Exam Simulation

Technologies for Information Systems
November, 11 2011

Exam simulation

An important TV director wants to set up a new TV show with a great anchor and VIP guests. Unfortunately, the TV company database contains only data about TV shows and the director needs to integrate them with the data of another database concerning a wider variety of shows.

Data source DS1

The first source is a standard relational database

SHOW (title, date, type)

CAST (show-title, show-date, cast-member-name, appearance-duration)

CAST-MEMBER (name, address, fee-per-hour, agent-name)

AGENT (name, company, phone-number)

ROLE (name, type)

Primary keys are underlined, while foreign keys are italicized.

Remarks:
• The duration of the show is expressed in hours
• The fees are expressed in dollars
• Dates are expressed as yyyy-mm-dd
• Type can be (guest|cast-member|technician)

Data source DS2

The second source is the modern web information system of the TV company which relies on an XML database

<!DOCTYPE tv-shows-DB SYSTEM "showsDB.dtd">

<!ELEMENT tv-shows-DB (tv-show*)>

<!ELEMENT tv-show (anchor+, schedule+)>  

<!ELEMENT anchor EMPTY>

<!ELEMENT schedule (day+)>  

<!ELEMENT day (guest+)>  

<!ELEMENT guest EMPTY>

<!ATTLIST tv-show title CDATA #REQUIRED>

<!ATTLIST tv-show edition CDATA #REQUIRED>

<!ATTLIST tv-show duration CDATA #IMPLIED>

<!ATTLIST anchor name CDATA #REQUIRED>

<!ATTLIST anchor engagement-fee CDATA #IMPLIED>

<!ATTLIST anchor agent-phone-contact CDATA #IMPLIED>

<!ATTLIST day date CDATA #REQUIRED>

<!ATTLIST day special CDATA (0,1) "0">

<!ATTLIST guest name CDATA #REQUIRED>

<!ATTLIST guest role CDATA #IMPLIED>

<!ATTLIST guest engagement-fee CDATA #IMPLIED>

<!ATTLIST guest address CDATA #IMPLIED>

DS2: an example of a valid XML file

<?xml version="1.0" encoding="UTF-8"?>

<tv-shows-DB>

<tv-show title="The David Lettermann Show" edition="2007/2008" duration="1">

<anchor name="David Lettermann" engagement-fee="60,000"/>

<schedule>

<day date="2007-09-14" special="0">

<guest name="Mike Tyson" engagement-fee="200,000"/>

</day>

<day date="2007-09-21">

<guest name="Hillary Clinton" engagement-fee="30,000"/>

<guest name="Monika Lewinsky" engagement-fee="100,000"/>

</day>

</schedule>

</tv-show>

</tv-shows-DB>
Data source DS2

Remarks:
- Consider the required attributes as primary keys
- The duration of the show is expressed in hours
- The fees are expressed in dollars
- Dates are expressed as yyyy-mm-dd.

Exam

1. Propose a data integration solution which is able to represent the data of both data sources without information loss, discussing which integration technique you suggest for this scenario (GAV, LAV, GLAV, etc.). You may assume the stability of the data sources.
2. Reengineer the sources to obtain their conceptual models, listing the most relevant mismatches between the two schemata and propose a solution.
3. Propose a global conceptual model as integration solution.
4. Present the mappings (in terms of SQL or Xquery views) needed to map the two data sources to the merged schema you have proposed occasions.

5. (OPTIONAL) Write the following query on the merged schema and show its translation in terms of the data sources:
   "Select all the guests and their fee per hour of TV shows that have been on air in special occasions"

Proposed solution

- We propose a data integration solution based on a merged-schema expressed in the relational data model.
- As the data sources are known and stable we will build the DIS (Data integration system) using GAV mappings.
- This will also ease the query processing for the query proposed at point 5.

Reverse engineering

DSI - conceptual schema

We assume that a cast member may be in the database even if (s)he has never been included in a show, hence assumption for the role.

Reverse engineering

DSI - conceptual schema

We don't have a direct relationship between the role of a cast member and the show.
Reverse engineering

DS2 – conceptual schema

Conflict analysis

1. The whole DS2 represent a subset of the reality represented by DS1 (TV-SHOWS are a subset of SHOWS). Since we are interested only in tv shows we do not need all the data contained in DS1.

2. The way DS1 is designed is however more general, we use that conceptual model as a basis for the integrated schema. For example, the anchor is a cast member as the others.

3. Fees are represented as fees per hour in DS1 and in fees per engagement in DS2. We need an appropriate conversion.

4. In DS2 agent is missing we keep the modeling solution of DS1 and we need a function to generate a unique identifier.

5. Special date are present only in DS2. We can use DS2 tuples to reconstruct the missing information in DS1. We assume DS2:DAY as a “calendar” table containing all the days with indication of special days (e.g. Christmas).

6. The type attribute is no longer necessary. The global schema contains only tv shows.

GS conceptual schema

Proposed solution

- As the data sources are known and stable we will build the DIS (Data integration system) using GAV mappings.
- This will also ease the query processing for the query proposed at point 5.
GS logical schema

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

GS logical schema

TV-SHOW (title, date, edition, duration, special)  
CAST-MEMBER (name, address, fee-per-hour, agent-name)  
AGENT (name, company, phone-number)  
CAST (show-title, show-date, cast-member-name, appearance-duration)  
CAST-ROLE (cast-member-name, role-name)  
ROLE (name, type)

Source schemata translation

• DS1  
  no translation needed

• DS2  
  TV-SHOW (title, edition, duration)  
  ANCHOR (name, engagement-fee, agent-contact-phone)  
  GUEST (name, role, address, engagement-fee)  
  DAY (date, special)  
  SHOW-ANCHOR (anchor-name, show-name, show-edition)  
  SCHEDULE (show-name, show-edition, guest-name, show-date)

GAV mappings

CREATE VIEW GS:TV-SHOW (title, date, edition, special) AS  
{  
  SELECT DS1:SHOW.title, DS1:SHOW.date, null, null, DS2:DAY:special  
  FROM DS1:SHOW, DS2:DAY  
  WHERE DS1:SHOW.type = "tv-show" AND DS1:SHOW.date = DS2:DAY.date  
}  
UNION  
SELECT DS2:TV-SHOW.title, DS2:SCHEDULE.date, null, null,  
DS2:TV-SHOW.duration, DS2:DAY:special  
FROM DS2:TV-SHOW, DS2:SCHEDULE, DS2:DAY  
WHERE DS2:SCHEDULE.show-name = DS2:TV-SHOW.title AND  
DS2:SCHEDULE.show-date = DS2:DAY.date  
}

GAV mappings

CREATE VIEW GS:CAST-MEMBER (name, address, fee-per-hour, agent-name) AS  
{  
  SELECT DS1:CAST-MEMBER.name, DS1:CAST-MEMBER.address,  
  DS1:CAST-MEMBER.fee-per-hour, DS1:CAST-MEMBER.agent-name  
  FROM DS1:CAST-MEMBER, DS1:SHOW, DS1:CAST  
  WHERE DS1:CAST-MEMBER.name = DS1:CAST-member-name AND  
  DS1:CAST.show-title = DS1:SHOW.title AND  
  DS1:CAST.show-date = DS1:SHOW.date AND  
  DS1:SHOW.type = "tv-show"  
}  
UNION  
SELECT DS2:GUEST.name, DS2:GUEST.address,  
DS2:GUEST.engagement-fee,DS2:TV-SHOW.duration, null  
FROM DS2:GUEST, DS2:TV-SHOW, DS2:SCHEDULE  
WHERE DS2:GUEST.name = DS2:SCHEDULE.guest-name AND  
DS2:TV-SHOW.title = DS2:SCHEDULE.show-title AND  
DS2:TV-SHOW.date = DS2:SCHEDULE.show-date  
}

GAV mappings

UNION  
SELECT DS2:ANCHOR.name, null,  
DS2:ANCHOR.agent-contact-phone,DS2:TV-SHOW.duration,  
null,DS2:TV-SHOW.title,DS2:SHOW-ANCHOR.show-title AND  
DS2:SHOW-ANCHOR.show-date  
FROM DS2:ANCHOR, DS2:TV-SHOW, DS2:SHOW-ANCHOR  
WHERE DS2:ANCHOR.name = DS2:SHOW-ANCHORanchor-name AND  
DS2:TV-SHOW.title = DS2:SHOW-ANCHOR.show-title AND  
DS2:TV-SHOW.date = DS2:SHOW-ANCHOR.show-date  
}

Query on the global schema

"Select all the guests and their fee per hour of TV shows that have been on air in special occasions"

SELECT GS:CAST-MEMBER.name, GS:CAST-MEMBER.fee-per-hour, GS:ROLE.type  
FROM GS:CAST-MEMBER, GS:CAST-ROLE, GS:ROLE  
WHERE GS:CAST-MEMBER.name=GS:CAST-ROLE.member-name AND  
GS:CAST-ROLE.role-name=GS:ROLE.name AND  
GS:ROLE.type = "guest" AND  
GS:CAST-MEMBER.name in (  
  SELECT GS:CAST.name  
  FROM GS:CAST, GS:TV-SHOW  
  WHERE GS:CAST.show-title = GS:TV-SHOW.title  
  AND GS:CAST.show-date = GS:TV-SHOW.date  
  AND GS:TV-SHOW.special = "1")