Data Analytics: Data Warehousing, Data Mining

Decision Support Systems

- **Decision-support systems** are used to make business decisions, often based on data collected by on-line transaction processing systems

- Examples of business decisions:
  - What items to stock?
  - What insurance premium to change?
  - To whom to send advertisements?

- Examples of data used for making decisions
  - Retail sales transaction details
  - Customer profiles (income, age, gender, etc.)

Decision-Support Systems: Overview

- **Data analysis** tasks are simplified by specialized tools and SQL extensions
  - Example tasks
    - For each product category and each region, what were the total sales in the last quarter and how do they compare with the same quarter last year
    - As above, for each product category and each customer category

- **Statistical analysis** packages (e.g.,: S++) can be interfaced with databases
  - Statistical analysis is a large field, but not covered here

- **Data mining** seeks to discover knowledge automatically in the form of statistical rules and patterns from large databases

- A **data warehouse** archives information gathered from multiple sources, and stores it under a unified schema, at a single site
  - Important for large businesses that generate data from multiple divisions, possibly at multiple sites
  - Data may also be purchased externally

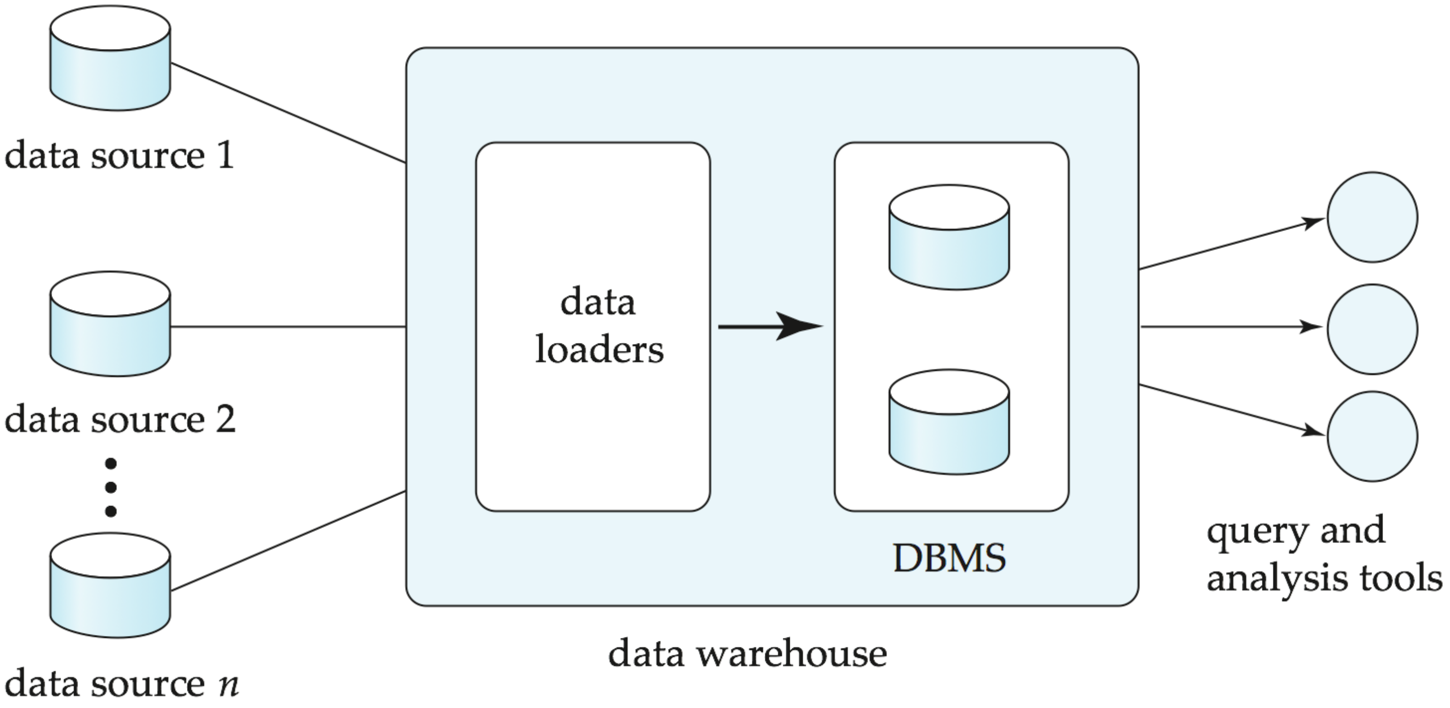
Data Warehousing

- Data sources often store only current data, not historical data

- Corporate decision making requires a unified view of all organizational data, including historical data

- A **data warehouse** is a repository (archive) of information gathered from multiple sources, stored under a unified schema, at a single site
  - Greatly simplifies querying, permits study of historical trends
  - Shifts decision support query load away from transaction processing systems

Data Warehousing



Design Issues

When and how to gather data

- **Source driven architecture**: Data sources transmit new information to warehouse, either continuously or periodically (e.g., at night)
- **Destination driven architecture**: Warehouse periodically requests new information from data sources
- Keeping warehouse exactly synchronized with data sources (e.g., using two-phase commit) is too expensive
  - Usually OK to have slightly out-of-date data at warehouse
  - Data/updates are periodically downloaded form online transaction processing (OLTP) systems

What schema to use

- Schema integration

More Warehouse Design Issues

- *Data cleansing*
  - E.g., correct mistakes in addresses (misspellings, zip code errors)
  - **Merge** address lists from different sources and **purge** duplicates

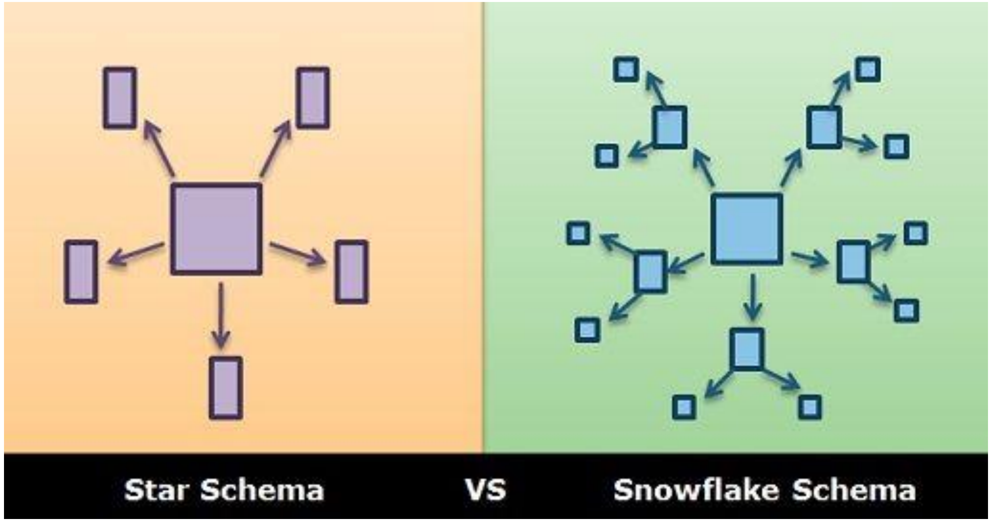
- *How to propagate updates*
  - Warehouse schema may be a (materialized) view of schema from data sources

- *What data to summarize*
  - Raw data may be too large to store on-line
  - Aggregate values (totals/subtotals) often suffice
  - Queries on raw data can often be transformed by query optimizer to use aggregate values

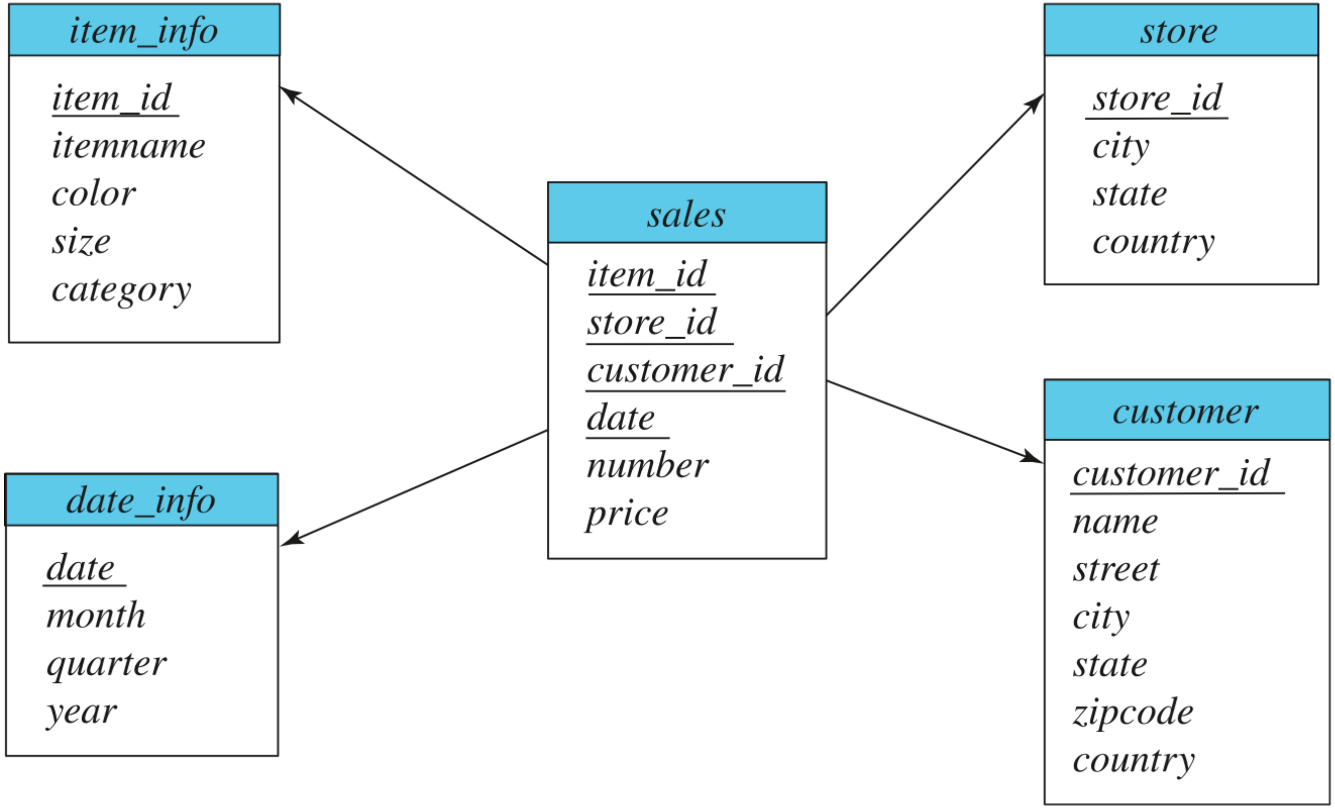
Warehouse Schemas

- Dimension values are usually encoded using small integers and mapped to full values via dimension tables

- Resultant schema is called a **star schema**
  - More complicated schema structures
    - **Snowflake schema**: Multiple levels of dimension tables
    - **Constellation**: Multiple fact tables



Data Warehouse Schema



item_info

item_id
itemname
color
size
category

sales

item_id
store_id
customer_id
date
number
price

store

store_id
city
state
country

date_info

date
month
quarter
year

customer

customer_id
name
street
city
state
zipcode
country

Data Mining

- Data mining is the process of semi-automatically analyzing large databases to find useful patterns

- **Prediction** based on past history
  - Predict if a credit card applicant poses a good credit risk, based on some attributes (income, job type, age, ..) and past history
  - Predict if a pattern of phone calling card usage is likely to be fraudulent

- Some examples of prediction mechanisms:
  - **Classification**
    - Given a new item whose class is unknown, predict to which class it belongs
  - **Regression** formulae
    - Given a set of mappings for an unknown function, predict the function result for a new parameter value

Data Mining

- **Descriptive Patterns**
  - **Associations**
    - Find books that are often bought by "similar" customers
    - If a new such customer buys one such book, suggest the others too
  - Associations may be used as a first step in detecting **causation**
    - E.g., association between exposure to chemical X and cancer
  - **Clusters**
    - E.g., typhoid cases were clustered in an area surrounding a contaminated well
    - Detection of clusters remains important in detecting epidemics

Classification Rules

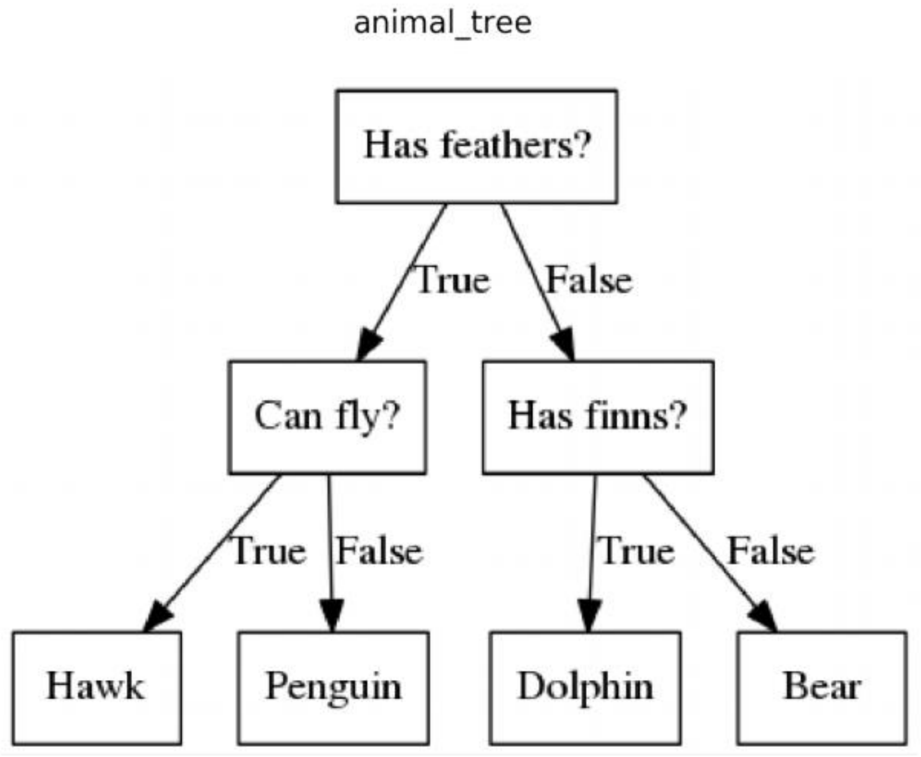
- Classification rules help assign new objects to classes
  - E.g., given a new automobile insurance applicant, should he or she be classified as low risk, medium risk or high risk?

- Classification rules for above example could use a variety of data, such as educational level, salary, age, etc.
  - $\forall$ person P, P.degree = masters **and** P.income > 75,000 $\Rightarrow$ P.credit = excellent
  - $\forall$ person P, P.degree = bachelors **and** (P.income $\geq$ 25,000 and P.income $\leq$ 75,000) $\Rightarrow$ P.credit = good

- Rules are not necessarily exact: There may be some misclassifications

- Classification rules can be shown compactly as a decision tree

Decision Tree



Construction of Decision Trees

- **Training set**: A data sample in which the classification is already known

- **Greedy** top down generation of decision trees
  - Each internal node of the tree partitions the data into groups based on a **partitioning attribute**, and a **partitioning condition** for the node
  - **Leaf** node:
    - All (or most) of the items at the node belong to the same class, or
    - All attributes have been considered, and no further partitioning is possible

Best Splits

- Pick best attributes and conditions on which to partition

- The purity of a set S of training instances can be measured quantitatively in several ways
  - Notation: Number of classes = *k*, number of instances = |S|, fraction of instances in class *i* = p_i

- The **Gini** measure of purity is defined as:

$$\text{Gini (S)} = 1 - \sum_{i- 1}^{k} p^2{}_i$$

  - When all instances are in a single class, the Gini value is 0
  - It reaches its maximum (of 1 – 1/*k*) if each class the same number of instances

Best Splits

- Another measure of purity is the **entropy** measure, which is defined as:

$$\text{entropy}(S) = -\sum_{i- 1}^{k} p_i log_2 p_i$$

- When a set S is split into multiple sets S_i, I = 1, 2, …, r, we can measure the purity of the resultant set of sets as:

$$\text{purity}(S_1, S_2, \ldots.., S_r) = \sum_{i=1}^{r} \frac{|S_i|}{|S|} \text{purity}(S_i)$$

- The information gain due to particular split of S into S_i, i = 1, 2, …., r

- **Information-gain** $(S, \{S_1, S_2, \ldots., S_r) = \text{purity}(S) - \text{purity}(S_1, S_2, \ldots S_r)$

Best Splits

- Measure of “cost” of a split:

$$\text{Information-content}\ (S, \{S_1, S_2, \ldots.., S_r\})) = -\sum_{i-1}^{r} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

- **Information-gain ratio** = $\dfrac{\text{Information-gain}\ (S, \{S_1, S_2, \ldots\ldots, S_r\})}{\text{Information-content}\ (S, \{S_1, S_2, \ldots.., S_r\})}$

- The best split is the one that gives the maximum information gain ratio

Finding Best Splits

- Categorical attributes (with no meaningful order):
  - Multi-way split, one child for each value
  - Binary split: Try all possible breakup of values into two sets, and pick the best

- Continuous-valued attributes (can be sorted in a meaningful order)
  - Binary split:
    - Sort values, try each as a split point
      - E.g., if values are 1, 10, 15, 25, split at ≤1, ≤ 10, ≤ 15
    - Pick the value that gives best split

  - Multi-way split:
    - A series of binary splits on the same attribute has roughly equivalent effect

Decision-Tree Construction Algorithm

```
Procedure GrowTree (S)
    Partition (S);

Procedure Partition (S)
    if ( purity (S ) > $\delta_p$ or |S| < $\delta_s$ ) then
        return;
    for each attribute A
        evaluate splits on attribute A;
    Use  best split found (across all attributes) to partition S into $S_1$, $S_2$, …., $S_r$,
    for i = 1, 2, ….., r
        Partition ($S_i$ );
```

Next Lecture

Data Analytics: Data Warehousing, Data Mining

Thank you for your attention...

Any question?

Contact:
Department of Information Technology, NITK Surathkal, India
6th Floor, Room: 13
Phone: +91-9477678768
E-mail: shrutilipi@nitk.edu.in