DATA INTEGRATION

CS561-SPRING 2012
WPI, MOHAMED ELTABAKH
DATA INTEGRATION

• Motivation
  • Many databases and sources of data that need to be integrated to work together
  • Almost all applications have many sources of data

• Data Integration
  • Is the process of integrating data from multiple sources and probably have a single view over all these sources
    • And answering queries using the combined information
  • Integration can be **physical** or **virtual**
    • **Physical**: Coping the data to warehouse
    • **Virtual**: Keep the data only at the sources
• Data integration is also valid within a single organization
  • Integrating data from different departments or sectors

Diagram:
- Synthesized Information
- Financial Application
- Shipping and Distribution Application
- CRM Application
- Order Management Application
- Contract Management Application
HETEROGENEITY PROBLEMS

• The main problem is the heterogeneity among the data sources

• Source Type Heterogeneity
  • Systems storing the data can be different
HETEROGENEITY PROBLEMS

• Communication Heterogeneity
  • Some systems have web interface others do not
  • Some systems allow direct query language others offer APIs

• Schema Heterogeneity
  • The structure of the tables storing the data can be different (even if storing the same data)
HETEROGENEITY PROBLEMS

• **Data Type Heterogeneity**
  • Storing the same data (and values) but with different data types
  • E.g., Storing the phone number as *String* or as *Number*
  • E.g., Storing the name as *fixed length* or *variable length*

• **Value Heterogeneity**
  • Same logical values stored in different ways
  • E.g., ‘Prof’, ‘Prof.’, ‘Professor’
HETEROGENEITY PROBLEMS

• **Semantic Heterogeneity**
  - Same values in different sources can mean different things
  - E.g., Column ‘Title’ in one database means ‘Job Title’ while in another database it means ‘Person Title’
REASONS OF HETEROGENEITY

- Schemas
- Generalization
  - Specialization
- Aggregation
- Typing
- Completeness

- Model

- Syntactical
- Data "Conflicts"
- Semantic

- Values
- Cognitive
- Taxonomy
- Completeness

- Cognitive

- Schemas
- Generalization
  - Specialization
- Aggregation
- Typing
- Completeness

- Model

- Syntactical
- Data "Conflicts"
- Semantic

- Values
- Cognitive
- Taxonomy
- Completeness

- Cognitive

MODELS OF DATA INTEGRATION

• Federated Databases

• Data Warehousing

• Mediation
FEDERATED DATABASES

- Simplest architecture
- Every pair of sources can build their own mapping and transformation
- Source X needs to communicate with source Y → build a mapping between X and Y
  - Does not have to be between all sources (on demand)

### Advantages
1. If many sources and only very few are communicating

### Disadvantages
1. If most sources are communicating ($n^2$ mappings)
2. If sources are dynamic (need to change many mappings)
DATA WAREHOUSING

- Very common approach
- Data from multiple sources are **copied and stored** in a warehouse
  - Data is materialized in the warehouse
- Users can then query the warehouse database only

**ETL: Extract-Transform-Load process**

- ETL is totally performed outside the warehouse
- Warehouse only stores the data
DATA WAREHOUSING: SYNCHRONIZATION

• How to synchronize the data between the sources and the warehouse???

• Two approaches
  • Complete rebuild
    • Periodically re-build the warehouse from the sources (e.g., every night or every week)
    • (+) The procedure is easy
    • (-) Expensive and time consuming
  • Incremental update
    • Periodically update the warehouse based on the changes in the sources
    • (+) Less expensive and efficient
    • (-) More complex to perform incremental update
    • (-) Requires sources to keep track of their updates

In both approaches the warehouse is not up-to-date at all times
DATA WAREHOUSING

Enterprise “Database”

Customers
Orders
Vendors
Etc...

Transactions

Data Warehouse

Simple queries

Copied, organized summarized

Data Mining

Complex and OLAP queries
TRADITIONAL DW ARCHITECTURE
MEDIATION

• Mediator is a virtual view over the data (it does not store any data)
  • Data is stored only at the sources

• Mediator has a virtual schema that combines all schemas from the sources

• The mapping takes place at query time
  • This is unlike warehousing where mapping takes place at upload time
Given a user query

• Query is mapped to multiple other queries
• Each query (or set of queries) are sent to the sources
• Sources evaluate the queries and return the results
• Results are merged (combined) together and passed to the end-user
MEDIATION: EXAMPLE

• Mediator Schema
  - Cust (ID, firstName, LastName, …)
  - CustPhones (ID, Type, PhoneNum, …)

• Source 1 Schema
  - Customers (ID, firstName, lastName, homePhone, cellPhone, …)

• Source 2 Schema
  - Customers (ID, FullName, …)
  - CustomersPhones (ID, Type, PhoneNum)

What if we need, first name, last name, and cell phone of customer ID = 100?
MEDIATION: EXAMPLE

- Mediator Schema
  
  ```
  Cust (ID, FirstName, LastName, ...)
  CustPhones (ID, Type, PhoneNum, ...)
  ```

  ```
  Select C.FirstName, C.LastName, P.PhoneNum
  From Cust C, CustPhones P
  Where C.ID = P.ID
  And C.ID = 100
  And P.Type = "celll";
  ```

  Map to source 1

- Source 1 Schema

  ```
  Customers (ID, firstName, lastName, homePhone, cellPhone, ...)
  ```

  ```
  Select firstName, lastName, cellPhone
  From Customers
  Where C.ID = 100;
  ```
MEDIATION: EXAMPLE

• Mediator Schema

  **Cust** (ID, FirstName, LastName, …)

  **CustPhones** (ID, Type, PhoneNum, …)

  ```sql
  SELECT C.FirstName, C.LastName, P.PhoneNum
  FROM Cust C, CustPhones P
  WHERE C.ID = P.ID
  AND C.ID = 100
  AND P.Type = "cell1";
  ```

  Function that returns the first name

• Source 2 Schema

  **Customers** (ID, FullName, …)

  **CustomersPhones** (ID, Type, PhoneNum)

  ```sql
  SELECT First(C.FullName), Last(C.FullName), P.PhoneNum
  FROM Customers C, CustomersPhones P
  WHERE C.ID = P.ID
  AND C.ID = 100
  AND P.Type = "cell1";
  ```

  Map to source 2
MEDIATION: WRAPPERS

- Usually wrappers are the components that perform the mapping of queries.

- One approach is to use templates with parameters:
  - If the mediator query matches a template, then replace the parameters and execute the query.
  - If no template is found, return empty results.

Designing these templates is a complex process because they need to be flexible and represent many queries.
MEDIATOR TYPES

• Global As View (GAV)
• Local As View (LAV)
GLOBAL AS VIEW (GAV)

• Mediator schema acts as a view over the source schemas

• Rules that map a mediator query to source queries

• Like regular views, what we see through the mediator is a subset of the available world

-- Limited view over the data

-- Cannot integrate/combine data from multiple sources to create new data beyond each source
LOCAL AS VIEW

• Sources are defined in terms of the global schema using expressions.

• Every source provides expressions on how it can generate pieces of the global schema.

• Mediator can combine these expressions to find all possible ways to answer a query.

-- Covers more data beyond each source individually
-- more complex than GAV
QUERY PROCESSING

Given a user query over the global schema:

• **Global AS view (GAV)**
  - Mediator follows the existing rules and templates to translate the query into source-specific queries
  - Send new queries to wrappers for execution

• **Global AS view (GAV)**
  - Mediator searches all possible expressions and how they can be combined to answer the given query
LAV EXAMPLE

• Assume the mediator has virtual relation $\text{Par}(c,p)$ that links child objects (c) with their parent objects (p)

• Source S1 can provide some info about $\text{Par}(c,p)$
  \[ V1(c,p) \leftarrow \text{Par}(c,p) \]

• Source S2 can provide info only about grandparents
  \[ V2(c,g) \leftarrow \text{Par}(c,p) \text{ And } \text{Par}(p,g) \]

Notice that V1 and V2 (which are the sources) are expressed using Par (which is the global view)
LAV EXAMPLE (CONT’D)

• Now given a query asking for great-grandparent

\[ Q(x,w): \text{Par}(x,y) \text{ And Par}(y,z) \text{ And Par}(z,w) \]

How to answer this query???

\[ Q(x,w): \text{V1}(x,y) \text{ And V1}(y,z) \text{ And V1}(z,w) \]
\[ + \]
\[ Q(x,w): \text{V2}(x,z) \text{ And V1}(z,w) \]
\[ + \]
\[ Q(x,w): \text{V1}(x,y) \text{ And V2}(y,w) \]

That is all possible answers from sources S1 and S2 for Q(x,w)
GAV vs. LAS

**GAV**
- (+) Simpler to design and implement
- (-) Narrows the view of all possible data that can be generated

**LAV**
- (+) More extensible. New sources just define what can they contribute to the global schema
- (-) More complex to design and implement
WHAT WE COVERED SO FAR …

- **Data integration** is the process of integrating data from multiple sources and answering queries using the combined information.

- **Models of Data Integration**
  - Federated Database
  - Data Warehouse
  - Mediators
    - Global As View (GAV)
    - Local As View (LAV)
ENTITY RESOLUTION

- Data coming from different sources may be different even if representing the same objects

- **Entity resolution** is the process of:
  - Figuring out which records represent the same thing
  - Linking relevant records together

    (John William, 252 Star rd., MA, 01609, 508-543-2222)
    
    (John Will., 252 Star road, MA, 01609, 508-543-2222)
    
    (John William, 252 Star rd., Massachusetts, 01609-3321, 508-543-2222)
    
    (John William, 252 Star rd., MA, 01609, (508) 543-2222)

    All of these are the same objects but they are not identical

    *If structure is different, it becomes even harder*
REASONS OF MISMATCHING

• Misspelling
  - “Smith”, “Smeth”, “Snith”

• Variant names, synonyms, and abbreviations
  - “St.”, “St”, “Street”…..“Prof”, “Professor”….”car”, “vehicle”

• Different systems

• Different domains
  - “YES/NO”, “1/0”, “T/F”
MECHANISMS FOR ENTITY RESOLUTION

- **Edit Distance**
  - Compare string fields using edit distance function
  - Can assign different weights to different fields

- **Normalization & Ontology**
  - Using a dictionary, replace all abbreviations with a standard forms
  - Ontology helps in synonyms

- **Clustering and Partitioning**
  - Run a clustering-based algorithm over the returned records
  - Tuples belonging to the same cluster can be further tested for matching
MERGING SIMILAR RECORDS

• How to merge similar records???

• In some cases, e.g., misspelling synonyms, it is possible to merge results

• In other cases, e.g., conflicts, there is no easy way to find the correct values
  • Report all the results we have

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Address</th>
<th>phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Susan Williams</td>
<td>123 Oak St.</td>
<td>818-457-1245</td>
</tr>
<tr>
<td>100</td>
<td>Susan Will.</td>
<td>456 Maple St.</td>
<td>818-457-1245</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Address</th>
<th>phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Susan Williams</td>
<td>{123 Oak St., 456 Maple St.}</td>
<td>818-457-1245</td>
</tr>
</tbody>
</table>
AUTOMATED DATA INTEGRATION

- Data integration requires a lot of manual effort
  - Data warehouse → designing and implementing the ETL module
  - Mediators → designing and implementing the wrappers
  - Federated database → designing and implementing the mapping modules (wrappers)

Can we automate this process ???
A Generic Framework for Integration

Consider several database schemas for different bookstores

- How to match their schemas automatically \( \leftarrow \text{schema matching techniques} \)
- How to find matching records \( \leftarrow \text{record linkage techniques} \)
- How to find errors, synonyms, etc. and correct them \( \leftarrow \text{data cleansing techniques} \)