeric and are added across some or all of the dimensions For example, sales are numeric and users can look at totals for a product, or category, or subcategory, and by any time period.
  - While dimensional tables are short and fat, the fact table is typically long and skinny; they are long as they includes records represented by the product in all dimension tables.

Data Warehouse – Star Schema

- Assume product, time, and store dimensions. With 10 years of daily data, 200 stores, and 500 products, there is (3650 * 200 * 500 = 365,000,000) records in the fact table.
- Fact table is skinny (small # of attributes) as it holds mainly the primary key of the dimension tables, typically integers.
- Granularity, frequency, of the fact table is determined by the lowest level of granularity of each dimension table.



Traditional DW Star Schema: Why Star Schema?

- **Star schema** is used for OLAP system to speed up the retrieval, not transactions, and holds read-only, historical, and possibly aggregated data.
- Star Schema simplify aggregation. The level of aggregation in a star schema depends on the scenario. Many star schema are aggregated to some base level, called **the grain**, however, it is more common to rely on cubes building engines to summarize to a base level of granularity.
- Star Schema allows us to view data as aggregated numbers broken down along different criteria - dimensions
- OLAP/Star Schema may be the actual data warehouse, typically we will build cube structures from the relational data warehouse in order to provide faster, more powerful analysis on the data.

Traditional DW Star Schema: Why Denormalizing?

- Database normalization is the process of removing repeated information.
- Normalization makes sense with OLTP environment as it reduces repeated information and fewer indexes per table. Given a transaction which does insert/update/delete will need to do index maintenance for affected indexes, then less indexes is good for transaction environment.
- Data warehouse denormalize for speed! DW is not built for I/U/D but rather for retrieval (Select). Furthermore, by denormalizing, replicating data, we can have joins with less number of tables which is critical in a data warehouse.

Traditional DW Star Schema: Data Loading

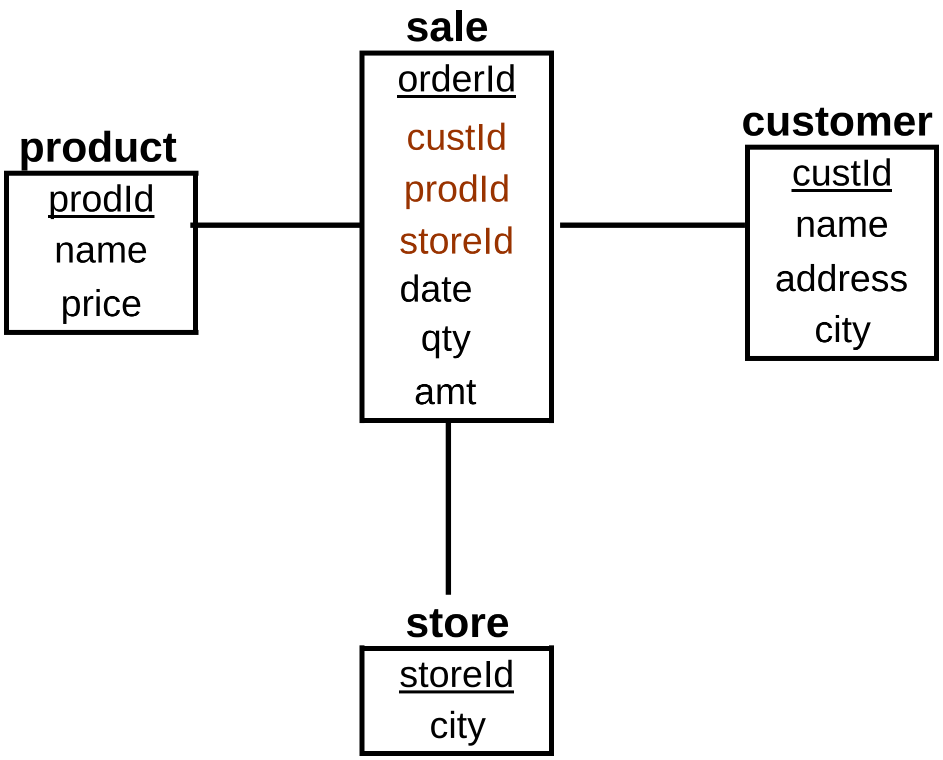
- Data sources “pushes” the data to the data warehouse in the batch window or in near real-time via (FTP).
- Cron job checks for new data and start loading:
  - Loads Dimension tables when data has changed.
  - Loads daily General Journal File.
- High performance loader loads data into a staging database; including some transformations.
- SQL program extracts relevant records from the daily journal and loads into the fact tables.
- SQL program loads and **rebuilds the dimension tables**.
- **Summary tables** are updated by Triggers and SQL.

Data Warehouse Models & Operations

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

Data Warehouse Models & Operations

- Star Schema



sale
orderId
custId
prodId
storeId
date
qty
amt

product
prodId
name
price

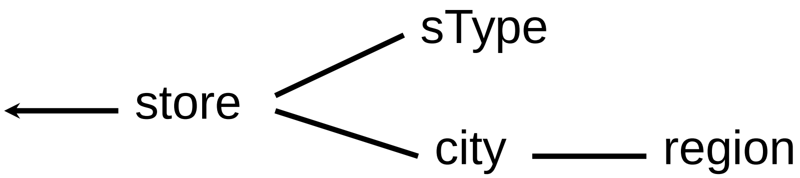
customer
custId
name
address
city

store
storeId
city

- Terms:
  - **Fact table**: PK is the aggregate of the dimensions' PK - 2nd NF.
  - **Dimension tables**: "by" condition (**sales** by product) - one view for how the data will be analyzed – 3rd NF.
  - **Measures**: aggregates – sales/day, sales/quarter, etc.
  - **Summary Tables**: pre-summarized data to meet 90% of user queries. Use triggers/stored procedures to maintain summary tables current on insert/delete.

Data Warehouse Models & Operations

- **Star Schema - Dimension Hierarchies**



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

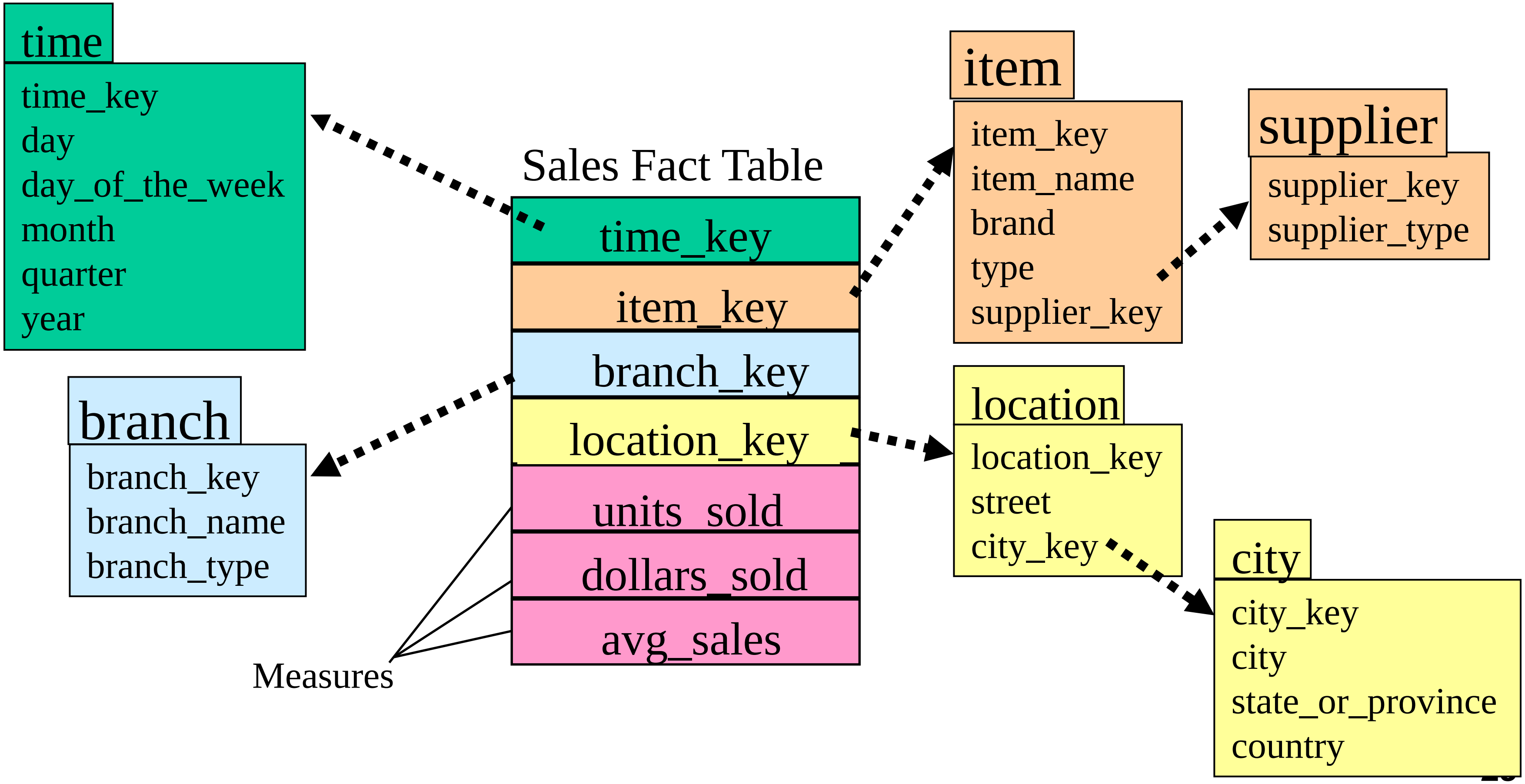
sType	tId	size	location
	t1	small	downtown
	t2	large	suburbs

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

region	regId	name
	north	cold region
	south	warm region

Data Warehouse Models & Operations

- **Snowflakes schema**, a dimension table have the hierarchy broken into set of tables – more complicated and slower queries.



Sales Fact Table

time_key
item_key
branch_key
location_key
units_sold
dollars_sold
avg_sales

Measures

time
time_key
day
day_of_the_week
month
quarter
year

branch
branch_key
branch_name
branch_type

item
item_key
item_name
brand
type
supplier_key

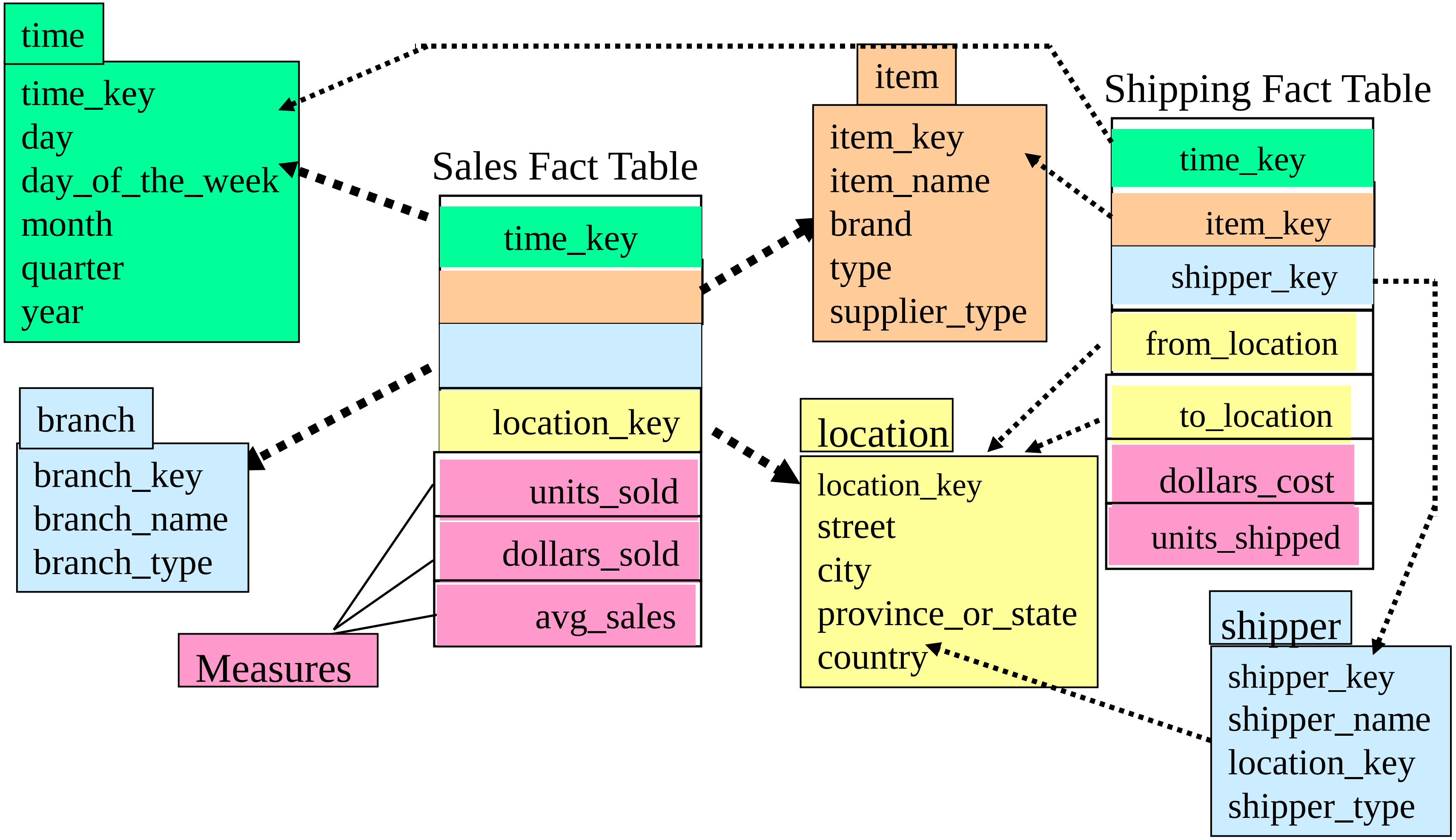
supplier
supplier_key
supplier_type

location
location_key
street
city_key

city
city_key
city
state_or_province
country

Example of Fact Constellation

- **Fact Constellations schema**, multiple fact tables that share the set of dimensional tables; viewed as a collection of stars.



Summary

Summary

- We covered what is Data Warehouse.
- Different flavors of DW.
- Different popular schemas used by DW.
- High-level comparison between traditional operational transactional databases (OLTP) and Data Warehouses.

END

Data Warehouse Development - Implementation Details

Phase-1	Phase-2	Phase-3	Phase-4	Phase-5	Phase-6
Creating a Data Warehouse Prototype	Procuring the Data Warehouse Equipment and Consulting Service	Developing the DW software and Converting the Initial Data	Installing the DW Hardware, Software and Converted Data	Training the DW users and Operational Staff	Refining the DW Data, Queries and Reports
Data Warehouse Prototype	Contracts and Implementation Plan	Operational Software, Initial Queries, reports and Data	DW goes Live	Software Documentation and User Manuals	Revised Queries and Reports
6 weeks	1 Month	3 Months	1 Month	1 Month	3 Months

Data Warehouse Models & Operations

Fact table view:

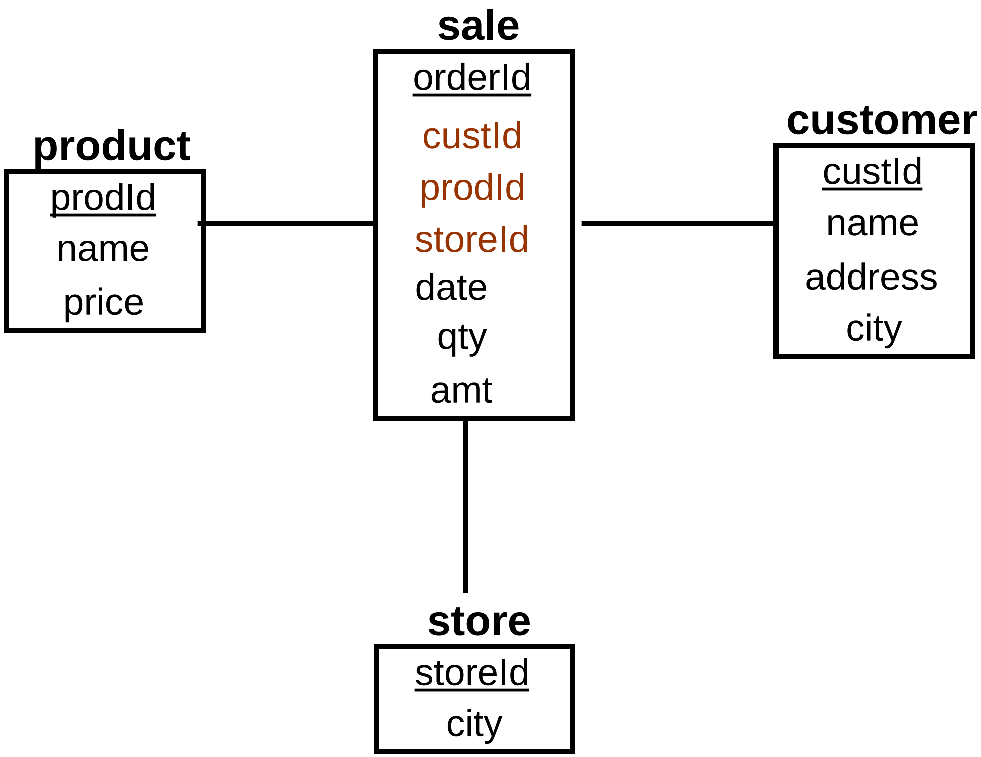
sale	prodId	storeId	amt
	p1	c1	12
	p2	c1	11
	p1	c3	50
	p2	c2	8

↕

Multi-dimensional cube:

	c1	c2	c3
p1	12		50
p2	11	8	

dimensions = 2



sale

orderId

custId

prodId

storeId

date

qty

amt

product

prodId

name

price

customer

custId

name

address

city

store

storeId

city

Star Schema

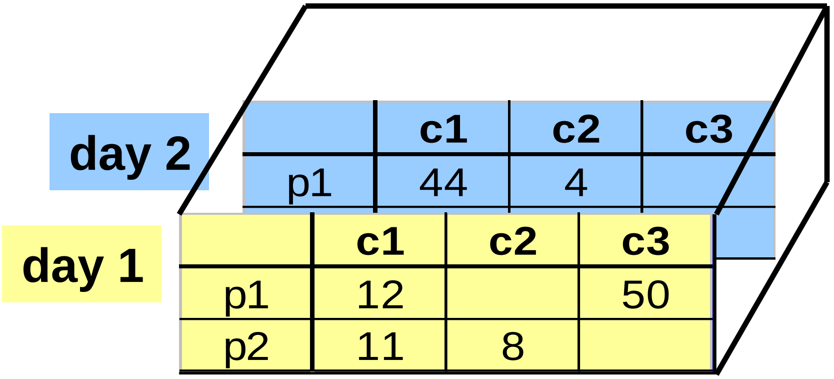
Data Warehouse Models & Operations

- 3D 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:



day 2

	c1	c2	c3
p1	44	4	

day 1

	c1	c2	c3
p1	12		50
p2	11	8	

dimensions = 3

Data Warehouse Models & Operations

- 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

Data Warehouse Models & Operations

- Aggregates (Contd.)
  - 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

sale	date	sum
	1	81
	2	48

Data Warehouse Models & Operations

- Aggregates (Contd.)
  - □ 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 →

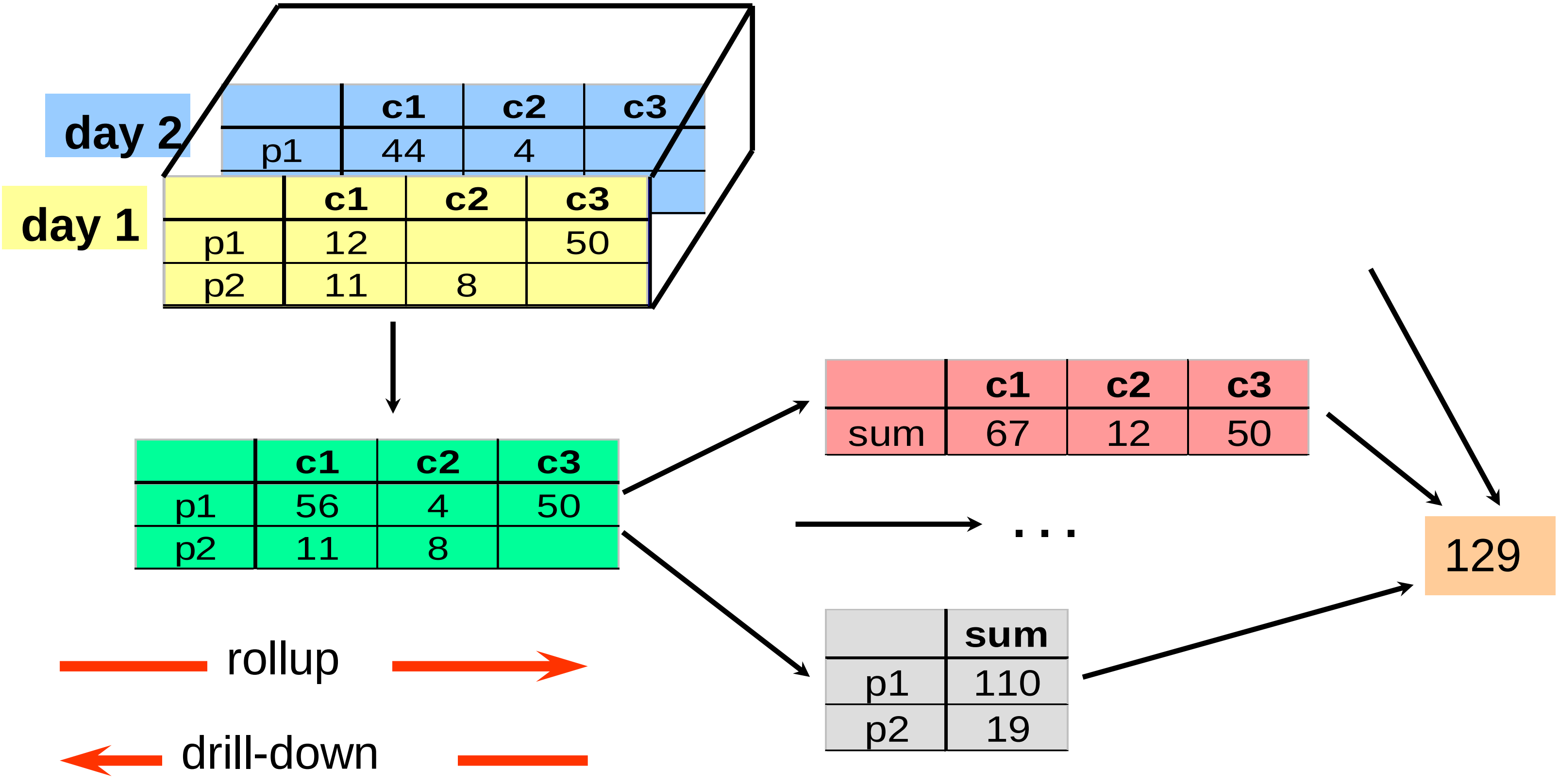
← drill-down

Data Warehouse Models & Operations

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

Data Warehouse Models & Operations

- Cube Aggregations



day 2

	c1	c2	c3
p1	44	4	

day 1

	c1	c2	c3
p1	12		50
p2	11	8	

	c1	c2	c3
p1	56	4	50
p2	11	8	

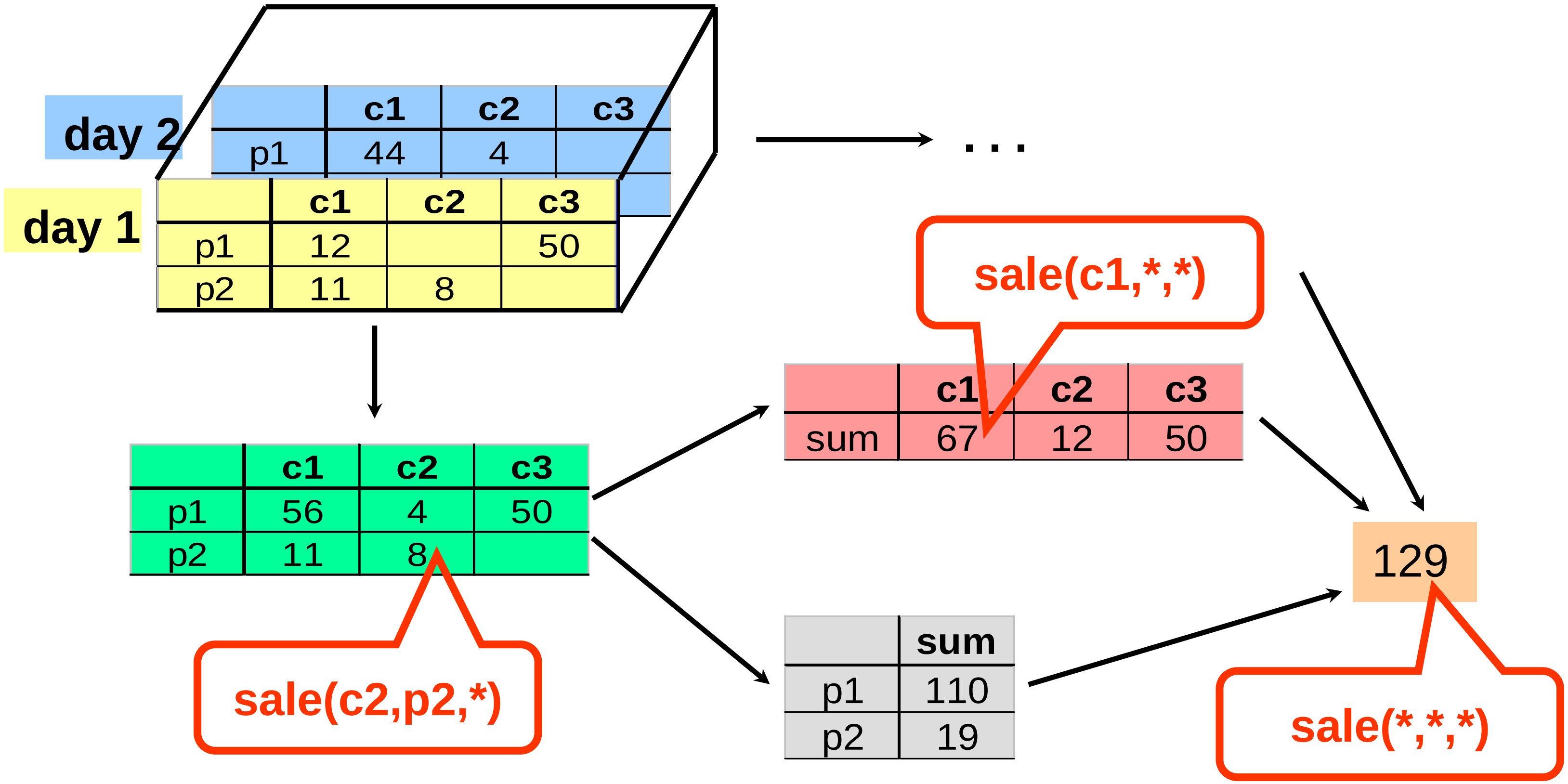
	c1	c2	c3
sum	67	12	50

	sum
p1	110
p2	19

Example: computing sums

Data Warehouse Models & Operations

- Cube Operators



day 2

	c1	c2	c3
p1	44	4	

day 1

	c1	c2	c3
p1	12		50
p2	11	8	

. . .

	c1	c2	c3
p1	56	4	50
p2	11	8	

sale(c2,p2,*)

	c1	c2	c3
sum	67	12	50

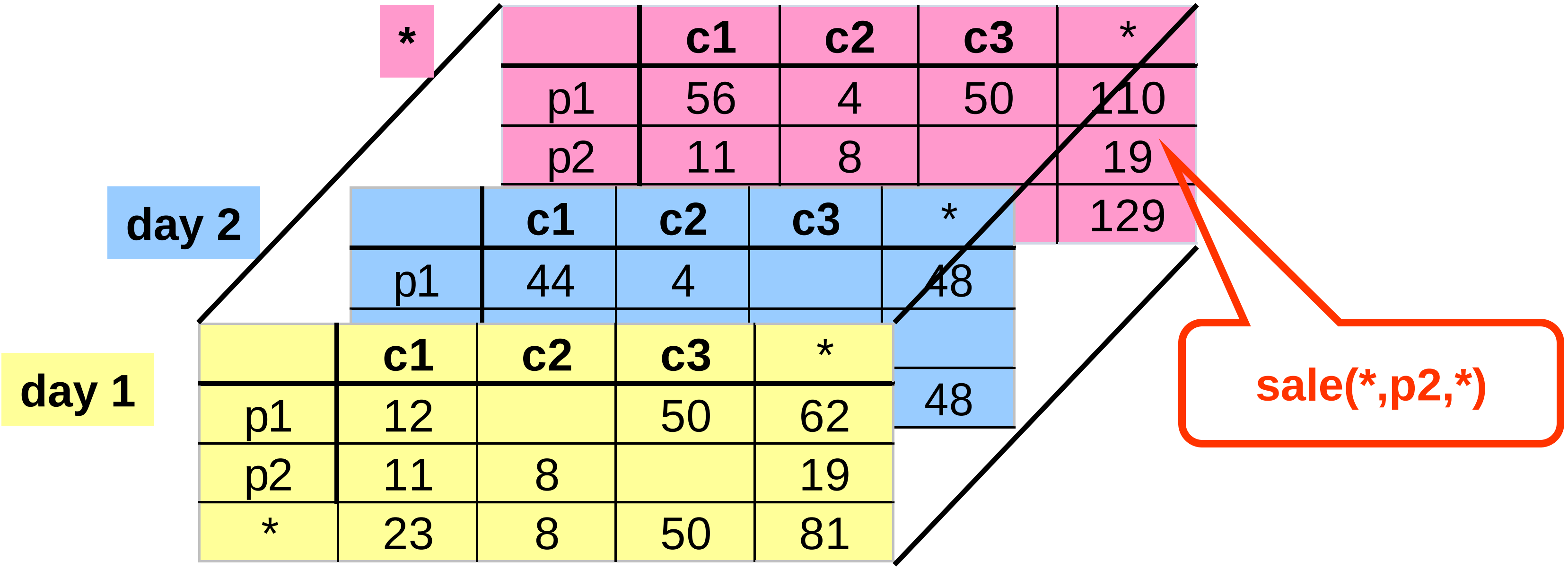
sale(c1,*,*)

	sum
p1	110
p2	19

129

sale(*,*,*)

- Extended Cube



*

	c1	c2	c3	*
p1	56	4	50	110
p2	11	8		19
				129

day 2

	c1	c2	c3	*
p1	44	4		48
				48

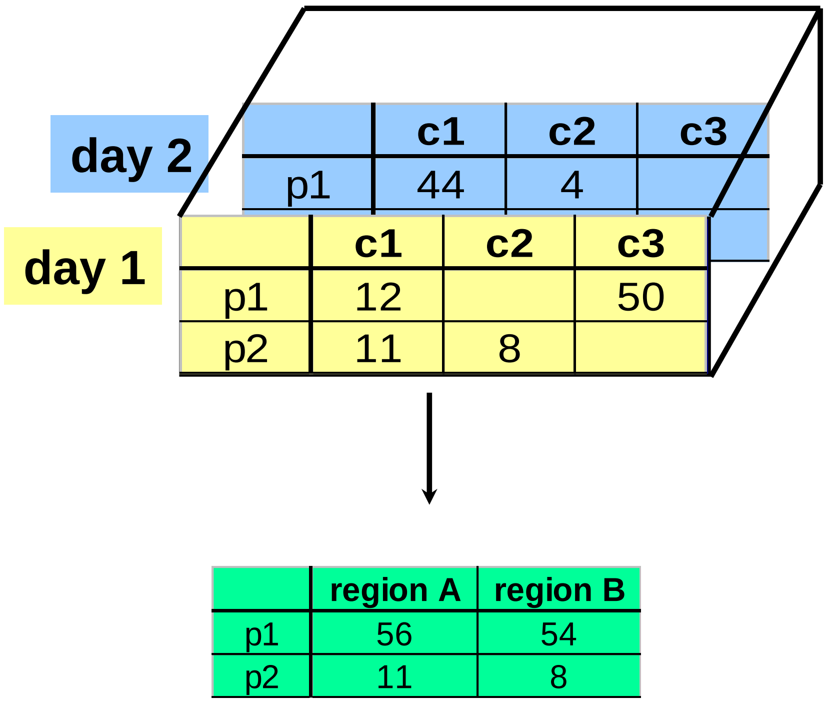
day 1

	c1	c2	c3	*
p1	12		50	62
p2	11	8		19
*	23	8	50	81

sale(*,p2,*)

Data Warehouse Models & Operations

- Aggregation Using Hierarchies



day 2

	c1	c2	c3
p1	44	4	

day 1

	c1	c2	c3
p1	12		50
p2	11	8	

	region A	region B
p1	56	54
p2	11	8

customer

region

country

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

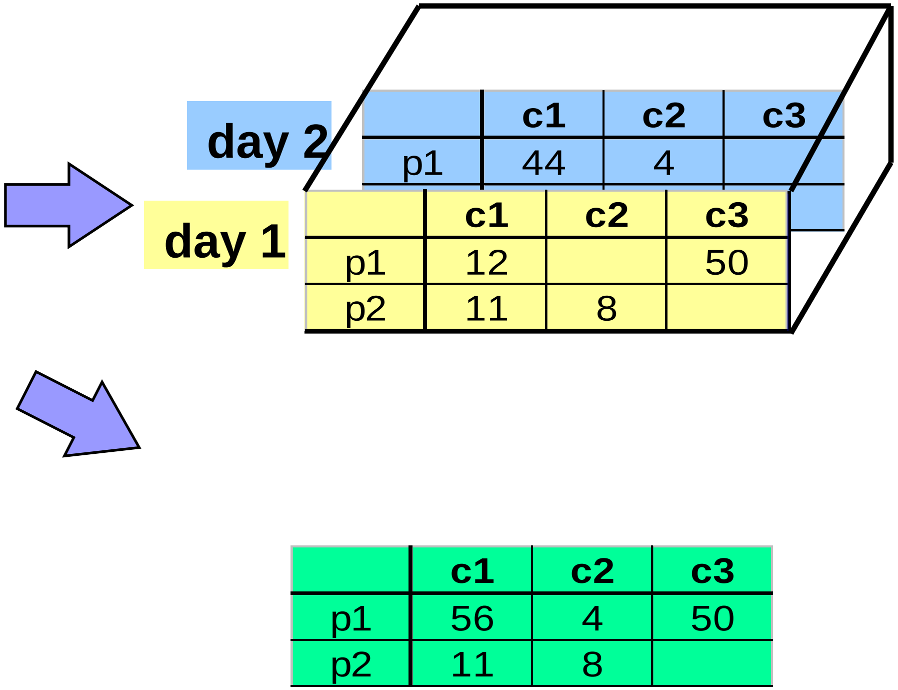
Data Warehouse Models & Operations

- 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:



day 2

	c1	c2	c3
p1	44	4	

day 1

	c1	c2	c3
p1	12		50
p2	11	8	

	c1	c2	c3
p1	56	4	50
p2	11	8	