Data integration (Heterogeneous data sources)

Technologies for Information Systems
November, 11 2011

Exercise

There are 3 data sources with different schemas and data models
• UNIVERSITY-DB (DS1)
  • Relational data source
• TAX-POSITION (DS2)
  • XML data source
• COMPUTER-SCIENCE (DS3)
  • Object-oriented data source

Differently from previous exercises, we are dealing with heterogeneous data sources
• Different data models

Exercise - main steps

In the presence of heterogeneous data sources, we need to change some steps of the integration approach:
1. Schema analysis and identification of the sources data models
2. Reverse engineering (conceptual models)
3. Identification and resolution of conflicts
4. Conceptual models integration
5. Choice of the target data model (for global conceptual schema translation)
6. Source schemas translation to the target data model (by means of adapters)
7. Conceptual model translation
8. Definition of data views (mappings)

UNIVERSITY-DB (DS1)

DEPARTMENT(dept-code, dept-name, budget)
RESEARCH_STAFF(email, name, dept-code, s-code)
SCHOOL_MEMBER(email, name, school, year)
SESSION(s-code, session-name, length, room-code)
ROOM(room-code, seats-number, notes)

N.B.:
• School members are both research staff members and students
• Sessions are intended as courses
• Each session is related to only one research staff member
• People names are encoded as "name$surname"

TAX-POSITION (DS2)

N.B.:
• Assume the content of s-code as key

COMPUTER-SCIENCE (DS3)

CS_PERSON (first-name, last-name)
PROFESSOR:CS_PERSON (belongs_to:DIVISION, rank)
STUDENT:CS_PERSON (year, takes:set<COURSE>, rank, email)
DIVISION (code, description, address:LOCATION)
LOCATION (city, street, number, country)
COURSE (course-name, taught-by:PROFESSOR)
**Data model identification**

- DS1: relational
- DS2: XML
- DS3: Object Oriented

**Reverse engineering**

**DS1 - conceptual schema**

**DS2 - conceptual schema**

**DS3 - conceptual schema**

**Conflict analysis**

- Synonyms:
  - department ↔ division
  - course ↔ session
  - research staff ↔ professor
  - school member ↔ person
  - student is a subset of person
- Location ↔ room
- Cardinality conflicts:
  - Cardinality conflict between professor and course
    - DS1: relationship session (course) - research staff (professor): one to one
    - DS3: relationship course - (professor): one to many
- Key-conflicts:
  - Person
    - DS1: email → p-code
    - DS2: s-code → p-code
    - DS3: (first name, last-name) → p-code

**GS conceptual schema**
**GS logical schema**

Choice of the target data model
- We select the relational data model

**Source schemata translation**

- **DS1**
  - no translation needed

- **DS2**
  - DS2.STUDENT(s_code, name, school-name, tax-fee)

- **DS3**
  - DS3.PERSON(f1(first-name, last-name), role, year, email, division-code, name)
  - DS3.COURSE(course-code, professor-first-name, professor-last-name)
  - DS3.TAKES(student-first-name, student-last-name, course-name)
  - DS3.DIVISION(name, description)
  - DS3.LOCATION(street, number, city, country, division-code)

---

**Logical mappings**

Suppose that the sources' schemata are very stable and there exists only the given sources in the system. What kind of integration strategy do you propose?

**GAV: Global as view**

```sql
CREATE VIEW GS.PERSON(p_code, first-name, last-name, email, role, year, rank, school-name, tax-fee, dept-code) AS
SELECT email, f1(name), f2(name), email, "research-staff", year, NULL, school, NULL, dept-code
FROM DS1.RESEARCH_STAFF, DS1.SCHOOL_MEMBER
WHERE DS1.RESEARCH_STAFF.email=DS1.SCHOOL_MEMBER.email
UNION
SELECT email, f1(name), f2(name), email, "student", year, NULL, school, NULL, NULL
FROM DS1.SCHOOL_MEMBER
WHERE email NOT IN (SELECT email FROM DS1.RESEARCH_STAFF)
UNION
SELECT s_code, f1(name), f2(name), email, "student", NULL, NULL, school-name, tax-fee, NULL
FROM DS2.STUDENT
```
Logical mappings

UNION

SELECT concat(first-name, last-name), first-name, last-name, email, role, year, rank, NULL, NULL, division-code
FROM DS3.PERSON

• f1(.) --> returns the first part of the name (that we assume to be the only first name)
• f2(.) --> returns the second part of the name
• concat(..., ...) --> concatenates the first and the last name to build a valid key

We need to perform the same steps for each entity in the global schema.

Adapters - DS2

• We need an adapter able to offer a relational view over the XML database
  • e.g., by mixing the OO language and XQuery

• The following piece of code will generate the java-inspired code for adding to a result set the result of the GAV mappings between GS.PERSON and DS2.STUDENT

  for $x$ in doc("tax-positions.xml")/listOfStudents/student
  return resultSet.add($x/school-name), myfunct.f1($x/name), email, "student", null, null, $x/school-name), ($x/tax-fee), null);

Adapters - DS3

• We need an adapter able to offer a relational view over the Object-Oriented database

• The following piece of code will generate the java-inspired code for adding to a result set the result of the GAV mappings between GS.PERSON and DS3.PERSON

  for each person |
  if (person instanceof professor)
  resultSet.add(professor.getFirstName(), professor.getLast_name(),
  professor.getFirstName(), professor.getLastName(), null, "professor",
  null, professor.getRank(), null, null,
  professor.getDivision().getDivisionCode());
  else |
  resultSet.add(student.getFirstName(), student.getLastName(),
  student.getFirstName(), student.getLastName(), student.getEmail(),
  "student", student.getYear(), student.getRank(), null, null);
  }