Recall …

- Three approaches to information integration:
  - Federated databases
  - Data warehousing
  - Mediation

Warehousing

- Growing industry: $8 billion in 1998
- Range from desktop to huge:
  - Walmart: 900-CPU, 2,700 disk, 23TB Teradata system
- Lots of buzzwords, hype
  - slice & dice, rollup, MOLAP, pivot, ...

Outline

- What is a data warehouse?
- Why a warehouse?
- Models & operations
- Implementing a warehouse
- Future directions

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

What is a Warehouse?

- Collection of tools
  - gathering data
  - cleansing, integrating, ...
  - querying, reporting, analysis
  - data mining
  - monitoring, administering warehouse
Warehouse Architecture

Motivating Examples

- Forecasting
- Comparing performance of units
- Monitoring, detecting fraud
- Visualization

Why a Warehouse?

- Two Approaches:
  - Mediation (Lazy)
  - Warehouse (Eager)

Mediation Approach

Advantages of Warehousing

- High query performance
- Queries not visible outside warehouse
- Local processing at sources unaffected
- Can operate when sources unavailable
- Easier to query data not stored in a DBMS
- Extra information at warehouse
  - Modify, summarize (store aggregates)
  - Add historical information

Advantages of Mediation

- No need to copy data
  - less storage
  - no need to purchase data
- More up-to-date data
- Query needs can be unknown
- Only query interface needed at sources
- May be less draining on sources
OLTP vs. OLAP

- **OLTP**: On Line Transaction Processing
  - Describes processing at operational sites
- **OLAP**: On Line Analytical Processing
  - Describes processing at warehouse

**OLTP**
- Mostly updates
- Many small transactions
- Mb-Tb 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

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

```
<table>
<thead>
<tr>
<th>product</th>
<th>prodId</th>
<th>name</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>bolt</td>
<td>p1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>nut</td>
<td>p2</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>bolt</td>
<td>p3</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>nut</td>
<td>p4</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>store</th>
<th>storeId</th>
<th>city</th>
</tr>
</thead>
<tbody>
<tr>
<td>c1</td>
<td>1</td>
<td>nyc</td>
</tr>
<tr>
<td>c2</td>
<td>2</td>
<td>sfo</td>
</tr>
<tr>
<td>c3</td>
<td>3</td>
<td>la</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>sale</th>
<th>orderId</th>
<th>date</th>
<th>custId</th>
<th>prodId</th>
<th>storeId</th>
<th>qty</th>
<th>amt</th>
</tr>
</thead>
<tbody>
<tr>
<td>o100</td>
<td>1/7/97</td>
<td>53</td>
<td>p1</td>
<td>c1</td>
<td>1</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>o102</td>
<td>2/7/97</td>
<td>53</td>
<td>p2</td>
<td>c1</td>
<td>2</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>o105</td>
<td>3/8/97</td>
<td>111</td>
<td>p3</td>
<td>c3</td>
<td>5</td>
<td>50</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>customer</th>
<th>custId</th>
<th>name</th>
<th>address</th>
<th>city</th>
</tr>
</thead>
<tbody>
<tr>
<td>53</td>
<td>john</td>
<td>10 main st</td>
<td>nyc</td>
<td></td>
</tr>
<tr>
<td>81</td>
<td>fred</td>
<td>12 main st</td>
<td>sfo</td>
<td></td>
</tr>
<tr>
<td>111</td>
<td>sally</td>
<td>80 willow la</td>
<td>la</td>
<td></td>
</tr>
</tbody>
</table>
```

Star Schema
Terms

- Fact table
- Dimension tables
- Measures

Cube

Fact table view:

<table>
<thead>
<tr>
<th>sale</th>
<th>prodId</th>
<th>storeId</th>
<th>amt</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>c1</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>p2</td>
<td>c1</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>p1</td>
<td>c2</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>p1</td>
<td>c1</td>
<td>2</td>
<td>44</td>
</tr>
<tr>
<td>p2</td>
<td>c2</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

Multi-dimensional cube:

dimensions = 2

3-D Cube

Fact table view:

<table>
<thead>
<tr>
<th>sale</th>
<th>prodId</th>
<th>storeId</th>
<th>date</th>
<th>amt</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>c1</td>
<td>1</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>p2</td>
<td>c1</td>
<td>1</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>p1</td>
<td>c3</td>
<td>1</td>
<td>1</td>
<td>50</td>
</tr>
<tr>
<td>p2</td>
<td>c2</td>
<td>1</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>p1</td>
<td>c1</td>
<td>2</td>
<td>2</td>
<td>44</td>
</tr>
<tr>
<td>p1</td>
<td>c2</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

Multi-dimensional cube:

dimensions = 3

Aggregates

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

<table>
<thead>
<tr>
<th>sale</th>
<th>prodId</th>
<th>storeId</th>
<th>date</th>
<th>amt</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>c1</td>
<td>1</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>p2</td>
<td>c1</td>
<td>1</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>p1</td>
<td>c3</td>
<td>1</td>
<td>1</td>
<td>50</td>
</tr>
<tr>
<td>p2</td>
<td>c2</td>
<td>1</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>p1</td>
<td>c1</td>
<td>2</td>
<td>2</td>
<td>44</td>
</tr>
<tr>
<td>p1</td>
<td>c2</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

Aggregates

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

<table>
<thead>
<tr>
<th>sale</th>
<th>prodId</th>
<th>storeId</th>
<th>date</th>
<th>amt</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>c1</td>
<td>1</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>p2</td>
<td>c1</td>
<td>1</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>p1</td>
<td>c3</td>
<td>1</td>
<td>1</td>
<td>50</td>
</tr>
<tr>
<td>p2</td>
<td>c2</td>
<td>1</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>p1</td>
<td>c1</td>
<td>2</td>
<td>2</td>
<td>44</td>
</tr>
<tr>
<td>p1</td>
<td>c2</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

Add: 81
Another Example

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

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

Other Operations

- Time functions
  - e.g., time average
- Computed Attributes
  - e.g., commission = sales * rate
- Text Queries
  - e.g., find documents with words X AND B
  - e.g., rank documents by frequency of words X, Y, Z

Data Mining

- Clustering
- Association Rules

  Also:
  - Decision trees, time-series, ...

Clustering

Another Example: Text

- Each document is a vector
  - e.g., <100110...> contains words 1,4,5,...
- Clusters contain “similar” documents
- Useful for understanding, searching documents

Issues

- Given desired number of clusters?
- Finding “best” clusters
- Are clusters semantically meaningful?
  - e.g., “yuppies” cluster?
- Using clusters for disk storage
Association Rule Mining

sales records:

- Trend: Products p5, p8 often bought together
- Trend: Customer 12 likes product p9

Association Rule

- Rule: \{p_1, p_3, p_8\}
- Support: number of baskets where these products appear
- High-support set: support ≥ threshold s
- Problem: find all high support sets

Finding High-Support Pairs

- Baskets(basket, item)
- SELECT l.item, J.item, COUNT(l.basket) FROM Baskets I, Baskets J WHERE I.basket = J.basket AND l.item < J.item GROUP BY l.item, J.item HAVING COUNT(l.basket) >= s;

Example

<table>
<thead>
<tr>
<th>basket</th>
<th>item</th>
<th>basket</th>
<th>item2</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>p2</td>
<td>t2</td>
<td>p6</td>
</tr>
<tr>
<td>t1</td>
<td>p5</td>
<td>t1</td>
<td>p6</td>
</tr>
<tr>
<td>t1</td>
<td>p8</td>
<td>t1</td>
<td>p6</td>
</tr>
<tr>
<td>t2</td>
<td>p8</td>
<td>t2</td>
<td>p11</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

check if count ≥ s

Issues

- Performance for size 2 rules
- Performance for size k rules

Implementing a Warehouse

- Monitoring: Sending data from sources
- Integrating: Loading, cleansing,...
- Processing: Query processing, indexing, ...
- Managing: Metadata, Design, ...
Monitoring

- Source Types: relational, flat file, IMS, WWW, news-wire, ...
- Incremental vs. Periodic Refresh

<table>
<thead>
<tr>
<th>customer</th>
<th>name</th>
<th>address</th>
<th>city</th>
</tr>
</thead>
<tbody>
<tr>
<td>53</td>
<td>Joe</td>
<td>10 main</td>
<td>Sfo</td>
</tr>
<tr>
<td>81</td>
<td>Fred</td>
<td>12 main</td>
<td>Sfo</td>
</tr>
<tr>
<td>111</td>
<td>Sally</td>
<td>80 Willow</td>
<td>La</td>
</tr>
</tbody>
</table>

Monitoring Techniques

- Periodic snapshots
- Database triggers
- Log shipping
- Data shipping (replication service)
- Transaction shipping
- Polling (queries to source)
- Screen scraping
- Application level monitoring

Advantages & Disadvantages!!

Monitoring Issues

- Frequency
  - periodic: daily, weekly, ...
  - triggered: on “big” change, lots of changes, ...
- Data transformation
  - convert data to uniform format
  - remove & add fields (e.g., add date to get history)
- Standards (e.g., ODBC)
- Gateways

Integration

- Data Cleaning
- Data Loading
- Derived Data

Data Cleaning

- Migration (e.g., yen $ dollars)
- Scrubbing: use domain-specific knowledge (e.g., social security numbers)
- Fusion (e.g., mail list, customer merging)
- Auditing: discover rules & relationships (like data mining)

Loading Data

- Incremental vs. periodic refresh
- Off-line vs. on-line
- Frequency of loading
  - At night, 1x a week/month, continuously
  - Parallel/Partitioned load
Derived Data

- Derived Warehouse Data
  - indexes
  - aggregates
  - materialized views (next slide)
- When to update derived data?
- Incremental vs. periodic refresh

Materialized Views

- Define new warehouse relations using SQL expressions

```
<table>
<thead>
<tr>
<th>sale</th>
<th>prodId</th>
<th>storeId</th>
<th>date</th>
<th>amt</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>c1</td>
<td>1</td>
<td>1</td>
<td>1.1</td>
</tr>
<tr>
<td>p2</td>
<td>c2</td>
<td>1</td>
<td>5</td>
<td>9.0</td>
</tr>
<tr>
<td>p3</td>
<td>c2</td>
<td>2</td>
<td>1.1</td>
<td>8</td>
</tr>
<tr>
<td>p2</td>
<td>c1</td>
<td>2</td>
<td>4</td>
<td>4.4</td>
</tr>
<tr>
<td>p1</td>
<td>c1</td>
<td>2</td>
<td>4.4</td>
<td>4</td>
</tr>
</tbody>
</table>
```

- Materialized Views:
  - Define new warehouse relations using SQL expressions.

Processing

- ROLAP servers vs. MOLAP servers
- Index Structures
- What to Materialize?
- Algorithms

ROLAP Server

- Relational OLAP Server

```
<table>
<thead>
<tr>
<th>sale</th>
<th>prodId</th>
<th>storeId</th>
<th>date</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>c1</td>
<td>1</td>
<td>1</td>
<td>62</td>
</tr>
<tr>
<td>p2</td>
<td>c1</td>
<td>1</td>
<td>1</td>
<td>19</td>
</tr>
<tr>
<td>p1</td>
<td>c2</td>
<td>2</td>
<td>4</td>
<td>48</td>
</tr>
</tbody>
</table>
```

- ROLAP Server

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, r37
    - Answer is intersection: r18

Bit Maps

- 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: 0100000000
  - Good if domain cardinality small
  - Bit vectors can be compressed

Join

- “Combine” SALE, PRODUCT relations
  - In SQL: SELECT * FROM SALE, PRODUCT...

Join Indexes

- Product: id, item, price, index
- Sale: rId, prodId, storeId, date, amt

<table>
<thead>
<tr>
<th>joinTo</th>
<th>prodId</th>
<th>name</th>
<th>price</th>
<th>storeId</th>
<th>date</th>
<th>amt</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>c2</td>
<td>2</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>p2</td>
<td>c1</td>
<td>11</td>
<td>11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>p2</td>
<td>c2</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>p3</td>
<td>c1</td>
<td>44</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>sale</th>
<th>rId</th>
<th>prodId</th>
<th>storeId</th>
<th>date</th>
<th>amt</th>
</tr>
</thead>
<tbody>
<tr>
<td>r1</td>
<td>p1</td>
<td>c1</td>
<td>1</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>r2</td>
<td>p2</td>
<td>c1</td>
<td>1</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>r3</td>
<td>p1</td>
<td>c3</td>
<td>1</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>r4</td>
<td>p1</td>
<td>c3</td>
<td>1</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>r5</td>
<td>p1</td>
<td>c1</td>
<td>2</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>r6</td>
<td>p1</td>
<td>c2</td>
<td>2</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>
What to Materialize?

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

<table>
<thead>
<tr>
<th>Day</th>
<th>City</th>
<th>Product</th>
<th>Sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Materialization Factors

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

Cube Aggregates Lattice

Managing

- Metadata
- Warehouse Design
- Tools

Metadata

- Administrative
  - definition of sources, tools, ...
  - schemas, dimension hierarchies, ...
  - rules for extraction, cleaning, ...
  - refresh, purging policies
  - user profiles, access control, ...

Metadata

- Business
  - business terms & definition
  - data ownership, charging
- Operational
  - data lineage
  - data currency (e.g., active, archived, purged)
  - use stats, error reports, audit trails
### 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

### Future Directions
- Better performance
- Larger warehouses
- Easier to use
- What are companies & research labs working on?

### Research (1)
- Incremental Maintenance
- Data Consistency
- Data Expiration
- Recovery
- Data Quality
- Error Handling (Back Flush)

### Research (2)
- Rapid Monitor Construction
- Temporal Warehouses
- Materialization & Index Selection
- Data Fusion
- Data Mining
- Integration of Text & Relational Data
Conclusions

- Massive amounts of data and complexity of queries will push limits of current warehouses
- Need better systems:
  - easier to use
  - provide quality information