Data Integration and Exchange
Traditional approach to databases

- A single large repository of data.
- Database administrator in charge of access to data.
- Users interact with the database through application programs.
- Programmers write those (embedded SQL, other ways of combining general purpose programming languages and DBMSs)
- Queries dominate; updates less common.
- DMBS takes care of lots of things for you such as
  - query processing and optimisation
  - concurrency control
  - enforcing database integrity
Traditional approach to databases cont’d

- This model works very within a single organisation that either
  - does not interact much with the outside world, or
  - the interaction is heavily controlled by the DB administrators

- What do we expect from such a system?
  1. Data is relatively clean; little incompleteness
  2. Data is consistent (enforced by the DMBS)
  3. Data is there (resides on the disk)
  4. Well-defined semantics of query answering (if you ask a query, you know what you want to get)
  5. Access to data is controlled
The world is changing

- The traditional model still dominates, but the world is changing.
- Many huge repositories are publicly available
  - In fact many are well-organised databases, e.g., imdb.com, the CIA World Factbook, many genome databases, the DBLP server of CS publications, etc etc etc)
- Many queries cannot be answered using a single source.
- Often data from various sources needs to be combined, e.g.
  - company mergers
  - restructuring databases within a single organisation
  - combining data from several private and public sources
Course info

- No text.
  - Because there is no text at this time...
- Slides will be posted on the course webpage:
  - http://homepages.inf.ed.ac.uk/libkin/teach/dataintegr08
- Tutorials by Lenzerini and Kolaitis (see links on the webpage)
- 3 assignments
- final exam
- Office hours: by appointment (usually works better for UG4)
Why do you need this course

• Databases are everywhere these days (> $2 \cdot 10^{10}$/year business — whatever that means today)

• Every enterprise has a database; they merge, combine data – hence data integration

• In addition, a lot of data is available on the web, but often one needs many sources to answer a query

• Hence (almost) everyone needs to integrate data

• Huge investment from leading companies, IBM, Oracle, Microsoft

• Very ad hoc solutions; but finally we understand what the real problems in data integration are, and have some solutions (but not all!)
Background

• Requirement: Database Systems (3rd year)
• or fluency in relational databases:
  ◦ relational model
  ◦ relational algebra/calculus
  ◦ SQL
• An understanding of the basic mathematical tools that serve as the foundation of computer science:
  ◦ basic set theory,
  ◦ graph theory,
  ◦ theory of computation,
  ◦ first-order logic.
Outline of the course

• Introduction to the problems of data integration and exchange. Key new components:
  ◦ incomplete information
  ◦ query rewriting
  ◦ certain answers

• Data integration scenarios:
  ◦ global-as-view, local-as-view, combined
  ◦ virtual vs materialized

• How to distinguish easy queries from hard queries?

• Query answering in data integration scenarios:
  ◦ view-based rewritings
Outline of the course cont’d

• Incomplete information in databases
  ◦ theory, tables, complexity
  ◦ practice (the ugly reality – SQL)
  ◦ Open and closed worlds

• Data exchange: settings, source-to-target constraints, solutions

• Data exchange query answering:
  ◦ conjunctive (select-project-join) queries
  ◦ full relational algebra queries
    closed vs open worlds
Outline of the course cont’d

- **Data exchange: XML data**
  - tree patterns
  - consistency problems
  - query answering

- **Schema management:**
  - composition, other operations, schema evolution

- **Inconsistent databases, repairs, query answering**

- **If time permits:** ranking queries
Query answering from multiple sources

- Data resides in several different databases
- They may have different structures, different access policies etc
- Our view of the world may be very different from the view of the databases we need to use.
- Only portions of the data from some database could be available.
- That is, the sources do not conform to the schema of the database into which the data will be loaded.
What industry offers now: ETL tools

• ETL stands for Extract–Transform–Load
  ◦ Extract data from multiple sources
  ◦ Transform it so it is compatible with the schema
  ◦ Load it into a database

• Many self-built tools in the 80s and the 90s; through acquisition fewer products exist now

• The big players – IBM, Microsoft, Oracle – all have their ETL products; Microsoft and Oracle offer them with their database products.

• A few independent vendors, e.g. Informatica PowerCenter.

• Several open source products exist, e.g. Clover ETL.
ETL tools

• Focus:
  ◦ Data profiling
  ◦ Data cleaning
  ◦ Simple transformations
  ◦ Bulk loading
  ◦ Latency requirements

• What they don’t do yet:
  ◦ nontrivial transformations
  ◦ query answering

• But techniques now exist for interesting data integration and for query answering – and we shall learn them.

• They soon will be reflected in products (IBM and Microsoft are particularly active in this area)
Data profiling/cleaning

• Data profiling: gives the user a view of data:
  ○ Samples over large tables
  ○ statistics (how many different values etc)
  ○ Graphical tools for exploring the database

• Cleaning:
  ○ Same properties may have different names
    e.g. Last_Name, L_Name, LastName
  ○ Same data may have different representations
    • e.g. (0131)555–1111 vs 01315551111,
    • George Str. vs George Street
  ○ Some data may be just wrong
Data transformation

- Most transformation rules tend to be simple:
  - Copy attribute LName to Last_Name
  - Set age to be current_year – DOB

- Heavy emphasis on industry specific formats

- For example, Informatica B2B Data Exchange product offers versions for Healthcare and Financial services as well as specialised tools for formats including:
  - MS Word, Excel, PDF, UN/EDIFACT (Data Interchange For Administration, Commerce, and Transport), RosettaNet for B2B, and many specialised healthcare and financial form.

- These are format/industry specific and have little to do with the general tasks of data integration.
Data integration, scenario 1

GLOBAL SCHEMA

QUERY: Q?
Data integration

GLOBAL SCHEMA

QUERY: Q?
Data integration

Answer to Q is obtained by querying the views $V_1, \ldots, V_n$. 

GLOBAL SCHEMA

QUERY: Q?
Data integration, query answering

- We have our view of the world (the Global Schema).
- We can access (parts of) databases $DB_1, \ldots, DB_n$ to get relevant data.
- It comes in the form of views, $V_1, \ldots, V_n$.
- Our query against the global schema must be reformulated as a query against the views $V_1, \ldots, V_n$.
- The approach is completely virtual: we never create a database that conforms to the global schema.
Data integration, query answering, a toy example

- List courses taught by permanent teaching staff during Winter 2007
- We have two databases:
  - $D_1$(name, age, salary) of permanent staff
  - $D_2$(teacher, course, semester, enrollment) of courses
- $D_1$ only publishes the value of the name attribute
- $D_2$ does not reveal enrollments
- The views:
  \[
  V_1 = \pi_{\text{name}}(D_1) \\
  V_2 = \pi_{\text{teacher}, \text{course}, \text{semester}}(D_2)
  \]
- Next step: establish correspondence between attributes name of $V_1$ and teacher of $V_2$
To answer query, we need to import the following data:

\[ V_1 \]

\[ W_2 = \sigma_{\text{semester}='\text{Winter 2007}'}(V_2) \]

Answering query:

\[ \{ \text{course} \mid \exists \text{name, sem } V_1(\text{name}) \land W_2(\text{name, course, sem}) \} \]

Or, in relational algebra

\[ \pi_{\text{course}}(V_1 \bowtie_{\text{name=teacher}} W_2) \]
Toy example, lessons learned

• We don’t have access to all the data

• Some human intervention is essential (someone needs to tell us that teacher and name refer to the same entity)

• We don’t run a query against a single database. Instead, we
  ○ run queries against different databases based on restrictions they impose
  ○ get results to use them locally
  ○ run another query against those results
Toy example, things getting more complicated

- Find informatics permanent staff who taught during the Winter 2007 semester, and their phone numbers

- We have additional personnel databases:
  - an informatics database $D_3(\text{employee, phone, office})$, and
  - a university-wide database $D_4(\text{employee, school, phone})$
  - for simplicity, assume all this information is public

- Now we have a choice:
  - use $D_3$ to get information about phones
  - use $D_4$ to get information about phones
  - use both $D_3$ and $D_4$ to get information about phones
Toy example cont’d

• First, we need some human involvement to see that employee, name, and teacher refer to the same category of objects

• If one uses $D_3$, then the query is

$$\{name, \text{ phone} \mid \exists \text{sem}, \text{office} \ V_1(name) \wedge W_2(name, \text{course}, \text{sem}) \wedge D_3(name, \text{phone}, \text{office})\}$$

• If one uses $D_4$, then the query is

$$\{name, \text{ phone} \mid \exists \text{sem}, \text{school} \ V_1(name) \wedge W_2(name, \text{course}, \text{sem}) \wedge D_4(name, \text{school}, \text{phone})\}$$

• But what if one uses both $D_3$ and $D_4$?
Toy example cont’d

• We could insist on the phone number being:
  ○ in either $D_3$ or $D_4$
  ○ in both $D_3$ and $D_4$, but not necessarily the same
  ○ in both $D_3$ and $D_4$, and the same in both databases

• One can write queries for all the cases, but which one should we use?

• New lessons:
  ○ databases that are being integrated are often inconsistent
  ○ query answering is by no means unique – there could be several ways to answer a query
  ○ different possibilities for answering queries are a result of inconsistencies and incomplete information
Toy example cont’d

• Suppose phone numbers in $D_3$ and $D_4$ are different.
• What is a sensible query answer then?
• A common approach is to use certain answers – these are guaranteed to be true.
• Another question: what if there is no record at all for the phone number in $D_3$ and $D_4$?
• Then we have an instance of incomplete information.
A different scenario

• So far we looked at virtual integration: no database of the global schema was created.

• Sometimes we need such a database to be created, for example, if many queries are expected to be asked against it.

• In general, this is a common problem with data integration: materialize vs federate.

• Materialize = create a new database based on integrating data from different sources.

• Federate = the virtual approach: obtain data from various sources and use them to answer queries.
Virtual vs Materialization

- A common situation for the materialization approach: merger of different organizations.
- A common situation for the federated approach: we don’t have full access to the data, and the data changes often.
Common tasks in data integration

• How do we represent information?
  ◦ Global schema, attributes, constraints
  ◦ data formats of attributes
  ◦ reconciling data from different sources
  ◦ abbreviations, terminology, ontologies

• How do we deal with imperfect information?
  ◦ resolve overlaps
  ◦ handling missing data
  ◦ handling inconsistencies
Common tasks in data integration cont’d

• How do we answer queries?
  ○ what information is available?
  ○ Can we get the answer?
  ○ if not, what is the semantics of query answering?
  ○ Is query answering feasible?
  ○ Is it possible to compute query answers at all?
  ○ If now, how do we approximate?

• Materialize or federate?
Common tasks in data integration cont’d

- Do it from scratch or use commercial tools?
  - many are available (just google for “data integration”)
  - but do we fully understand them?
  - lots of them are very ad hoc, with poorly defined semantics
  - this is why it is so important to understand what really happens in data integration
Data Exchange

Source Schema $S$           Target Schema $T$
Data Exchange

Source Schema $S$  

Target Schema $T$
Data Exchange

Query over the target schema: \( Q \)

How to answer \( Q \) so that the answer is consistent with the data in the source database?
Data exchange vs Data integration

Data exchange appears to be an easier problem:

• there is only one source database;
• and one has complete access to the source data.

But there could be many different target instances.

Problem: which one to use for query answering?
When do we have the need for data exchange

- A typical scenario:
  - Two organizations have their legacy databases, schemas cannot be changed.
  - Data from one organization 1 needs to be transferred to data from organization 2.
  - Queries need to be answered against the transferred data.
Data exchange – towards multiple instances

- A simple example: we want to create a target database with the schema

  \[\text{Flight(city1,city2,aircraft,departure,arrival)}\]

  \[\text{Served(city,country,population,agency)}\]

- We don’t start from scratch: there is a source database containing relations

  \[\text{Route(source,destination,,departure)}\]

  \[\text{BG(country,city)}\]

- We want to transfer data from the source to the target.
Data exchange – relationships between the source and the target

How to specify the relationship?

Semantics?? For example, arrows from city – is the meaning and or or?
Data exchange – relationships between the source and the target

- Formal specification: we have a *relational calculus query* over both the source and the target schema.
- The query is of a restricted form, and can be thought of as a sequence of rules:

  \[
  \text{Flight}(c_1, c_2, \_ , \text{dept}, \_ ) \leftarrow \ \text{Route}(c_1, c_2, \text{dept})
  \]

  \[
  \text{Served}(\text{city}, \text{country}, \_ , \_ ) \leftarrow \ \text{Route}(\text{city}, \_ , \_ ), \ \text{BG}(\text{city}, \text{country})
  \]

  \[
  \text{Served}(\text{city}, \text{country}, \_ , \_ ) \leftarrow \ \text{Route}(\_ , \text{city}, \_ ), \ \text{BG}(\text{city}, \text{country})
  \]
Data exchange – targets

• Target instances should satisfy the rules.
• What does it mean to satisfy a rule?
• Formally, if we take:

\[
\text{Flight}(c_1, c_2, \_ , \_ , \text{dept}, \_ ) \leftarrow \text{Route}(c_1, c_2, \text{dept})
\]

then it is satisfied by a source \( S \) and a target \( T \) if the constraint

\[
\forall c_1, c_2, d (\text{Route}(c_1, c_2, d) \rightarrow \exists a_1, a_2 (\text{Flight}(c_1, c_2, a_1, d, a_2)))
\]

• This constraint is a relational calculus query that evaluates to true or false
Data exchange – targets

• What happens if there is no value for some attributes, e.g. *aircraft*, *arrival*?

• We put in *null values* or some real values.

• But then we may have multiple solutions!
Data exchange – targets

Source Database:

<table>
<thead>
<tr>
<th>Source</th>
<th>Destination</th>
<th>Departure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edinburgh</td>
<td>Amsterdam</td>
<td>0600</td>
</tr>
<tr>
<td>Edinburgh</td>
<td>London</td>
<td>0615</td>
</tr>
<tr>
<td>Edinburgh</td>
<td>Frankfurt</td>
<td>0700</td>
</tr>
</tbody>
</table>

Look at the rule

\[
\text{Flight}(c1, c2, \_, \text{dept}, \_) \quad \text{:-} \quad \text{Route}(c1, c2, \text{dept})
\]

The right hand side is satisfied by

\[
\text{Route}(\text{Edinburgh, Amsterdam, 0600})
\]

But what can we put in the target?
Data exchange – targets

Rule:  \( \text{Flight}(c1, c2, \_\_\_, dept, \_\_\_) \leftarrow \text{Route}(c1, c2, dept) \)

Satisfied by:  \( \text{Route}(\text{Edinburgh}, \text{Amsterdam}, 0600) \)

Possible targets:

- \( \text{Flight}(\text{Edinburgh}, \text{Amsterdam}, \bot_1, 0600, \bot_2) \)
- \( \text{Flight}(\text{Edinburgh}, \text{Amsterdam}, \text{B737}, 0600, \bot) \)
- \( \text{Flight}(\text{Edinburgh}, \text{Amsterdam}, \bot, 0600, 0845) \)
- \( \text{Flight}(\text{Edinburgh}, \text{Amsterdam}, \text{B737}, 0600, 0845) \)

They \textbf{all} satisfy the constraints!
Data exchange – queries

• Now consider two queries:
  ◦ $Q_1$: Is there a flight from Edinburgh to Amsterdam that departs before 7am?
  ◦ $Q_2$: Is there a flight from Edinburgh to Amsterdam that arrives before 9am?

• What is the difference?
  ◦ $Q_1$ can be answered with certainty: in every solution we have a tuple
    \[ \text{Flight}(\text{Edinburgh}, \text{Amsterdam}, \_ , 0600, \_) \]
  ◦ $Q_2$ cannot be answered with certainty: in some solutions we don’t have a tuple
    \[ \text{Flight}(\text{Edinburgh}, \text{Amsterdam}, a, t_1, t_2) \] with $t_2$ earlier than 9am.

• Our goal is to find certain answers.
Data exchange – queries

• But computing certain answers requires checking seemingly an infinite number of databases!
• How else can we do it?
• Create a good target instance $T_{good}$ so that:
  ○ for a query $Q$ we can define a query $Q_r$ (its rewriting)
  ○ that satisfies the property:
    $$\text{certain answers to } Q = Q_r(T_{good})$$

• Questions:
  ○ can we always find such a $T_{good}$ and a rewriting algorithm $Q \mapsto Q_r$?
  ○ and if not, what restrictions do we impose on data exchange settings and/or queries?
Inconsistencies in databases

- If we integrate data, we shall always have inconsistencies:
  - One database says that we have John Smith with salary 20K in office 100
  - another says that we have John Smith with salary 30K in office 100
  - and the database must satisfy a key constraint: the name field is a key.

- Hence if we put
  
<table>
<thead>
<tr>
<th>Name</th>
<th>Office</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Smith</td>
<td>100</td>
<td>20K</td>
</tr>
<tr>
<td>John Smith</td>
<td>100</td>
<td>30K</td>
</tr>
<tr>
<td>....</td>
<td>....</td>
<td>....</td>
</tr>
</tbody>
</table>

  in our database, we have inconsistent data.
Inconsistencies in databases: query answering

- $Q_1$: Does John Smith sit in office 100?
- $Q_2$: Does John Smith make 20K?

**Difference:**
- $Q_1$ can be answered with certainty;
- $Q_2$ cannot be.

- What does it mean to answer a query with certainty?
- If we repair a database so that it satisfies the constraints, the answer is true – no matter how we repair it.
Inconsistencies in databases: query answering

• In our example, two ways to repair:

<table>
<thead>
<tr>
<th>Name</th>
<th>Office</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Smith</td>
<td>100</td>
<td>20K</td>
</tr>
<tr>
<td>....</td>
<td>....</td>
<td>....</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Office</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Smith</td>
<td>100</td>
<td>30K</td>
</tr>
<tr>
<td>....</td>
<td>....</td>
<td>....</td>
</tr>
</tbody>
</table>

• $Q_1$ is always true, $Q_2$ is not.
• But – the number of repairs could be very large (exponential – why?).
• Hence prohibitively expensive query answering algorithm.
• Question: when can query answering be made efficient?
• Perhaps it involves a rewriting of the original query.
• The key idea: query rewriting to obtain certain answers.
Schema mappings

- Last subject we deal with in this course.
- Still the least understood, but extremely important.
- Schema evolution: schema changes over time.
- Question – how to transfer data?
- Single step – data exchange.
- But what if we go through many steps? How do we transfer data, how do we answer queries?
Schema mappings

• Two data exchange scenarios:
  \[
  \text{Schema1} \quad \text{Schema2} \quad \text{Constraints12} \\
  \text{Schema2} \quad \text{Schema3} \quad \text{Constraints23}
  \]

• Suppose we know how to move data from \text{Schema1} to \text{Schema2}, and then from \text{Schema2} to \text{Schema3}?

• Can we describe this by a single set of schema constraints:
  \[
  \text{Schema1} \quad \text{Schema3} \quad \text{Constraints13}
  \]

• This turns out to be a very nontrivial task, but it occurs very often in database schema management.

• And there are other operations – inverse, for example:
  \[
  (\text{Schema1} \quad \text{Schema2} \quad \text{Constraints12}) \\
  \downarrow
  (\text{Schema2} \quad \text{Schema1} \quad \text{Constraints21})
  \]