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

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
Alternative to Warehousing

- Two Approaches:
  - Query-Driven (Lazy)
  - Warehouse (Eager)

Query-Driven Approach

Advantages of Warehousing

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

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
- Only query interface needed at sources
- May be less draining on sources

Warehouse Models & Operators

- Data Models
  - relational
  - cubes
- Operators

Star

<table>
<thead>
<tr>
<th>product</th>
<th>price</th>
<th>store</th>
<th>custid</th>
<th>item</th>
<th>qty</th>
<th>amt</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>10</td>
<td>c1</td>
<td>53</td>
<td>pl</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>p2</td>
<td>5</td>
<td>c2</td>
<td>55</td>
<td>nut</td>
<td>2</td>
<td>11</td>
</tr>
<tr>
<td>p3</td>
<td>18</td>
<td>c3</td>
<td>111</td>
<td>pl</td>
<td>5</td>
<td>50</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>sale</th>
<th>date</th>
<th>custid</th>
<th>item</th>
<th>qty</th>
<th>amt</th>
</tr>
</thead>
<tbody>
<tr>
<td>110</td>
<td>1/7/97</td>
<td>53</td>
<td>pl</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>102</td>
<td>2/7/97</td>
<td>53</td>
<td>pl</td>
<td>2</td>
<td>11</td>
</tr>
<tr>
<td>105</td>
<td>3/8/97</td>
<td>111</td>
<td>pl</td>
<td>5</td>
<td>50</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>jake</td>
<td>j 10 main</td>
<td>san</td>
<td></td>
</tr>
<tr>
<td>81</td>
<td>fred</td>
<td>12 main</td>
<td>san</td>
<td></td>
</tr>
<tr>
<td>111</td>
<td>sally</td>
<td>80 wilson</td>
<td>la</td>
<td></td>
</tr>
</tbody>
</table>
Star Schema

**Terms**
- Fact table
- Dimension tables
- Measures

**Dimension Hierarchies**

**Cube**

**Operators**
- Traditional
  - Selection
  - Aggregation
  - ...
- Analysis
  - Clean data
  - Find trends
  - ...
- Relational
  - Cube
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>12</td>
<td>12</td>
</tr>
<tr>
<td>p2</td>
<td>c1</td>
<td>1</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>p1</td>
<td>c3</td>
<td>1</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>p2</td>
<td>c2</td>
<td>1</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>p1</td>
<td>c1</td>
<td>2</td>
<td>44</td>
<td>44</td>
</tr>
<tr>
<td>p1</td>
<td>c2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

\[\text{ans} = 112\]

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

<table>
<thead>
<tr>
<th>ans</th>
<th>date</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>81</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>48</td>
</tr>
</tbody>
</table>

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

Another Example

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

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

Cube Aggregation

Example: computing sums

<table>
<thead>
<tr>
<th>day 2</th>
<th>c1</th>
<th>c2</th>
<th>c3</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>12</td>
<td>8</td>
<td>50</td>
<td>129</td>
</tr>
<tr>
<td>p2</td>
<td>11</td>
<td>11</td>
<td>8</td>
<td>129</td>
</tr>
</tbody>
</table>

- rollup
- drill-down

Cube Operators

Example: computing sums

<table>
<thead>
<tr>
<th>day 2</th>
<th>c1</th>
<th>c2</th>
<th>c3</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>10</td>
<td></td>
<td></td>
<td>129</td>
</tr>
<tr>
<td>p2</td>
<td>10</td>
<td></td>
<td></td>
<td>129</td>
</tr>
</tbody>
</table>

- rollup
- drill-down
Extended Cube

Aggregation Using Hierarchies

Data Analysis

- Decision Trees
- Clustering
- Association Rules

Decision Trees

Example:
- Conducted survey to see what customers were interested in new model car
- Want to select customers for advertising campaign

One Possibility

Another Possibility
Issues

- Decision tree cannot be “too deep”
  - would not have statistically significant amounts of data for lower decisions
- Need to select tree that most reliably predicts outcomes

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

- Trend: Products p5, p8 often bought together
- Trend: Customer 12 likes product p9
**Association Rule**
- **Rule:** \( \{ p_1, p_2, p_3 \} \)
- **Support:** number of baskets where these products appear
- **High-support set:** support \( \geq \) threshold \( s \)
- **Problem:** find all high support sets

**Implementation Issues**
- ETL (Extraction, transformation, loading)
  - Getting data to the warehouse
  - Entity Resolution
  - What to materialize?
  - Efficient Analysis
    - Association rule mining
    - ...

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

**ETL: Data Cleaning**
- Migration (e.g., yen \( \Rightarrow \) dollars)
- Scrubbing: use domain-specific knowledge (e.g., social security numbers)
- Fusion (e.g., mail list, customer merging)
  - billing DB \( \rightarrow \) customer1(Joe) \( \rightarrow \) merged_customer(Joe)
  - service DB \( \rightarrow \) customer2(Joe)
- Auditing: discover rules & relationships (like data mining)

**More details: Entity Resolution**

**Applications**
- comparison shopping
- mailing lists
- classified ads
- customer files
- counter-terrorism
Why is ER Challenging?
- Huge data sets
- No unique identifiers
- Lots of uncertainty
- Many ways to skin the cat

Taxonomy: Pairwise vs Global
- Decide if $r, s$ match only by looking at $r, s$?
- Or need to consider more (all) records?

Global matching complicates things a lot!
- e.g., change decision as new records arrive

Taxonomy: Outcome
- Partition of records
  - e.g., comparison shopping
- Merged records

Taxonomy: Record Reuse
- One record related to multiple entities?
Taxonomy: Record Reuse

- Partitions
  \[ r \quad s \quad t \]
- Merges
  \[ r \quad rs \quad s \quad st \quad t \]

- Record reuse ➔ complex and expensive!

Taxonomy: Multiple Entity Types

- Person 1
  - Brother
  - Member
- Person 2
  - Member
  - Business
  - Organization A
- Organization B

Taxonomy: Exact vs Approximate

- Products
  - Cameras
  - CDs
  - Books
  - ER
  - Resolved cameras
  - Resolved CDs
  - Resolved books

- ER
- Terrorists
  - Sort by age
  - B Cooper 30
  - Match against ages 25-35
Implementation Issues

- ETL (Extraction, transformation, loading)
  - Getting data to the warehouse
  - Entity Resolution
- What to materialize?
- Efficient Analysis
  - Association rule mining
  - ...

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

<table>
<thead>
<tr>
<th>states</th>
<th>cities</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>c1</td>
</tr>
<tr>
<td></td>
<td>CA</td>
</tr>
<tr>
<td>city</td>
<td>c2</td>
</tr>
<tr>
<td></td>
<td>NY</td>
</tr>
</tbody>
</table>

Dimension Hierarchies
Interesting Hierarchy

- all
- years
- weeks
- quarters
- months
- days

Implementation Issues

- ETL (Extraction, transformation, loading)
  - Getting data to the warehouse
  - Entity Resolution
- What to materialize?
- Efficient Analysis
  - Association rule mining
  - ...

Finding High-Support Pairs

- Baskets(basket, item)
- \[ \text{SELECT I.item, J.item, COUNT(I.basket)} \]
  \[ \text{FROM Baskets I, Baskets J} \]
  \[ \text{WHERE I.basket = J.basket AND} \]
  \[ I.item < J.item \]
  \[ \text{GROUP BY I.item, J.item} \]
  \[ \text{HAVING COUNT(I.basket) >= s;} \]

Example

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

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

 WHY? check if count \( \geq s \)
### Issues

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

### Association Rules

- How do we perform rule mining efficiently?
- Observation: If set $X$ has support $t$, then each $X$ subset must have at least support $t$

### Algorithm for 2-Sets

1. Find OK products
   - those appearing in $s$ or more baskets
2. Find high-support pairs
   - using only OK products

```sql
INSERT INTO okBaskets(basket, item)
SELECT basket, item
FROM Baskets
GROUP BY item
HAVING COUNT(basket) >= s;
```
Algorithm for 2-Sets

- INSERT INTO okBaskets(basket, item)
  SELECT basket, item
  FROM Baskets
  GROUP BY item
  HAVING COUNT(basket) >= s;
- Perform mining on okBaskets
  SELECT l.item, r.item, COUNT(l.basket)
  FROM okBaskets l, okBaskets r
  WHERE l.basket = r.basket AND
  l.item < r.item
  GROUP BY l.item, r.item
  HAVING COUNT(l.basket) >= s;

Counting Efficiently

- One way:

<table>
<thead>
<tr>
<th>basket</th>
<th>item</th>
<th>J.item</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>p5</td>
<td>p8</td>
</tr>
<tr>
<td>12</td>
<td>p8</td>
<td>p11</td>
</tr>
<tr>
<td>13</td>
<td>p2</td>
<td>p3</td>
</tr>
<tr>
<td>13</td>
<td>p8</td>
<td>p8</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

  threshold = 3

- Another way:

<table>
<thead>
<tr>
<th>basket</th>
<th>item</th>
<th>J.item</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>p5</td>
<td>p8</td>
</tr>
<tr>
<td>12</td>
<td>p8</td>
<td>p11</td>
</tr>
<tr>
<td>13</td>
<td>p2</td>
<td>p3</td>
</tr>
<tr>
<td>13</td>
<td>p8</td>
<td>p8</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

  threshold = 3

- One way:

<table>
<thead>
<tr>
<th>basket</th>
<th>item</th>
<th>J.item</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>p5</td>
<td>p8</td>
</tr>
<tr>
<td>12</td>
<td>p8</td>
<td>p11</td>
</tr>
<tr>
<td>13</td>
<td>p2</td>
<td>p3</td>
</tr>
<tr>
<td>13</td>
<td>p8</td>
<td>p8</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

  sort

<table>
<thead>
<tr>
<th>basket</th>
<th>item</th>
<th>J.item</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>p2</td>
<td>p8</td>
</tr>
<tr>
<td>12</td>
<td>p8</td>
<td>p11</td>
</tr>
<tr>
<td>13</td>
<td>p8</td>
<td>p8</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

  threshold = 3

- Another way:

<table>
<thead>
<tr>
<th>basket</th>
<th>item</th>
<th>J.item</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>p5</td>
<td>p8</td>
</tr>
<tr>
<td>12</td>
<td>p8</td>
<td>p11</td>
</tr>
<tr>
<td>13</td>
<td>p2</td>
<td>p3</td>
</tr>
<tr>
<td>13</td>
<td>p8</td>
<td>p8</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

  count & remove

<table>
<thead>
<tr>
<th>basket</th>
<th>item</th>
<th>J.item</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>p8</td>
<td>p8</td>
</tr>
<tr>
<td>5</td>
<td>p12</td>
<td>p18</td>
</tr>
<tr>
<td>1</td>
<td>p17</td>
<td>p22</td>
</tr>
<tr>
<td>2</td>
<td>p21</td>
<td>p23</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

  keep counter array in memory

threshold = 3
Counting Efficiently

- Another way: threshold = 3

Yet Another Way

- Another way: threshold = 3

false positive

keep counter array in memory

in-memory hash table

in-memory hash table

in-memory hash table

in-memory hash table


### Discussion

- Hashing scheme: 2 (or 3) scans of data
- Sorting scheme: requires a sort!
- Hashing works well if few high-support pairs and many low-support ones

### Implementation Issues

- ETL (Extraction, transformation, loading)
  - Getting data to the warehouse
  - Entity Resolution
- What to materialize?
- Efficient Analysis
  - Association rule mining
  - ...

### Extra: Data Mining in the InfoLab

**Recommendations in CourseRank**

<table>
<thead>
<tr>
<th>quarters</th>
<th>user</th>
<th>q1</th>
<th>q2</th>
<th>q3</th>
<th>q4</th>
</tr>
</thead>
<tbody>
<tr>
<td>u1</td>
<td>a:</td>
<td>5</td>
<td>b:</td>
<td>5</td>
<td>d:</td>
</tr>
<tr>
<td>u2</td>
<td>a:</td>
<td>1</td>
<td>e:</td>
<td>2</td>
<td>d:</td>
</tr>
<tr>
<td>u3</td>
<td>g:</td>
<td>4</td>
<td>h:</td>
<td>2</td>
<td>e:</td>
</tr>
<tr>
<td>u4</td>
<td>b:</td>
<td>2</td>
<td>g:</td>
<td>4</td>
<td>h:</td>
</tr>
<tr>
<td>u5</td>
<td>a:</td>
<td>5</td>
<td>g:</td>
<td>4</td>
<td>e:</td>
</tr>
</tbody>
</table>

u3 and u4 are similar to u

Recommend h
Extra: Data Mining in the InfoLab

Recommendations in CourseRank

<table>
<thead>
<tr>
<th>user</th>
<th>q1</th>
<th>q2</th>
<th>q3</th>
<th>q4</th>
</tr>
</thead>
<tbody>
<tr>
<td>u1</td>
<td>a</td>
<td>d</td>
<td>a</td>
<td>5</td>
</tr>
<tr>
<td>u2</td>
<td>a</td>
<td>e</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>u3</td>
<td>g</td>
<td>4</td>
<td>h</td>
<td>2</td>
</tr>
<tr>
<td>u4</td>
<td>b</td>
<td>2</td>
<td>g</td>
<td>4</td>
</tr>
<tr>
<td>v</td>
<td>a</td>
<td>b</td>
<td>e</td>
<td>4</td>
</tr>
</tbody>
</table>

Recommend d (and f, h)

Sequence Mining

- Given a set of transcripts, use \( Pr[x|a] \) to predict if \( x \) is a good recommendation given user has taken \( a \).
- Two issues...

Pr\( [x|a] \) Not Quite Right

<table>
<thead>
<tr>
<th>transcript</th>
<th>containing</th>
<th>target user's transcript: [ ... a ... ] unknown</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>a</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>a</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>a, x</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>a, x, a</td>
<td></td>
</tr>
</tbody>
</table>

Pr\( [x|a] = 2/3 \)
Pr\( [x|a^{-x}] = 1/2 \)

User Has Taken >= 1 Course

- User has taken \( T = \{ a, b, c \} \)
- Need \( Pr[x|T^{-x}] \)
- Approximate as \( Pr[x|a^{-x} \land b^{-x} \land c^{-x}] \)
- Expensive to compute, so...

CourseRank User Study

![CourseRank User Study Chart]