# Data Warehousing

INF 551

Wensheng Wu

# Warehousing

- Growing industry:
  - $8 billion in 1998
  - $16.9 billion in 2018
- Range from desktop to high-end system:
  - Wal-Mart in 1990's
    - 900-CPU, 2,700 disk, 23TB, Teradata system
  - What about 2007?
    - Wal-Mart Plans for Its 4PB Data Warehouse
    - 100 billion rows in tables
    - 276 million records/day from POS systems
- Lots of buzzwords, hype
  - Data cube, roll-up, drill-down, slice/dice, ...

# Outline

- What is a data warehouse?
- Why a warehouse?
- Data models & operations
- Implementing a warehouse

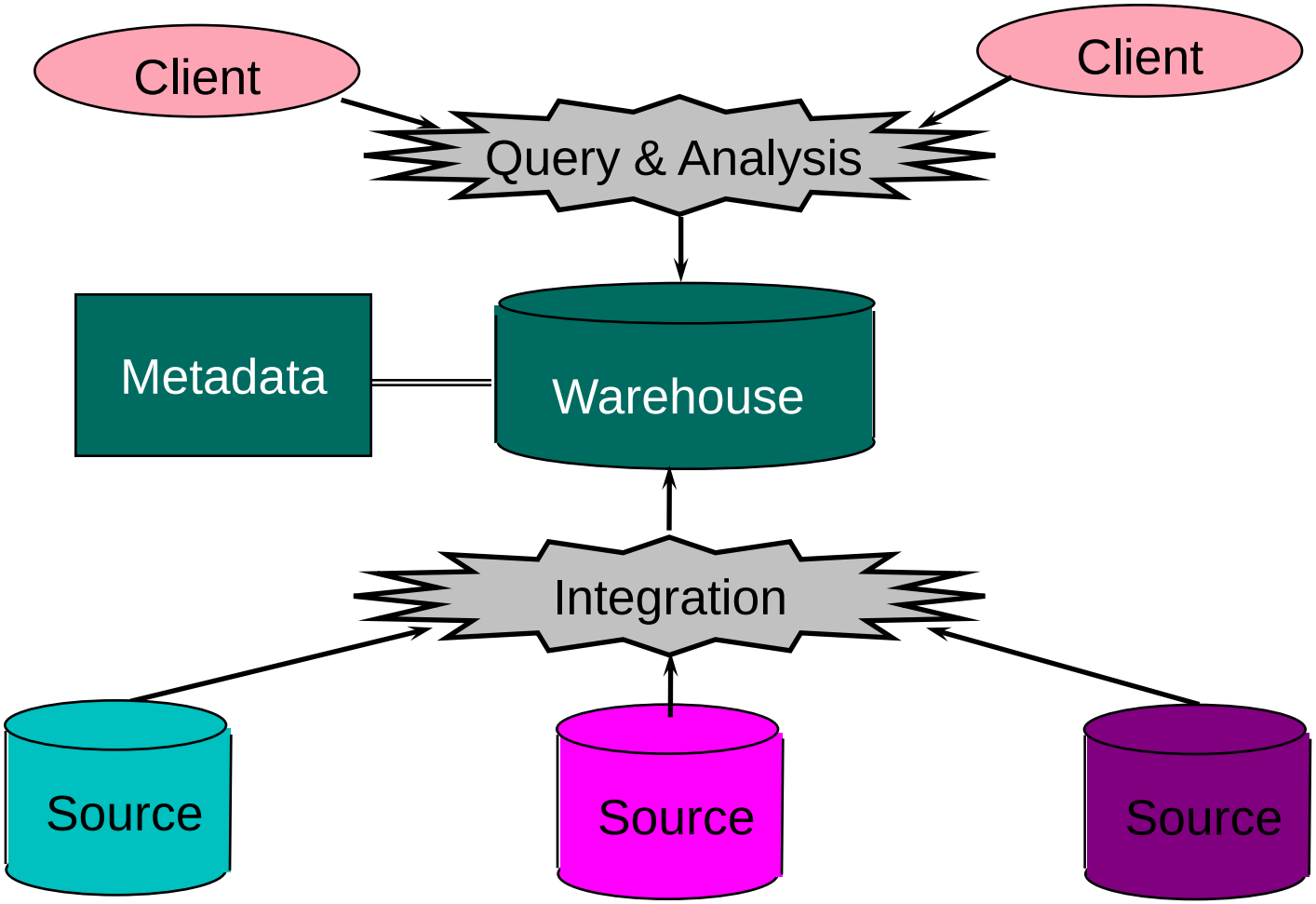
# What is a Warehouse?

- Collection of diverse data
  - subject (area) oriented, e.g., sales
  - aimed at executive, decision maker
  - often a copy of operational data
  - with value-added data (e.g., summaries, history)
  - integrated
  - time-varying: historical data, discovering trend
  - non-volatile: once in warehouse, data do not change

# What is a Warehouse?

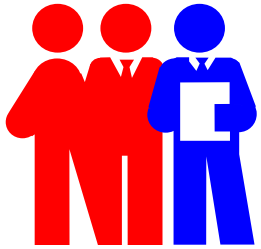
- Collection of tools
  - gathering data
  - cleansing & integrating data
  - querying, reporting, analyzing data
  - mining data
  - monitoring, administering warehouse

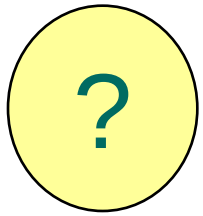
# Warehouse Architecture

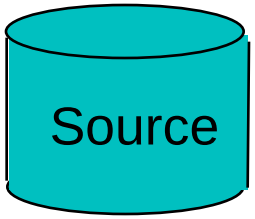


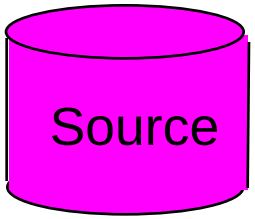
# Why a Warehouse?

- Two Approaches to Information Integration:
  - Warehouse (Eager)
  - Query-Driven (Lazy)

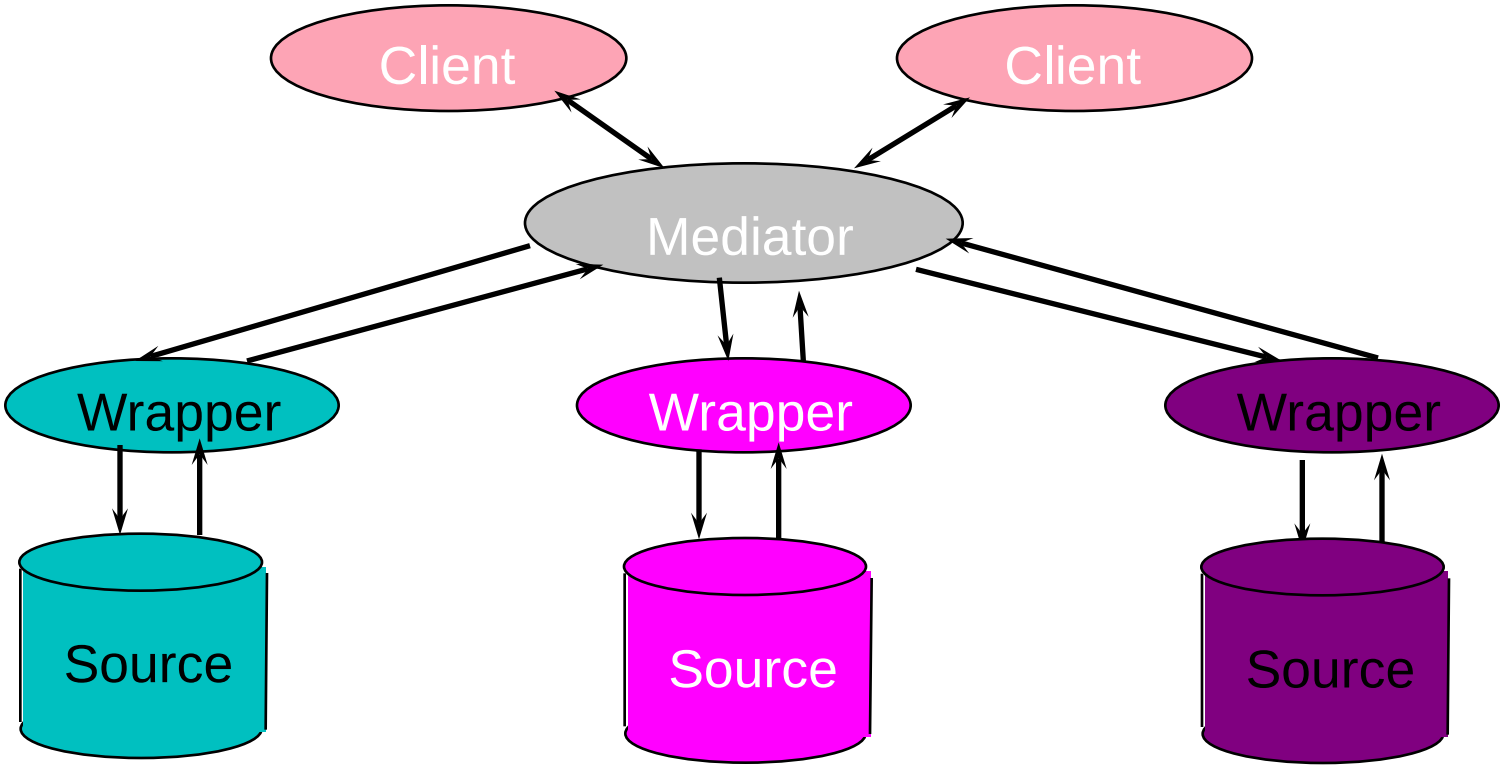




# Query-Driven Approach



# Advantages of Warehousing

- High query performance
- Local processing at sources unaffected
- Can operate when sources unavailable
- Extra information (e.g., pre-computed summary) at warehouse

# Disadvantages of Warehousing

- Decide what to store in advance
- Can only query data stored in warehouse
- Data get stale
- Must detect source changes & update warehouse

# Advantages of Query-Driven

- No need to copy data, less storage
- No need to purchase data
- More up-to-date data
- Query needs can be unknown

# Disadvantages of Query-Driven

- Inefficient/delay in query processing
  - source unreliable
  - slow network
  - expensive translation, filtering, merging
- Sources might not permit ad-hoc queries
  - Examples?

# OLTP vs. OLAP

- OLTP: On Line Transaction Processing
  - Describes processing at operational sites (order entry in POS/online, banking transactions, etc.)
- OLAP: On Line Analytical Processing
  - Describes processing (answering analytical queries: aggregation, rollup/drilldown, slice/dice, etc.) at warehouse

# OLTP vs. OLAP

## OLTP

- Mostly updates
- Many small transactions
- Mb-Tb of data
- Raw data
- Clerical users/clients/customers
- 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
- Historical data
- Query performance critical

# Data Marts

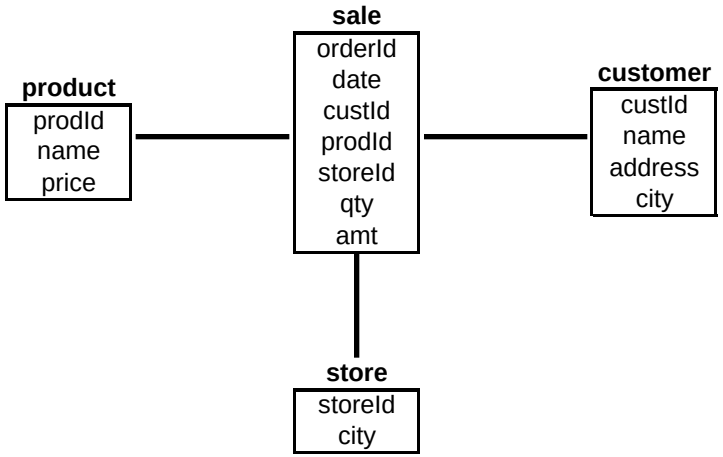
- Smaller warehouses
- Spans part of organization
  - e.g., marketing, sales, financial data marts
- Do not require enterprise-wide consensus
  - but long term integration problems?

# Warehouse Models & Operators

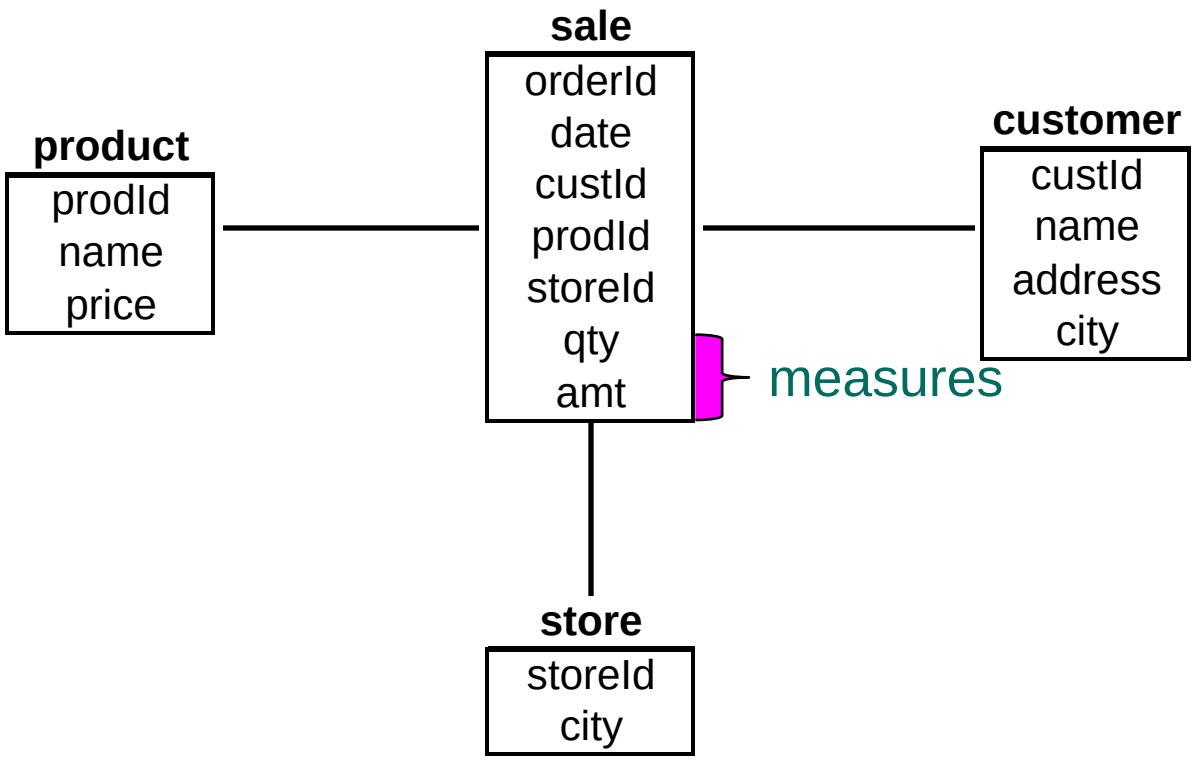
- Multi-dimensional data model (data cube)
  - Organizing measures/facts around different dimensions
- Relational implementation
  - Special schema design: star, snowflake, etc.
- Operators
  - roll-up, drill down
  - slice & dice
  - …

# Terms

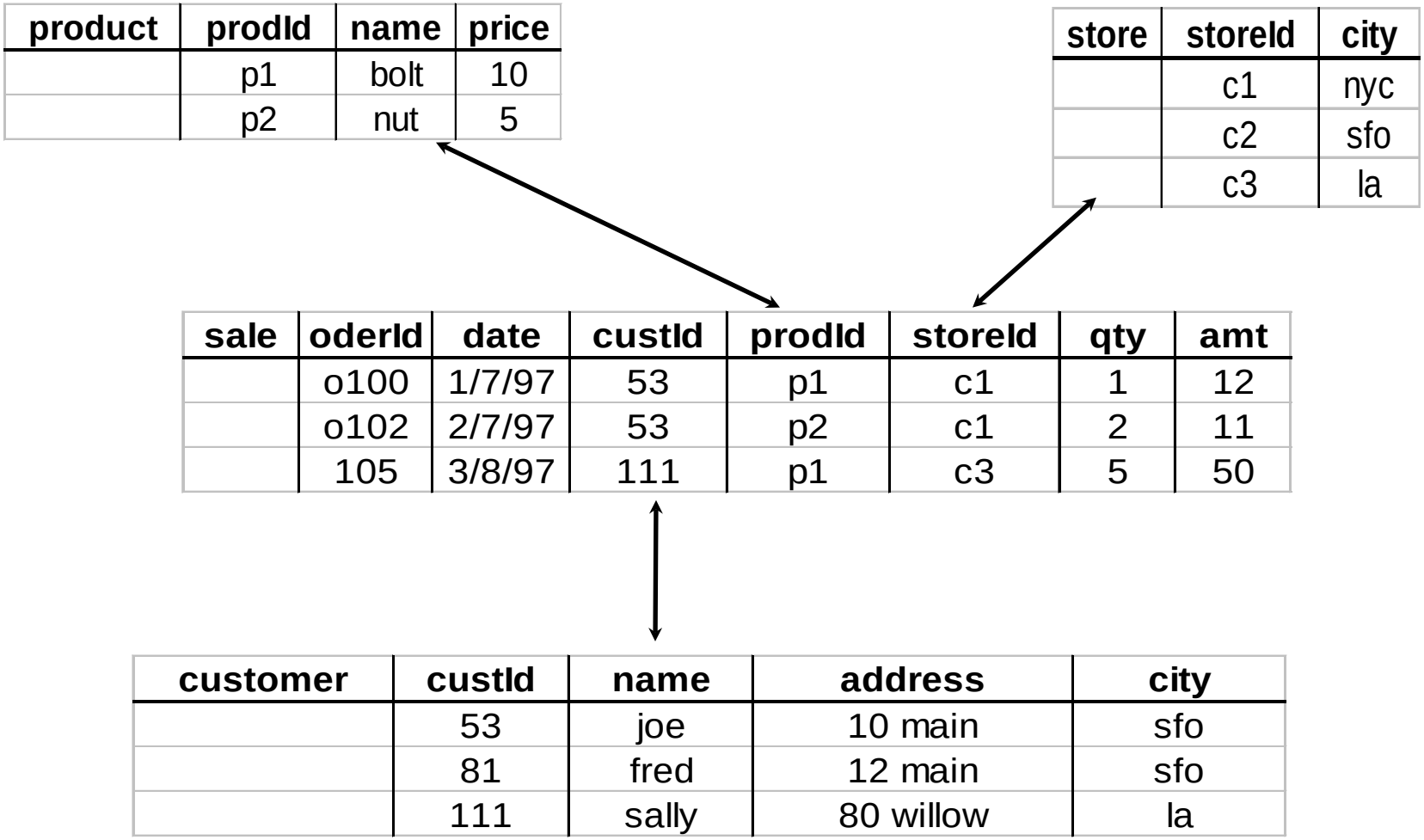
- Fact table
- Dimension tables
- Measures



# Star Schema


sale
orderId
date
custId
prodId
storeId
qty
amt
measures
product
prodId
name
price
customer
custId
name
address
city
store
storeId
city

# Star



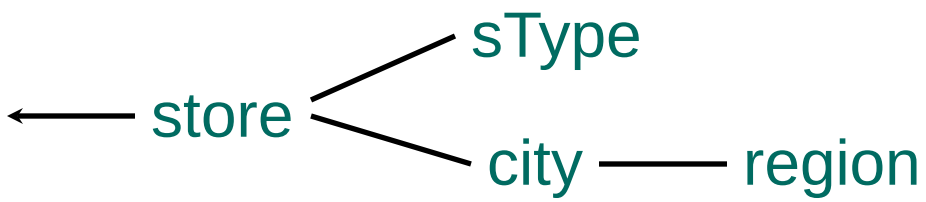
| product | prodId | name | price |
|---|---|---|---|
| | p1 | bolt | 10 |
| | p2 | nut | 5 |

| 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

# 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 | |

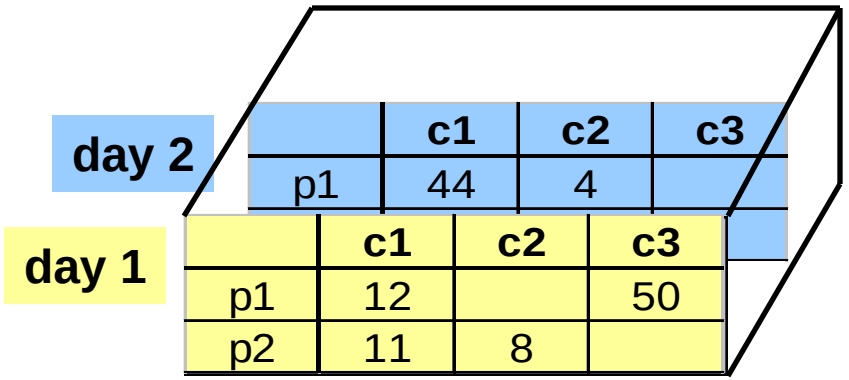
dimensions = 2

# 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:

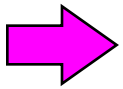



dimensions = 3

# 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 |

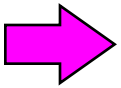


| ans | date | sum |
|---|---|---|
| | 1 | 81 |
| | 2 | 48 |


# Another Example

- Add up amounts by product, day
- In SQL: SELECT prodid, date, sum(amt)
  FROM SALE
  GROUP BY prodId, 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 | prodId | date | amt |
|---|---|---|---|
| | p1 | 1 | 62 |
| | p2 | 1 | 19 |
| | p1 | 2 | 48 |

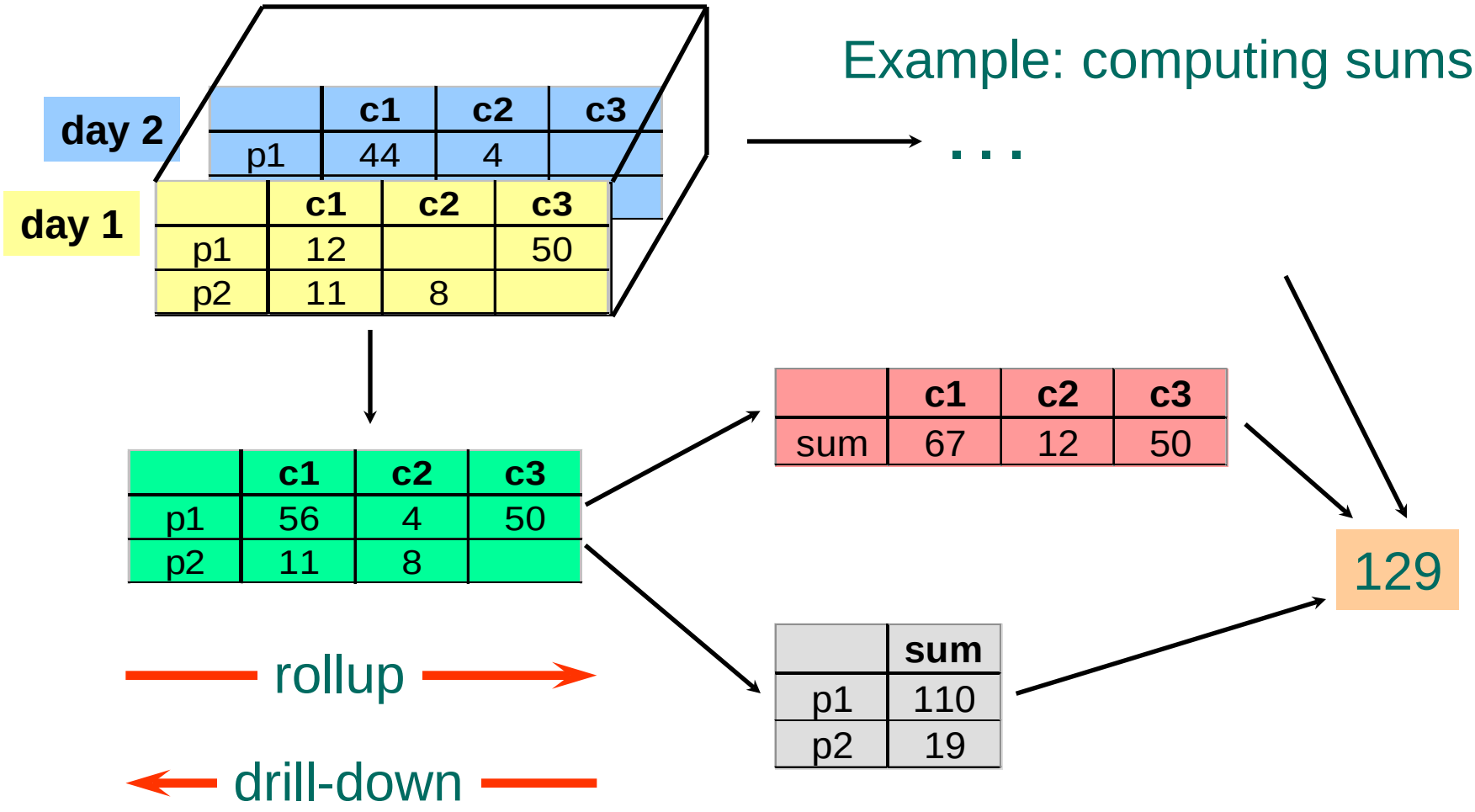
rollup →

← drill-down

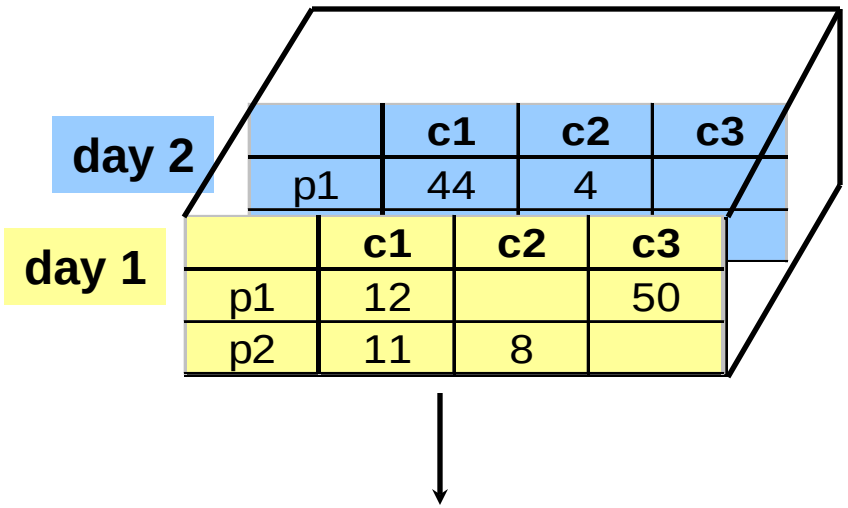
# Aggregates

- Operators: sum, count, max, min, avg
- May have “having” clause
- May also use dimension hierarchy, e.g.,
  - average by region (store dimension)
  - maximum by month (date dimension)

# Cube Aggregation



# Aggregation Using Hierarchies




| | region A | region B |
|---|---|---|
| p1 | 56 | 54 |
| p2 | 11 | 8 |

country

region

store

(store c1 in Region A;
stores c2, c3 in Region B)

# Implementing a Warehouse

- Extracting/monitoring: getting data from sources
- Integrating: cleaning, transforming, loading
- Processing: query processing, indexing, materialized view, ...
- Managing: metadata, performance monitoring, ...

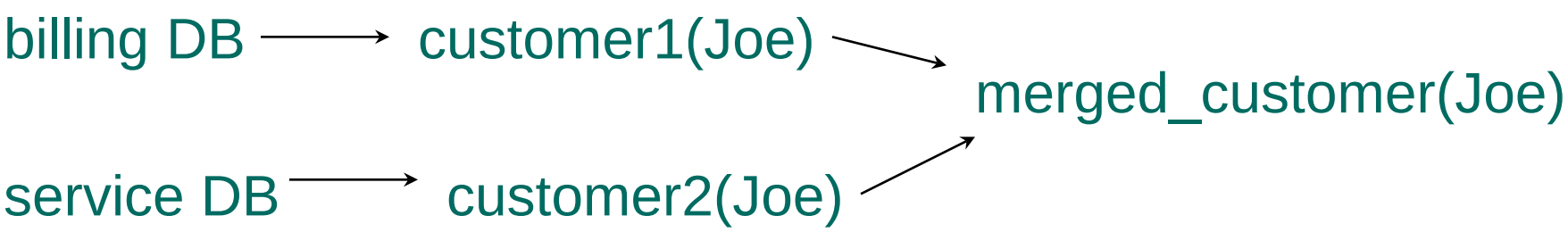
# Extracting & Monitoring

- Source Types: relational, flat file, XML, web sites, blogs, news feeds, …
- Incremental vs. complete refresh

| customer | id | name | address | city |
|---|---|---|---|---|
| | 53 | joe | 10 main | sfo |
| | 81 | fred | 12 main | sfo |
| | **111** | **sally** | **80 willow** | **la** |

# Data Cleaning

- Transformation (e.g., yen ⇨ dollars)
- Remove errors or inconsistency
  - E.g., 10 digit ssn, J. Smith vs Smith, John
- Fusion & entity resolution

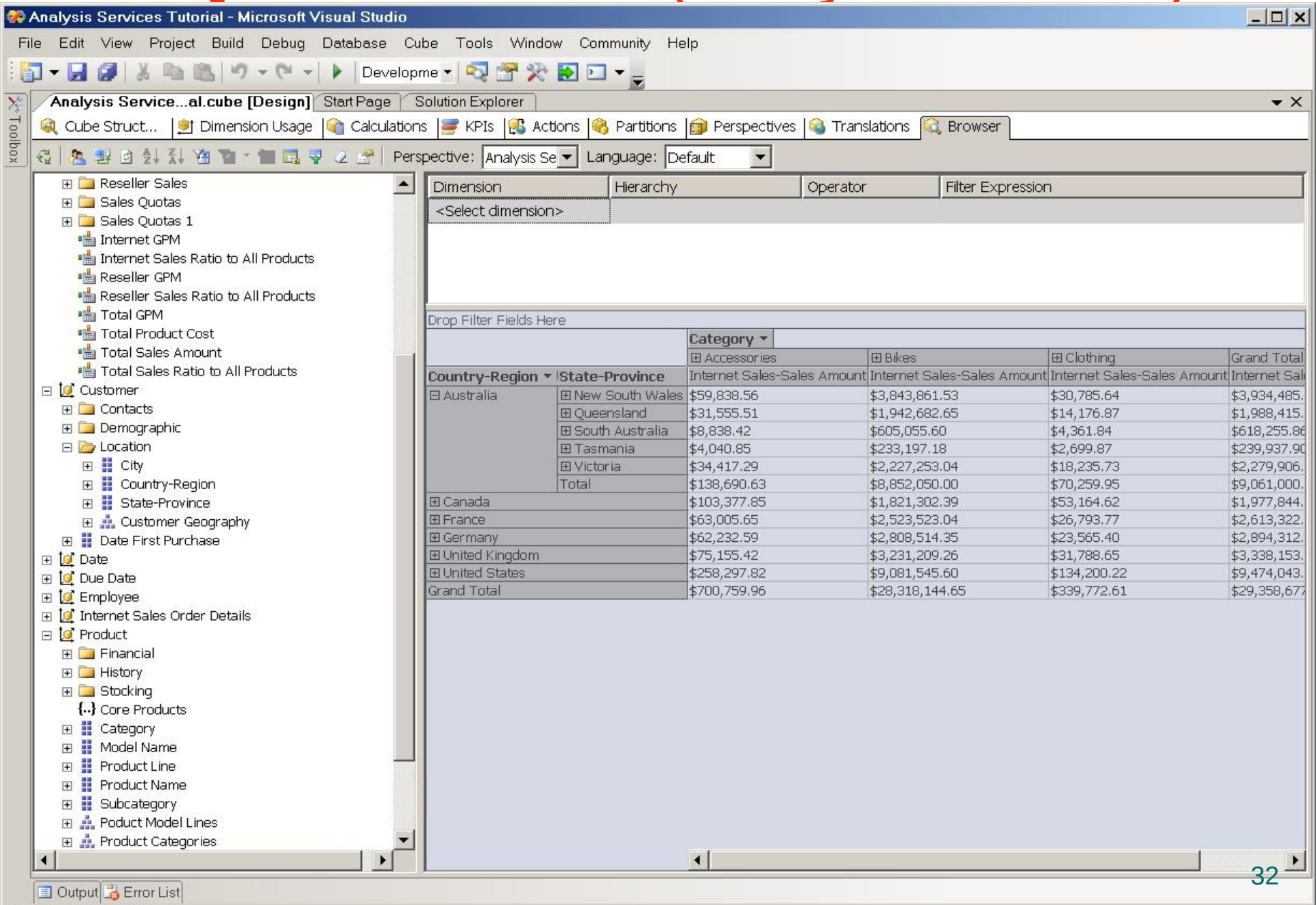



- Auditing: discover rules & relationships (e.g., discover FK/FK)

# 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
  - e.g., manage “reliable scripts” for cleaning & analyzing data

# SQL Server 2005 (Analysis Service)



# ETL Example

- Consider integrating data from two dealers
- Dealer 1 schema:
  - Cars(serialNo, model, color, autoTrans, navi)
  - Attributes starting from autoTrans are binary attributes for car options
  - E.g., autoTrans = 'Yes', if particular car has auto. transmission

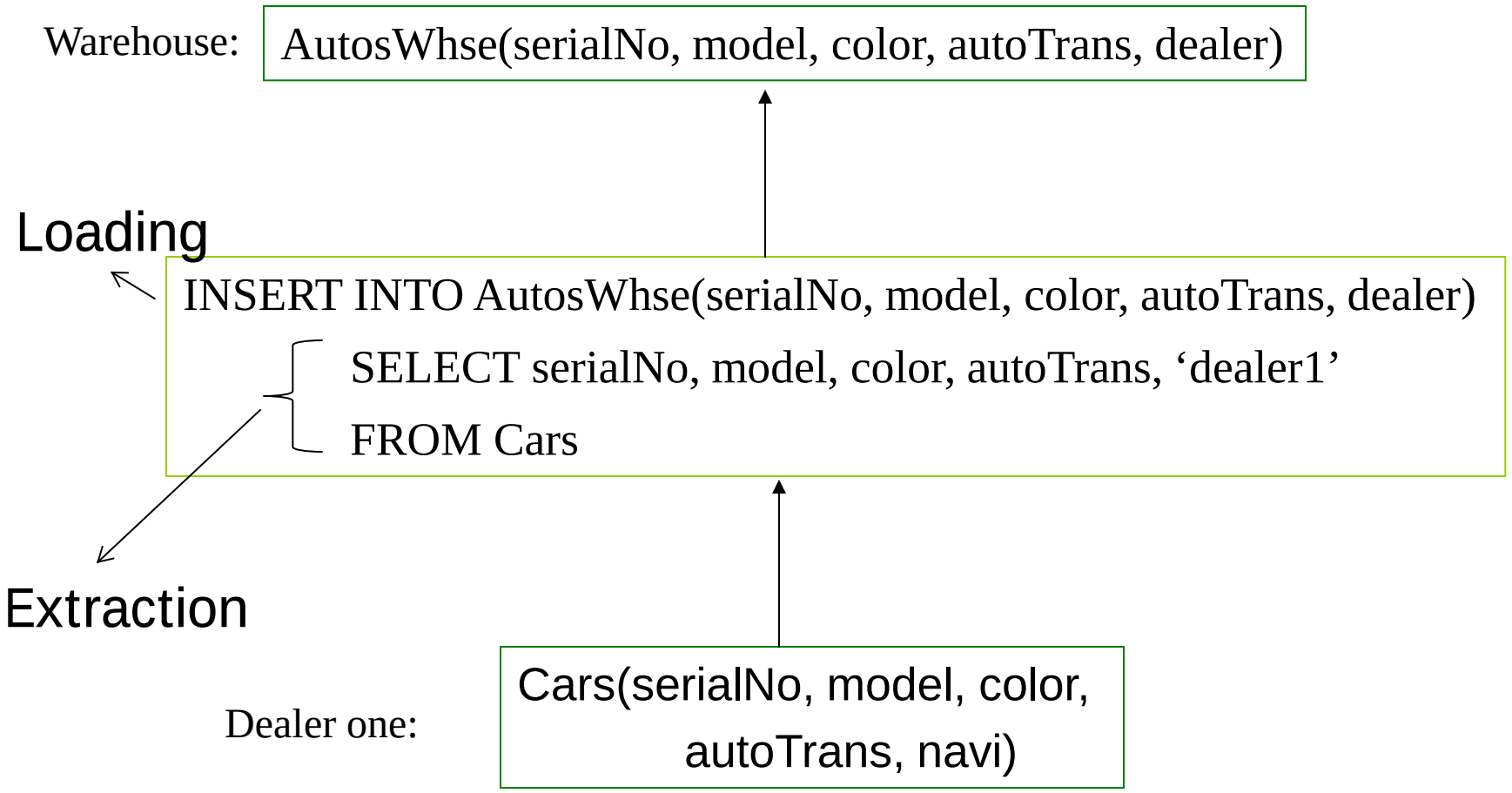
123, 'accord', 'silver', 'Yes', 'No'

- Dealer 2 schema:
  - Autos(serial, model, color)
  - Options(serial, option)

123, 'autoTrans'
123, 'navi'
...

- Warehouse schema:
  - AutosWhse(serialNo, model, color, autoTrans, dealer)

# Extract Data from Dealer One



# Exercise: Extracting from Dealer Two